



Leveraging Deep Learning and Word Embeddings to Detect Dish Names in Consumer Reviews

استغلال التعلم العميق وتضمين الكلمات في استخراج أسماء أطباق الطعام
المذكورة في تقييمات المستهلكين

by

DEENA YOUNIS ABOKHASHAN

**A thesis submitted in partial fulfilment
of the requirements for the degree of
DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE**

at

The British University in Dubai

February 2024



Leveraging Deep Learning and Word Embeddings to Detect Dish Names in Consumer Reviews

استغلال التعلم العميق وتضمين الكلمات في استخراج أسماء أطباق الطعام المذكورة في تقييمات المستهلكين

by

DEENA YOUNIS ABOKHASHAN

**A thesis submitted to the Faculty of Engineering & IT
in partial fulfilment of the requirements for the degree of
DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE**

at

The British University in Dubai

February 2024

Thesis Supervisor

Prof Sherief Abdallah

Approved for award:

Name
Designation

Name
Designation

Name
Designation

Name
Designation

Date: _____

Declaration

I warrant that the content of this research is the direct result of my own work and that any use made in it of published or unpublished copyright material falls within the limits permitted by international copyright conventions.

I understand that a copy of my research will be deposited in the University Library for permanent retention.

I hereby agree that the contents of this dissertation for which I am author and copyright holder may be copied and distributed by The British University in Dubai for the purposes of research, private study or education and that The British University in Dubai may recover from purchasers the costs incurred in such copying and distribution, where appropriate.

I understand that The British University in Dubai may make a digital copy available in the institutional repository.

I understand that I may apply to the University to retain the right to withhold or to restrict access to my thesis for a period which shall not normally exceed four calendar years from the congregation at which the degree is conferred, the length of the period to be specified in the application, together with the precise reasons for making that application.



Signature of the student

Copyright and Information to Users

The author whose copyright is declared on the title page of the work has granted to the British University in Dubai the right to lend his/her research work to users of its library and as an open source and to make partial or single copies for educational and research use.

The author has also granted permission to the University to keep or make a digital copy for similar use and for the purpose of preservation of the work digitally.

Multiple copying of this work for scholarly purposes may be granted by either the author, the Registrar or the Dean only.

Copying for financial gain shall only be allowed with the author's express permission.

Any use of this work in whole or in part shall respect the moral rights of the author to be acknowledged and to reflect in good faith and without detriment the meaning of the content, and the original authorship.

Abstract

Named Entity Recognition (NER) is crucial for extracting entities from unstructured text, offering significant insights for businesses through customer review analysis. This study fills a gap in recognizing dish names from customer reviews, as existing literature mainly addresses food entity recognition in recipe datasets and lacks annotated datasets for this specific NER task. Domain adaptation and deep learning approaches like BiGRUs and CNNs remain underexplored. The research proposes a deep learning NER framework to accurately identify dish names in customer reviews with efficient computational resource use. In addition to the existing dataset, *MenuNER* dataset, an annotated dataset, *ReviewsDB*, was created from Yelp reviews for evaluation. Initial experiments revealed a notable performance drop in domain adaptation from food names in recipe datasets to dish names in reviews, with the F1-score nearly 50% lower. A comparative analysis of 53 deep learning models using various word embeddings, including Glove, Word2Vec, and Bert variants, showed that a simple architecture with a single-layer Bidirectional Gated Recurrent Unit (BiGRU) and Conditional Random Field (CRF) layer achieved the best performance, with an F1-score of 93.07% using glove-twitter-100 embeddings in the *MenuNER* dataset. Additionally, a two-layer BiGRU with a CNN and CRF achieved an F1-score of 82.40% on the *ReviewsDB* dataset. The study attributes performance differences to variability in annotation lengths and the broader range of terms in *ReviewsDB*. In conclusion, the proposed NER framework, leveraging pre-trained embeddings, provides a valuable tool for the food industry to analyze customer feedback and enhance customer satisfaction.

خلاصة

يلعب التعرف على الكيانات المسماة (NER) أمر حيوي لاستخراج الكيانات من النصوص غير المنظمة، مما يوفر رؤى مهمة للأعمال من خلال تحليل مراجعات العملاء. تملأ هذه الدراسة فجوة في التعرف على أسماء الأطباق من مراجعات العملاء، حيث أن الأبحاث الحديثة تركز بشكل أساسي على التعرف على كيانات الطعام في مجموعات البيانات الخاصة بالوصفات وتفتقر إلى مجموعات البيانات المشروحة لمراجعات العملاء الخاصة بمهمة NER. اُضيف إلى ذلك لا يزال تكييف النطاق ونهج التعلم العميق مثل BiGRUs و CNNs غير مستكشفة بشكل كافٍ. يقترح البحث إطار عمل NER يعتمد على التعلم العميق للتعرف بدقة على أسماء الأطباق في مراجعات العملاء مع استخدام فعال لموارد الحوسبة. بالإضافة إلى مجموعة البيانات الموجودة، تم إنشاء مجموعة بيانات مشروحة، ReviewsDB، من مراجعات Yelp للتقييم. كشفت التجارب الأولية عن انخفاض ملحوظ في الأداء عند تكييف النطاق من أسماء الطعام في مجموعات البيانات الخاصة بالوصفات إلى أسماء الأطباق في مراجعات العملاء، مع انخفاض درجة F1 بنحو 50%. كما أظهر تحليل مقارن لـ 53 نموذجًا من نماذج التعلم العميق باستخدام العديد من التضمينات اللفظية، بما في ذلك Glove و Word2Vec و Bert، أن بنية بسيطة بطبقة واحدة من (BiGRU) وطبقة (CRF) حققت أفضل أداء، بدرجة F1 بلغت 93.07% باستخدام تضمينات glove-twitter-100 في مجموعة بيانات MenuNER. بالإضافة إلى ذلك، حقق نموذج BiGRU بطبقتين مع CNN و CRF درجة F1 بلغت 82.40% على مجموعة بيانات ReviewsDB. تعزو الدراسة اختلافات الأداء إلى تباين في أطوال التعليقات المشروحة والنطاق الأوسع للمصطلحات في ReviewsDB. في الختام، يوفر إطار NER المقترح، الذي يستفيد من التضمينات المدربة مسبقًا، أداة قيمة لصناعة الطعام لتحليل ملاحظات العملاء وتحسين رضا العملاء.

Dedication

Ten years may have passed since you left this world, yet the echo of your influence remains in my heart. Your departure was not merely a sorrowful goodbye, but a rebirth for me. This journey stands as a testament to the enduring lessons you instilled in me, and the strength derived from your memory. Though you are not here to see it, I dedicate this work to you, hoping that somewhere, you are looking with pride.

Always in my heart dad.

Acknowledgements

I am profoundly grateful to God for providing me with strength and guidance throughout this journey. “All praise is due to Allah by whose grace good deeds are accomplished”. I would like to extend my deepest gratitude to my supervisor, Dr. Sherief Abdallah, for his support and guidance throughout my research journey. His invaluable insights, patience, and mentorship have not only shaped this work but also my growth as a researcher. I am eternally grateful to my family, especially my beloved mother and dear brother Ahmed. Your continuous support and patience are the main support for me to finish my research. No words can express how much your love and belief in me have pushed me forward, even during the toughest moments. Your faith and encouragement have been the inspiration that guided me through my research journey, and for that, I am forever grateful. To my siblings Basma, Alia, Abdullah, and Mohamed, each of you holds a unique and irreplaceable place in my heart. Thanks for your continuous support. My dearest friends deserve special mention. Their companionship, words of encouragement, and occasional distractions have been the source of relief during the most demanding phases of my PhD journey. To Habab and Seema your friendship has been invaluable, and I am blessed to have friends like you by my side. To Varduhi, I can't thank you enough for always being there. I won't forget how much you've supported me. To Alia, when things got hard, you kept me on track. I'm grateful for your advice and support. Asia, you know how much you mean to me. You never let me down, always understood what I was going through, and have been a big part of my journey. Thanks for always being there. Lastly, I would like to thank all those who have contributed to this work, including the administrative staff of BUiD.

Table of Contents

Chapter 1 : Introduction.....	1
1.1 Research Motivation	1
1.2 Problem Statement	5
1.3 Research Questions	7
1.4 Aims and Objectives	8
1.5 Significance and Contributions	10
1.6 Thesis Structure	13
Chapter 2 : Background and Literature Review.....	15
2.1 Text Mining in Customer Reviews	15
2.1.1 Restaurant Customer Reviews.....	19
2.2 Named Entity Recognition Methodologies	23
2.2.1 Rule-Based NER Systems	24
2.2.2 Feature-Based Machine Learning Methods.....	25
2.2.3 Unsupervised Machine Learning-Based Methods.....	27
2.2.4 Deep Learning Based Methods.....	28
2.3 Named Entity Recognition in Food Domain.....	31
2.3.1 Named Entity Recognition to Detect Food Mentions.....	31
2.3.2 Named Entity Recognition to Detect Dish Mentions	63
2.4 Deep Learning Neural Networks.....	68
2.4.1 Recurrent Neural Network (RNN).....	68
2.4.2 Long Short-Term Models (LSTM).....	71
2.4.3 Gating Recurrent Unit (GRU)	73
2.4.4 Convolutional Neural Models - (CNN)	75
2.4.5 Conditional Random Field (CRF)	79

2.5 Language Models and Word Embeddings	81
2.5.1 Neural Probabilistic Language Model	83
2.5.2 Contextual Representation Model	96
2.6 Character Embeddings	103
2.7 POS and Syntactic Information	105
2.8 Research Gaps	108
Chapter 3 : Research Methodology	112
3.1 Our ReviewsDB Dataset	112
3.2 Other Datasets	118
3.2.1 FoodBase Dataset	119
3.2.2 MenuNER Dataset	121
3.2.3 Comparative Analysis of Customer Review Datasets	125
3.3 NER Model Evaluation and Optimization	126
3.3.1 Model Evaluation	127
3.3.2 Model Optimization and Regularization	135
3.4 Experimented Models	144
3.4.1 Baseline Experiments: Detecting Food Names in FoodBase Dataset	144
3.4.2 Domain Adaptation from Food Names Detection in FoodBase to Dish Name Detection in Customer Reviews	154
3.4.3 Training the BiLSTM Models on MenuNER Dataset Using Fixed Pre-Trained Word Embeddings	154
3.4.4 Enhancing BiLSTM-CRF Model Performance by Incorporating the POS Tags and Trainable Word Embeddings	155
3.4.5 The Utilization of Contextual Word Embeddings (Bert)	156
3.4.6 Experiment with Different Deep Learning Model Architectures	161

3.4.7 Experiment Settings.....	182
Chapter 4 : Experimental Results and Analysis.....	184
4.1 Analysis of the BiLSTM-CRF Model	184
4.1.1 Results and Analysis of Replicated Baseline Experiments by Cenikj et al. (2020).....	184
4.1.2 Results and Analysis of Domain Adaptation.....	187
4.1.3 Results and Analysis of Dish Name Detection in MenuNER Using Fixed Pre-Trained Embeddings	188
4.1.4 Results and Analysis of Models Enhanced with Part-of-Speech Tags	191
4.1.5 Results and Analysis of BiLSTM-CRF Model Using Contextual Word Embeddings Using BERT	195
4.1.6 Summary of Results and Analysis	198
4.2 Analysis of Deep Learning Models with Trainable Word Embeddings	201
4.2.1 Results and Analysis of Replicated Baseline Experiments by Syed and Chung (2021)....	202
4.2.2 Results and Analysis of Deep Learning Models with Single Recurrent Layer Type	206
4.2.3 Results and Analysis of Deep Learning Models with Recurrent Layers	211
4.2.4 Results and Analysis of Deep Learning Models with Recurrent and CNN Layers	215
4.2.5 Summary of Results and Analysis	220
4.3 Comparative Analysis: Our Proposed Models Vs. Syed and Chung (2021) Models .	227
4.4 Analysis of Experimental Results on the ReviewsDB Dataset	230
4.5 Comparing and Analyzing Model Performance: MenuNER Vs. ReviewsDB	233
Chapter 5 : Conclusion and Future Work	236
5.1 Research Summary	236
5.2 Key Findings	237
5.3 Research Contributions	242
5.3.1 Theoretical Contributions	242

5.3.2 Practical Implications	243
5.4 Limitations	244
5.5 Future Work	245
References	248
Appendices	266
Appendix A. Part-of-Speech Tags (POS)	266
Appendix B. Customer Reviews Extraction from Yelp	269
Appendix C. Further Results and Analysis of Research Experiments	277
C.1 Further Analysis of the BiLSTM-CRF Model Results	277
C.2 Further Analysis of Deep Learning Models With Trainable Word Embeddings on MenuNER Dataset.....	288
C.3 Further Analysis of Deep Learning Models With Trainable Word Embeddings on ReviewsDB Dataset.....	347

List of Tables

Table 2-1: Results Summary of the NER Tasks	46
Table 2-2: The F1 Scores for the Three Classes Detected by the FooDCoNER Model	56
Table 2-3: The Results of Detecting Food Entities by the Four Experimented Models	60
Table 2-4: Best Performance Results of Syed and Chung (2021) Models.....	67
Table 2-5: BERT Models	101
Table 3-1: Dish Names Distribution in Each Fold of Our ReviewsDB Dataset	115
Table 3-2: Number of Dish Names/Part of Dish Names in the Original MenuNER Dataset .	121
Table 3-3: Original Class Distribution in Original MenuNER Datasets.....	123
Table 3-4: Dish Names Distribution in Each Fold of MenuNER Dataset	124
Table 3-5: The Interpretation of TP, FN, and FP	130
Table 3-6: Word Embedding Models Used in Experiments	152
Table 3-7: BiLSTM-CRF with CharEmbd Model Performance When Using BERT Embeddings.....	158
Table 3-8: Bert Embeddings Variants	159
Table 3-9: All Utilized Word Embeddings in the Study	161
Table 3-10: Experiment With Different Parameter Values of the CNN Model.....	179
Table 3-11: All Experimented Deep Learning Models in Our Study.	180
Table 4-1: Macro Average F1 Scores of the Trained Baseline Models on FoodBase Dataset	185
Table 4-2: Macro Average F1 Scores of BiLSTM-CRF and BiLSTM-CRF With CharEmbd Models.....	187
Table 4-3: Macro-Average Evaluation Metric Scores of BiLSTM-CRF and BiLSTM-CRF With CharEmbd Models Using Fixed Pre-Trained Word Embeddings.....	189

Table 4-4: Macro-Average Evaluation Metric Scores of BiLSTM-CRF With CharEmbd Model Using POS Features as Additional Input.....	192
Table 4-5: Performance of BiLSTM-CRF With CharEmbd Model With the Use of Different Variations of Bert Embeddings	196
Table 4-6: Top Performance Results of BiLSTM-CRF with Character Embedding and POS Features as Additional Inputs to the NER Model	197
Table 4-7: Optimal Experimental Results of Syed and Chung (2021) Models	202
Table 4-8: Impact of Batch Size Combinations on Experiment Outcomes	204
Table 4-9: Comparison of Original Results By Syed and Chung (2021) and Our Replicated Experiment Outcomes	205
Table 4-10: Best Achieved Results When Experiment With BiLSTM-CRF and 2BiLSTM-CRF Models	206
Table 4-11: Comparative Performance of BiLSTM(50N)-CRF Architectures With Rmsprop and Adam Optimizers Across Various Word Embeddings.....	208
Table 4-12: Best Achieved Results When Experiment With BiGRU-CRF and 2BiGRU-CRF Models.....	210
Table 4-13: Best Achieved Results When Experiment With 1BiLSTM+1BiGRU-CRF Models	212
Table 4-14: Best Achieved Results When Experiment With 2BiLSTM-2BiGRU-CRF Models	214
Table 4-15: Best Achieved Results When Experiment With 1BiLSTM-CNN-CRF, 1BiGRU-1CNN-CRF , and 1BiLSTM-1BiGRU-1CNN-CRF Models	216
Table 4-16: Best Achieved Results When Experiment With 2BiGRU-1CNN-CRF Models	219

Table 4-17: The Top Four F1-Scores Achieved By Our Experimented Models on MenuNER

Dataset 225

Table 4-18: Comparison of State-of-the-Art Results: Our Proposed Models Vs. Syed and

Chung (2021)..... 228

Table 4-19: The Top Four F1-Scores Achieved by Our Experimented Models on ReviewsDB

Dataset 231

List of Figures

Figure 2-1: Schematic Representation of a Bidirectional RNN Model	70
Figure 2-2: Convolutional Neural Network Layer Architecture	77
Figure 3-1: BiLSTM-CRF Model Architecture	147
Figure 3-2: BiLSTM-CRF With CharEmbd Model Architecture	149
Figure 3-3: BiLSTM-CRF With CharEmbd Model Utilizing POS Features as Additional Input	156
Figure 3-4: 1BiLSTM-CRF Model Architecture	164
Figure 3-5: 2BiLSTM-CRF Model Architecture	165
Figure 3-6: 1BiGRU-CRF Model Architecture	167
Figure 3-7: 2BiGRU-CRF Model Architecture	168
Figure 3-8: 1BiLSTM-1BiGRU-CRF Model Architecture.....	170
Figure 3-9: 2BiLSTM-1BiGRU-CRF Model Architecture.....	172
Figure 3-10: 1BiLSTM-1CNN-CRF Model Architecture	174
Figure 3-11: 1BiGRU-1CNN-CRF Model Architecture.....	175
Figure 3-12: 1BiLSTM-1BiGRU-1CNN-CRF Model Architecture.....	176
Figure 3-13: 2BiGRU-1CNN-CRF Model Architecture.....	178
Figure 4-1: Macro Average F1 Score of BiLSTM-CRF With CharEmbd Model Before and After Adding POS Features as Additional Input.....	194

List of Abbreviations

ABSA: Aspect-Based Sentiment Analysis

AdaGrad: Adaptive Gradient Optimizer

BERT: Bidirectional Encoder Representations from Transformers

BiGRU: Bidirectional Gating Recurrent Unit

BiLSTM: Bidirectional Long Short-Term Memory

BioBERT: Bidirectional Encoder Representations from Transformers for Biomedical Text Mining

BPTT: Backpropagation Through Time

BuTTER: Bidirectional LSTM for Food Named-Entity Recognition

CBOW: Continuous Bag of Words

CharCNN: Character-level Convolutional Neural Network

CNN: Convolutional Neural Network

CPU: Central Processing Unit

CRF: Conditional Random Fields

DistilBERT: Distilled Bidirectional Encoder Representations from Transformers

Electra: Efficiently Learning an Encoder that Classifies Token Replacements Accurately

ELMo: Embeddings from Language Models

FFNN: Feedforward Neural Network

FINER: Food Ingredient NER

FOBI: Food-Biomarker Ontology

FoodCoNER: Food and Dietary Constituents Named Entity Recognition

FoodOn: Food Ontology

GloVe: Global Vectors for Word Representation

GPT: Generative Pretrained Transformer

GPU: Graphics Processing Unit

HMM: Hidden Markov Model

LDA: Latent Dirichlet Allocation

LSA: Latent Semantic Analysis

LUCK: Language Understanding with Knowledge-based Entity

MaxEnt: Maximum Entropy Model

MEMMs: Maximum Entropy Markov Models

MLM: Masked Language Modelling

NCBO: National Center for Biomedical Ontology

NEL: Named Entity Linking

NER: Named Entity Recognition

NLI: Natural Language Inference

NLP : Natural Language Processing

NLTK: Natural Language Toolkit

NLU: Natural Language Understanding

NNLM: Neural Net Language Model

NSP: Next Sentence Prediction

OOV: Out-of-Vocabulary word

POS: Part of Speech

PU Learning: Positive and Unlabeled Learning

QA: Question Answering

ReLU: Rectified Linear Unit

RMSprop: Root Mean Square Propagation

RNE: Recurrent Network-Based Ensemble Model

RNN: Recurrent Neural Network

RNNLM: Recurrent Neural Net Language Model

Roberta: Robustly Optimized BERT Pre-training Approach

SciBERT: Scientific BERT

SGD: Stochastic Gradient Descent

SNOMED CT: Systematized Nomenclature of Medicine Clinical Terms

SVM: Support Vector Machine

TF-IDF: Term Frequency - Inverse Document Frequency

UCREL: University Centre for Computer Corpus Research on Language

USAS: UCREL Semantic Analysis System

USDA: United States Department of Agriculture

Word2Vec: Word to Vector

XML: Extensible Markup Language

Chapter 1 : Introduction

This chapter is structured into six sections. It begins by tackling the challenges of extracting dish names from customer reviews, thereby establishing the core problem our research addresses. Then, we clarify the specific research questions we aim to resolve, along with the main goals and objectives of our study. After that, the key findings and the significant contributions of our research are discussed followed by a detailed outline of the thesis structure, offering a clear guide to the content and organization of the subsequent chapters.

1.1 Research Motivation

Named Entity Recognition (NER) serves as a cornerstone in the field of Natural Language Processing (NLP), allowing machines to understand, interpret, and label unstructured textual data into predefined classes. Over time, the use of NER has evolved beyond its conventional fields, gaining increased significance in diverse industry-specific use cases (Li et al., 2020). One sector that stands to gain considerably from specialized NER systems is the restaurant and food industry. Customer reviews on platforms like Yelp and social media channels are increasingly influencing the reputation of restaurants and affecting their business decisions. These reviews can offer invaluable insights when dish names are identified and extracted (Bai et al., 2023). Possible valuable applications are:

- *Improvement of the Restaurant's Menu:* Understanding which dishes are frequently praised or mentioned can help a restaurant refine its menu, focusing on its strengths or identifying areas for improvement. It can lead to a more curated and successful menu selection that aligns with customer tastes, potentially increasing sales, and customer satisfaction (Jia 2019).

- *Enhancing Customer Experience:* Analyzing customer preferences allows for a more personalized dining experience. Restaurants can highlight popular dishes to new customers or tailor their specials and promotions around the dishes that receive the most positive mentions. This not only improves service but can also enhance the overall customer experience, making it more likely that customers will return (Ali et al., 2022).

- *Better Analytics Results:* From an analytics viewpoint, extracting dish names and correlating them with customer sentiments can yield more accurate and actionable insights. It helps in understanding trends and the impact of new menu items, enabling data-driven decision-making (Sharma et al., 2023).

- *Dish Recommendations:* Utilizing past reviews to recommend dishes can be a powerful tool for upselling and improving service. It could be implemented in online ordering systems that can make informed suggestions, potentially leading to a more enjoyable dining experience (Peters & Remaud, 2020).

However, extracting this critical information from customer reviews presents several challenges. For one, customer reviews are generally informal, containing a variety of writing styles, misspellings, and abbreviations (Bai et al., 2023). The following review exemplifies the challenges in extracting data from informal online reviews:

“Omg, where do I strt? Istly, the ambiance was on fleek. Very IG-worthy. But, service? Uh-uh. Waited like 4ever for our table even tho we had a reso. Food was BOMB tho! Ordered the Calamari Fristo thingy. TBH, idk if I'd go bck”.

The review is inherently casual in tone, evident from the opening with *"Omg, where do I strt?"* and the mention that *"the ambiance was on fleek"*. This casualness also showcases itself through a mix of expressions, questions, and statements that add variety to the writing style, such as the statement *"Very IG-worthy"*. Additionally, the review has several misspellings and typographical errors, such as using *"strt"* instead of *"start"* and *"tho"* in place of *"though"*. Notably, the dish name *"Calamari Friso"* is incorrect; the actual name is *"Calamari Fritto"*. Moreover, abbreviations and slang terms are there such as the use of *"1stly"* for *"firstly"*, *"IG"* as a shorthand for Instagram, and *"BOMB"* to describe something very good.

Another layer of complexity is added by the absence of standardized naming conventions for dishes and the introduction of new, often creative, dish names by customers. For example,

"..... Had dinner at Fusion Haven last night. I was recommended the "Midnight Marvel" by a fellow diner. To my surprise, it was just a unique take on shrimp tacos..."

In this review, the name *"Midnight Marvel"* is non-standard, making it challenging for a NER system to accurately recognize and classify it as a specific dish name.

Furthermore, context also plays a significant role. The meaning of a word can often changes depending on the surrounding words or sentences, adding to the difficulty in accurately identifying dish names (Khurana et al., 2023). The following review is an example:

"Stopped by after a friend's recommendation. Loved the setting; felt like I was back in Italy. Tried the "Carbonara". At first, I thought it referred to the coal-themed decor in

the corner, but it turned out to be a delightful pasta dish! However, the actual Carbonara could have used more pecorino. The music was a bit too loud, making our conversation a tad challenging. Oh, and if you're in the mood for a drink, try the "Rome". Strange name for a cocktail, but it's a trip worth taking!"

In the review, "Carbonara" initially appears to be a reference to the restaurant's decor, which might involve coal or something resembling it. However, in the next mention, it's clear that "Carbonara" is a pasta dish. Similarly, "Rome" might typically be understood as the capital city of Italy, but in this context, it's a cocktail. This review highlights the importance of context in accurately interpreting and extracting information, especially dish names, from customer feedback.

On the other hand, the literature reveals that while there is rich research on food entity recognition from structured or semi-structured sources like recipe databases, there is an obvious gap when it comes to the specific task of extracting dish names from customer reviews. Most existing NER systems are general-purpose and are not designed to cope with the unique set of challenges presented by this task, such as the work of Popovski et al. (2019), Cenikj et al. (2020), Diwan et al. (2020), Stojanov et al. (2021), Perera et al. (2022), and Komariah and Sin (2022). Furthermore, the lack of specialized, annotated datasets that are based on customer reviews for training and testing models geared toward dish name extraction are very rare (Syed & Chung, 2021).

Moreover, the existing literature presents a limited exploration of certain deep learning model architectures for the NER task. Researchers like Cenikj et al. (2020), Stojanov et al. (2021), Syed and Chung (2021), Cenikj et al. (2022), and Fan and Hunter (2023) have

employed BiLSTM-CRF deep learning models in their work. However, there is a noticeable absence of experimentation with diverse architectures, such as Bidirectional Gated Recurrent Units (BiGRUs) and Convolutional Neural Networks (CNNs).

Furthermore, the literature does not leverage the advantages of domain adaptation, particularly when transitioning from detecting food names in recipe datasets to identifying dish names in customer reviews. As suggested by Smařdu et al. (2022), using the knowledge acquired by a deep learning model from one domain to another can potentially lead to good model performance.

A review of the literature also highlights that the state-of-the-art model proposed by Syed and Chung (2021) for detecting dish names in customer reviews is quite complex, requiring a substantial number of computational resources. Their top-performing model, boasting an F1-score of 92.5, mandated training the BERT (Bidirectional Encoder Representations from Transformers) model on a vast corpus of unannotated Yelp customer reviews in an unsupervised fashion. Subsequently, the generated BERT embeddings were used as input for the computationally intensive BiLSTM-CRF model to identify dish names.

1.2 Problem Statement

This research confronts the critical challenges of developing an advanced Named Entity Recognition (NER) system, specifically designed to efficiently and accurately interpret the multifaceted language used in customer reviews for the precise identification and extraction of dish names. Firstly, the linguistic variety and informality found in online customer reviews create significant difficulties. These reviews frequently contain colloquial language, idiomatic expressions, and culturally specific references, making it difficult for

traditional NER systems to recognize and correctly classify dish names. The variability in spelling, grammar, and syntax further aggravates this challenge, as NER systems typically rely on standard language structures for accurate identification.

Secondly, the dynamic nature of culinary terminology, particularly in the context of customer reviews, poses another layer of complexity. Restaurants frequently introduce new dishes, and customers often use creative or subjective descriptions, leading to an ever-evolving lexicon that NER systems struggle to keep up with. This issue is compounded by the lack of standardized naming conventions for dishes, making it challenging to establish a comprehensive and reliable dataset for training NER models. In fact, when our research commenced, no annotated customer reviews dataset focusing on dish names existed.

Moreover, the context in which dish names are mentioned in reviews can be highly variable and subject to interpretation leading to a third challenge in the domain. A single word or phrase could refer to a dish, an ingredient, or something entirely unrelated, depending on the surrounding text. This ambiguity necessitates advanced contextual analysis capabilities in NER systems, which current models may not adequately provide.

In addition, the current literature and existing NER methodologies demonstrate a gap in effectively handling these complexities, especially in the critical absence of specialized, annotated datasets for customer reviews. Also, there's limited exploration in applying diverse deep learning model architectures, like Bidirectional Gated Recurrent Units (BiGRU's) and Convolutional Neural Networks (CNNs), for this specific task.

Finally, the computational intensity and resource requirements of existing state-of-the-art models further limit their applicability in practical settings. Many high-performing NER

systems demand extensive computational resources, making them impractical for real-time analysis or use by small to medium-sized enterprises in the restaurant industry.

Thus, the main problem that this research seeks to address is the development of an adaptable, efficient, and accurate deep learning-based NER system that can overcome these linguistic, terminological, contextual, and computational challenges. This system should be capable of effectively processing the unique language used in customer reviews, continuously adapting to new culinary terms, accurately interpreting context, and doing so with manageable computational demands and an innovative approach in terms of model architecture. The end goal is to address the practical challenges to empower the restaurant and food industry with the capability to employ customer feedback for improved business outcomes and heightened customer satisfaction and to address the academic gap in this field of NER research.

1.3 Research Questions

Towards the achievement of the main thesis goal, the following research questions have been formulated. These questions aim to address gaps in the existing literature, inspect methodological choices, and offer avenues for practical applications.

The questions are:

- **Research Question 1 (RQ1):** How does the performance of NER models trained on our manually annotated *ReviewsDB* dataset compared to those trained on the benchmark *MenuNER* dataset in the specific task of dish name detection?

- **Research Question 2 (RQ2):** How does domain adaptation influence the performance of a model trained on a recipe dataset for food name detection when applied to the domain of dish name detection in customer reviews?
- **Research Question 3 (RQ3):** How does the choice of pre-trained word embeddings like Glove, Word2Vec, and FastText influence the effectiveness of NER models designed for dish name identification?
- **Research Question 4 (RQ4):** How can various neural network architectures, including BiLSTM, BiGRU, and CNN, be effectively utilized in developing a computationally efficient deep learning NER model for the accurate detection of dish names in customer reviews compared to the computationally expensive state of art model?
- **Research Question 5 (RQ5):** What are the implications of employing contextualized embeddings like BERT for Named Entity Recognition in the specific area of dish name detection? Does it offer a significant advantage over non-contextualized embeddings?

1.4 Aims and Objectives

The main aim of this research is to advance the field of Named Entity Recognition (NER) within the context of the restaurant and food industry by developing a deep learning model that effectively identifies dish names from customer reviews with moderate computational resources. The objectives are shaped around our research questions that target the identification of optimal strategies and methodologies for NER in this specialized domain.

Each objective is designed to address the gaps and challenges identified in the existing literature.

The research objectives are:

1. To develop and create a specialized annotated dataset, *ReviewsDB*, for training and testing NER models in the context of detecting dish names in customer reviews.
2. To evaluate the performance of NER models trained on our manually annotated dataset, *ReviewsDB*, against those trained on the existing benchmark dataset, *MenuNER*, with the specific task of detecting dish names from customer reviews. This comparison will help determine the effectiveness of our curated dataset in improving the accuracy of dish name recognition.
3. To assess the impact of domain adaptation techniques on the accuracy of NER models initially trained on structured recipe datasets and then applied to the unstructured domain of customer reviews. This investigation will shed light on the adaptability and versatility of NER systems across different domains.
4. To analyze the influence of different pre-trained word embeddings, such as GloVe, Word2Vec, and FastText, on the performance of NER systems. By analyzing these embeddings, the research intends to identify which offer the greatest benefits in terms of model effectiveness for the task at hand.

5. To investigate and compare the effectiveness of different neural network architectures, including BiLSTM, BiGRU, and CNN, for accurate dish name detection, focusing on computational efficiency.

6. To develop a deep learning NER model that achieves a balance between high accuracy in dish name detection and moderate computational resource usage.

7. To evaluate the potential advantages of using contextualized embeddings, such as BERT, over non-contextualized embeddings in the NER task of dish name detection. This would help ascertain if the additional contextual information captured by these embeddings translates into significantly improved model performance.

By systematically addressing these objectives, this thesis will not only contribute to the academic understanding of NER in a specialized context but also provide actionable insights for the restaurant and food industry. Ultimately, the research will aim to enable businesses to harness the power of NER for enhanced customer insight and satisfaction, thereby driving strategic and data-informed decision-making.

1.5 Significance and Contributions

The significance of this research is not confined to academic circles; it has tangible implications for the restaurant and food industry. Automating the extraction of dish names from customer reviews allows for quicker, more accurate data analytics. Such analytics could inform various aspects of business strategy, from menu planning and inventory management to targeted marketing initiatives. Thus, the potential for enhancing customer satisfaction is

considerable, given that businesses would be better positioned to respond to feedback more effectively.

Therefore, the primary findings and contributions of this research can be summarized as follows:

1. An annotated dataset for dish names in restaurant reviews.

I created a dataset and named it as *ReviewsDB* by extracting 1000 restaurant reviews from the Yelp platform and personally annotating the mentions of dish names. This dataset was vital to the research. It was utilized to assess the effectiveness of various deep-learning models in detecting dish names. Moreover, by comparing results from *ReviewsDB* with the benchmark dataset, *MenuNER*, we gained insights into the Named Entity Recognition (NER) task where both datasets were used to train and evaluate different deep-learning models. Furthermore, we consider our dataset as a valuable resource for future research in the field.

2. A comprehensive comparative analysis of different word embeddings and their effect on dish names NER.

This study delved into a detailed comparison of various word embeddings and deep learning structures, specifically focusing on their impact on dish name detection in customer reviews. We experimented with multiple pre-trained embeddings such as Glove, Word2Vec, and FastText. Additionally, we included contextual Bert embeddings, testing four of its variants to explore their influence.

Contrary to the anticipation of Bert embeddings outperforming, the pre-trained embeddings displayed commendable results, often surpassing Bert's performance. The incorporation of BERT-based contextual embeddings did not lead to substantial advantages for the specific task of dish name detection in customer reviews unless it is integrated into computationally burdensome models requiring fine-tuning millions of parameters as illustrated by Syed and Chung's (2021) model. It's noteworthy that these pre-trained embeddings underwent additional fine-tuning during the NER model's training phase. The findings from the two utilized datasets indicated that among the embeddings, Glove, consistently produced the most accurate results in dish name detection across several models. This underscores the significance of carefully selecting pre-trained word embeddings when designing NER models.

3. A state-of-the-art detection of dish names in restaurant reviews using a simplified deep-learning model.

Through our in-depth research, we achieved the best performance in detecting dish names from customer reviews, surpassing existing benchmarks in the literature. Our model, utilizing *glove-twitter-100 embeddings*, recorded an impressive F-score of 93.07. We demonstrated that a more intricate deep learning model doesn't always yield superior results. In contrast, our straightforward model, composed of a single BiGRU layer with 100 hidden neurons, coupled with a CRF layer (termed 1BiGRU(100N)-CRF), delivered the superior result. Therefore, we believe that the deployment of our model to automatically detect dish names will be a valuable tool for the food industry where moderate computational resources are needed to run our model.

These contributions not only advance our understanding of the different challenges in named entity recognition within the domain of customer reviews but also provide valuable insights for future research and practical applications.

1.6 Thesis Structure

The remaining content of the thesis is structured as follows:

- Chapter 2: This chapter delves into the critical role of text mining in analyzing customer reviews. It features a comprehensive literature review, focusing on various Named Entity Recognition (NER) methodologies, including those specifically tailored for the food domain. The chapter identifies gaps and limitations in the existing literature and introduces a range of deep learning models that contribute to the design of our proposed solutions. Further, it explores the architecture of different language models and underscores the significance of character embeddings and Part-of-Speech (POS) tags in Natural Language Processing tasks.
- Chapter 3: This chapter provides a meticulous overview of our dataset, *ReviewsDB*, in addition to other datasets employed in our study, along with the specific evaluation metrics and optimization algorithms that guide our deep learning models. The chapter offers an in-depth explanation of the models experimented with, including their configurations, and intended outcomes.
- Chapter 4: This chapter is committed to presenting the findings of our empirical experiments. For each result, we provide a detailed analysis and discussion to interpret the implications and evaluate the performance of our models.

- Chapter 5: The final chapter synthesizes the key insights garnered from the research and outlines the study's contributions to the broader field. It concludes with recommendations for future research in NER domain, highlighting potential avenues for further exploration and development.

This revised outline offers a clearer, more organized way of presenting the content across the various chapters.

In summary, this thesis aims to make a fundamental contribution to the field of Named Entity Recognition, specifically tailored to the unique challenges and opportunities present in the restaurant and food industry. The scope for impact is broad, both in terms of advancing the state of the art in NLP and in providing practical tools for businesses in a highly competitive sector.

Chapter 2 : Background and Literature Review

This Chapter consists of several sections to extensively explain and discuss the needed details to fully understand the thesis. It first discusses the importance of text-mining customer reviews and the challenges faced by natural language processing (NLP) tasks in this domain. Then, the chapter briefly describes the different named entity recognition (NER) methodologies with some examples from the literature followed by a literature review of existing NER systems for food name detection. Next, a summary of the key findings of the literature review and the current research gaps and limitations are discussed. Moreover, descriptions of different deep learning layers that have been utilized to implement our NER systems are presented. Finally, the utilized language models in our research are reviewed before introducing the importance of character embeddings and the information in the part of speech (POS) features when utilized as input to an NER system.

2.1 Text Mining in Customer Reviews

Online customer reviews have greatly grown with the rise of e-commerce platforms, social media, and other digital channels. For businesses, text mining offers a deep dive into customer sentiment and feedback. Analyzing these reviews provides insights into product performance, service quality, and potential areas of improvement. In addition, by analyzing the textual data, companies can identify patterns and understand hidden trends that could be unseen in structured data, such as numerical ratings. Companies using text mining can quickly respond to negative feedback in a timely manner, adjust product features, and design targeted marketing strategies. This real-time market intelligence can significantly enhance customer

experience and service offerings, foster brand loyalty, and drive business growth (Ali et al., 2022).

On the other hand, and according to many studies including Sun et al. (2022), customers rely on reviews to help them in making purchase decisions. However, the great volume of reviews can be overwhelming. Text mining assists in summarizing vast quantities of reviews, presenting potential buyers with a more digestible overview of existing sentiments (Bai et al., 2023). Tools that employ text mining can help consumers quickly identify the most relevant reviews, or even provide summarized insights about the product or service in question. Examples of such tools are *ReviewMeta*, *Fakespot*, *SentiGeek*, etc.

Many text-mining applications are currently employed to extract valuable insights from customer reviews. Some notable ones include:

- **Sentiment Analysis:** This task identifies the polarity of the text, determining whether the sentiment is positive, negative, or neutral (Wang et al., 2021). Applied to customer reviews, it allows businesses to quickly gauge overall customer satisfaction or dissatisfaction and allows customers to make decisions based on others' opinions.
- **Topic Modeling:** This task helps in identifying the key topics available in a large volume of text. By analyzing customer reviews, businesses can determine the most frequently discussed aspects of a product or service (Kwon et al., 2020).
- **Aspect-Based Sentiment Analysis.** This task goes beyond general sentiment analysis by focusing on specific aspects or features of a product or a service

and determining the sentiment related to each aspect (Wang et al., 2021). For instance, in reviews for a hotel, ABSA can differentiate sentiments about the room, customer support, location, facilities, food/drink, and Cleanliness (Özen & Katlav, 2023).

- **Named Entity Recognition (NER):** NER identifies and classifies named entities in unstructured text into predefined categories, for instance, person names, names of organizations, locations, dates, and more. In the context of reviews, it can help businesses identify specific product names or brand mentions (Syed & Chung, 2021).
- **Text Summarization:** This task involves generating a concise and coherent summary of a lengthy piece of text. For customer reviews, it can provide potential buyers with an overview of prevailing sentiments without reading through hundreds of reviews (Amplayo et al., 2022).
- **Text Classification:** This task categorizes text into predefined labels or groups. For customer reviews, it can be used to automatically sort feedback into categories like service, product quality, delivery, etc. (Deniz et al., 2022).
- **Anomaly Detection:** Identifies patterns that do not match the expected behavior, helping in spotting reviews or feedback that deviates significantly from the norm (Chen et al., 2019).
- **Machine Translation:** Translates text from one language to another. It can be applied to translate reviews from various languages, enabling businesses to understand feedback from a global audience (Berard et al., 2019).

Despite the different valuable insights and the multiple advantages to businesses and individuals, text-mining customer reviews are also facing many challenges. The increasing number of digital platforms has resulted in a variety of customer reviews, each with its own format, language style, and rating system. For instance, reviews on professional websites may be very different from those on social media channels. This diversity necessitates the need to tailor the preprocessing methods for each platform (Jia, 2019). In addition, reviews may not be representative of the overall population, reflecting the viewpoints of a particular demographic more than others. This bias, unintentional, can mistakably reinforce societal stereotypes or give businesses an incorrect view of overall customer sentiment. Therefore, the drawn insights can be skewed and the necessity of developing techniques for addressing these biases is crucial. (Askalidis et al., 2017). Moreover, another complexity of customer reviews relies on the inclusion of colloquial slang, regional dialects, and misspelling errors. These elements pose another barrier for algorithms. As language remains fluid, with ever-evolving slang terms and shifting cultural nuances, the potential for misinterpretation grows. Consequently, it's important for both businesses and researchers to continuously refinement for their algorithms and adopts innovative methodologies to remain accurate in their analysis (Khurana et al., 2023).

In sum, while the promise of text mining customer reviews is profound, it comes with its unique set of modern challenges that businesses and researchers must navigate and work on addressing them.

Our research focuses on a distinct category of customer feedback: restaurant customer reviews. The following section highlights the significance of analyzing restaurant reviews and provides insight into the prominent customer review platform, Yelp.

2.1.1 Restaurant Customer Reviews

Restaurant customer reviews hold a distinct significance in the food industry, acting as powerful motivation for consumer behavior and business strategy. One primary reason is their influential role in decision-making. Before going out to dine or order, many customers consult online reviews. These testimonials, especially if they concern recent dining experiences or newly introduced dishes, can attract diners to go for the positively rated restaurant or the opposite (Jia, 2019).

The personal nature of dining further supports the relevance of reviews as every diner has different food experiences and individual tastes and preferences are varying. Thus, a dish that is liked by one diner might leave another diner with an unpleasant experience. By offering diverse perspectives and opinions, customer reviews enable potential diners to detect consistent feedback trends and make their decisions based on their personal tastes (Jia, 2019).

The granularity of feedback found in restaurant reviews also sets them apart. Diners frequently comment on different aspects of a restaurant, ranging from the ambiance and service to the cleanliness of the place, and even the presentation of specific dishes in addition to their taste. This level of detail proves invaluable to both potential customers who are seeking insight and restaurant managers aiming to enhance their service and offerings.

Moreover, in our digitally connected age, a few negative comments, especially if they gain traction on social media, can considerably spoil a restaurant's image. This underscores the importance for restaurant managers to monitor feedback closely, addressing any issue promptly and professionally.

According to Bai et al. (2023), the influence of reviews extends to digital visibility as well. On platforms such as Google and Yelp, positive reviews and high ratings can enhance a restaurant's importance in local search results. This digital visibility can be a significant factor in attracting customers, especially tourists or those unfamiliar with the locale.

Beyond the numerical-level ratings, the textual content of reviews is a goldmine of insights. While a star rating provides an immediate and generalized assessment of a restaurant, the narratives within reviews dive into the specifics of diner sentiments. This offers restaurateurs deeper insights into areas that require attention or dishes that have become customer favorites.

Lastly, as society evolves, so do dining trends and cultural preferences. An analysis of reviews over time can shed light on these shifts. For instance, an increasing mention of terms like "vegan" or "organic" might indicate a broader change in consumer demands (Bai et al., 2023).

2.1.1.1 Yelp Reviews. Yelp, founded in 2004, is a popular American platform specializing in hosting online business reviews. It proved its importance in influencing consumer perceptions and guiding their steps in making decisions. Given its extensive user base, Yelp offers a comprehensive dataset, providing a good understanding of localized sentiments and trends. The platform primarily facilitates users in discovering local businesses,

including a wide range of industries from dining and beauty services to health and entertainment.

Reviews on Yelp adopt a structured approach, utilizing a star-rating system that spans from 1 to 5 stars. Complementing this quantitative sentiment, users are encouraged to provide detailed textual reviews of their experiences, adding depth and context to the given ratings. This dual mechanism of feedback ensures a holistic representation of user experiences across Yelp's diverse business categories, which include, but are not limited to, dining, professional services, and retail.

Beyond its primary function as a review platform, Yelp can be characterized as a social networking site. Not only does it allow users to articulate their experiences, but it also empowers its community, colloquially known as "Yelpers" with social features. This encompasses the ability to forge connections by adding friends, following fellow reviewers, discovering events, and identifying reviews bearing similarities to their own contributions (Ariyasriwatana & Quiroga, 2016).

Reviews on Yelp, like other platforms, reflect the dynamic nature of language and evolving user sentiments. Users often incorporate current events, slang, local dialects, and pop culture references into their reviews. This makes mining and analyzing Yelp reviews both a challenge and an opportunity to tap into localized and current sentiments (Yu et al., 2017).

How does Yelp recommend reviews?

As explained by Yelp (2019), Yelp employs a private Recommendation System that screens reviews predicated on three key parameters: the intrinsic quality and utility of the review, its reliability, and the corresponding reviewer's activity level on the platform. This recommendation mechanism is wholly automated, perpetually active, and avoids any human intervention. Notably, both advertising and non-advertising establishments on Yelp undergo the same standard of review. If a review fails to meet the requisite standards and is consequently not recommended, it is excluded from the business's aggregate review list and is also exempt from influencing the overarching star rating. Nevertheless, such non-recommended reviews are retained and kept at the bottom section of the business's profile. Interestingly, these reviews are not permanently dismissed; Yelp's dynamic system may reevaluate and potentially reinstate previously unrecommended reviews, upon factors like a demonstration of the reviewer's credibility, given that the platform incessantly re-examines all reviews for compliance with its criteria.

Yelp has garnered considerable attention within the research community in recent years, primarily because it offers datasets at no cost. Scholars frequently utilize the Yelp Dataset for many research endeavors, including user behavior analysis, sentiment interpretation, development of recommendation systems, and exploration of trends in user reviews. (Yu et al., 2017).

To conclude this section, text mining of online customer reviews offers businesses an in-depth understanding of customer sentiment and feedback. Amplified by the rapid expansion of e-commerce and social media, these reviews afford enterprises valuable perspectives on

product effectiveness, the quality of services, and areas in need of improvement. Employing text mining can reveal patterns, trends, and sentiments. In addition to enabling businesses to promptly address negative feedback, this analytic approach also supports strategic marketing initiatives. Furthermore, consumers utilize these reviews to make their purchase decisions, but the sheer volume necessitates tools, like *ReviewMeta* and *Fakespot*, to distill the information.

Text mining applications range from sentiment analysis to anomaly detection, aiding businesses in extracting insights from vast and varied customer feedback. Despite its advantages, text mining also faces challenges such as diversity in review sources, potential biases, and language barriers that demand continuous refinement. Focusing on the realm of restaurant reviews, this feedback is paramount in influencing consumer behavior and guiding business strategies in the food industry. Yelp, as a predominant platform, plays a critical role in harnessing these reviews, with its Recommendation System filtering feedback based on quality, reliability, and reviewer activity, emphasizing the platform's significant contribution to research with its extensive datasets.

2.2 Named Entity Recognition Methodologies

Named Entity Recognition (NER) stands as a primary task within the umbrella of natural language processing (NLP). The process of NER can be divided into two distinct stages: the identification of the named mention followed by the categorization of the detected mention into its corresponding predefined class. NER plays an instrumental role in supporting various applications, including but not limited to, question-answering systems, text summarization, and machine translation. The specific entities identified often vary based on the domain in question. Commonly recognized entities span locations, organizations, and

individual names. In more specialized domains, such as the medical sector, entities encompass diseases, genes, and chemicals, etc. (Li et al., 2020).

Numerous studies have embarked on the development of NER systems tailored for diverse languages, domains, and applications.

The following sections provide brief overviews of the varied methodologies adopted for the implementation of named entity recognition systems across distinct domains.

2.2.1 Rule-Based NER Systems

The rule-based approach to NER necessitates the formulation of precisely crafted rules to both detect and categorize named entities. These rules predominantly lean on the syntactic and lexical patterns specific to each entity class and often leverage domain-specific dictionaries and gazetteers for support (Khan et al., 2020).

In the detection phase, a rule-based algorithm begins by seeking out potential entities. Upon locating one, it compares the entity against the established rules. Should the entity conform to these rules, it is subsequently classified under the appropriate category (Khan et al., 2020).

Numerous NER systems, such as those mentioned in Shaalan and Raza (2009) and Alfred et al. (2014), have been constructed on the foundation of these linguistic rules.

However, while certain rule-based NER systems exhibit commendable performance, this approach is not lacking limitations. Some notable constraints include (Cenikj et al., 2020), (Khan et al., 2020):

- The need for continuous adaptation to shifts in domain-specific changes, makes the periodic revision of syntactic rules imperative.
- The vital requirement for human expertise, both in language proficiency and rule generation.
- The non-transferability of rules designed for one language to another undermines the portability of such NER systems.
- The labor-intensive and the time consuming nature of rule development.

2.2.2 Feature-Based Machine Learning Methods

With the availability of extensive training corpora, supervised learning models have gained prominence by deriving rules from annotated datasets. According to Al-Smadi et al. (2020), the first machine learning method for named entity recognition (NER) integrated rule-based systems in a hybrid fashion. Examples of systems that adapted to this approach have been proposed by Petasis et al. (2001) and Shah and Bhadka (2017).

The development of robust NER systems is supported by well-established supervised learning algorithms, including the Hidden Markov Model (HMM), Maximum Entropy Model (MaxEnt), Decision Trees, Support Vector Machines (SVMs), Conditional Random Fields (CRF), and others.

In the context of NER, the HMM algorithm aims to distinguish the sequence of hidden states representing targeted named entities within a domain. This is achieved by pinpointing the state sequence that boasts the maximum likelihood based on transition probabilities. NER

systems architected using the HMM algorithm include Zhou and Su (2001), Zhou and Su (2005), and Morwal et al (2012).

Conversely, the Maximum Entropy Model (MaxEnt) prioritizes entropy maximization by identifying the optimal probability distribution. In the context of NER, the MaxEnt model gives labels to word sequences based on the most probable entity label, while considering neighboring words and their intrinsic features, such as linguistic attributes like Part of Speech tags (POS) (Al-Smadi et al., 2020). Noteworthy NER systems developed using the MaxEnt approach can be found in Chieu and Ng (2002), and Chieu and Ng (2003).

Decision trees for NER employ tree nodes and branches rooted in the linguistic characteristics of word sequences. The primary objective is to navigate towards the leaf node, which correlates with a specific named entity label or class. Evaluations of the C4.5 decision tree algorithm for NER applications are detailed in Paliouras et al. (2000) and Szarvas et al. (2006).

Support Vector Machine (SVM) has been utilized in the implementation of various NER systems. SVMs are recognized for their generalization capabilities when used as classifiers. NER systems designed using the SVM algorithm are highlighted in references Ekbal and Bandyopadhyay (2010), Yusup et al. (2019), and Gridach and Haddad (2018).

Additionally, Conditional Random Fields (CRF) gained attention in NER after the study presented by Lafferty et al. (2001). This paper introduced CRF as a novel probabilistic framework for segmenting and labeling sequences, providing an in-depth mathematical description of CRF. It also details the parameter estimation using the maximum likelihood principle. Notably, Lafferty et al. (2001) elaborated on how CRFs address the label bias issue,

which is a shortcoming in Maximum Entropy Markov Models (MEMMs). Unlike MEMMs, CRFs provide a globally normalized probability distribution for tag sequences, given an observation sequence. When tested on text, CRFs showcased results that were either superior or competitive compared to Hidden Markov Models (HMMs) and MEMMs.

Encouraged by the results in Lafferty et al. (2001), researchers began exploring the utility of CRF for various natural language processing tasks, including NER. This exploration is evident in studies such as Settles (2004), as Patil et al. (2020), and Khan et al. (2022). As with other supervised learning algorithms, CRFs are trained on a labeled corpus. In post-training, the model assigns a conditional probability to tag or label sequences based on observed input features, with the goal of maximizing the probability of certain label sequences. This process is termed "inference" and CRFs can utilize various inference algorithms, like the Viterbi algorithm or belief propagation.

Overall, a key benefit of supervised machine learning NER systems is their adaptability to new domains with only minor modifications, provided they're trained on a domain-specific corpus. However, a challenge arises when extracting features; feature engineering is dependent on large, annotated datasets to ensure quality and such datasets might not always be accessible, especially for certain domains and languages (Sun et al., 2022).

2.2.3 Unsupervised Machine Learning-Based Methods

The unsupervised learning methods for named entity recognition (NER) are not as widespread as their supervised counterparts. At the heart of unsupervised methods lies clustering. The fundamental principle underpinning clustering-based NER systems is the

extraction of entities predicated upon contextual similarities within the resultant clustering groups (Nadeau & Sekine, 2007). Notable unsupervised learning NER systems have been proposed in references Etzioni et al. (2005), Nadeau et al. (2006), and Zhang and Elhadad (2013).

2.2.4 Deep Learning Based Methods

In recent years, the natural language processing (NLP) research community has witnessed the remarkable efficacy of deep learning models in the development of named entity recognition (NER) systems, thereby garnering substantial attention and scholarly interest. At the heart of these advanced techniques is their use of word embeddings. These specific embeddings are integrated into the neural model, effectively capturing the representation of each word within the text. Once this is accomplished, the model then goes deeper to systematically extracting the underlying semantic information of the text (Sun et al., 2022).

The journey of NER systems rooted in deep learning can be traced to the groundbreaking research showcased by Collobert et al. (2011). Within this pioneering work, an all-encompassing neural network model was proposed and implemented to address a range of NLP tasks, NER being a notable inclusion. The vital component of this model was the convolutional neural network (CNN), which, as per the insights of the authors, holds distinct efficacy in NLP undertakings. This efficacy is particularly evident when the task revolves around distilling local features embedded within a sequence. The innovative research encapsulated in Collobert et al. (2011) stands out as a pioneer, highlighting the importance of word embeddings as a fundamental element in NLP models. With an aim to capture the semantic information hidden within unlabeled datasets, the CNN model was designed. After

this initial phase, the model was then fine-tuned to cater to specialized NLP tasks, encompassing Part of Speech (POS) tagging, NER, and chunking. Nonetheless its commendable performance, the model had its limitations. Its use of the fixed window size mechanism, when attempting to comprehend contextual relationships within word sequences, constrained it to grasp only the immediate local context, leaving the broader, long-distance dependencies largely unaddressed (Wang & Su, 2022).

Deep learning layers, particularly recurrent neural network layers (RNNs), have become increasingly popular in the Named Entity Recognition (NER) framework. Variants of RNNs, such as the Bidirectional Long Short-Term Memory (BiLSTM) and Bidirectional Gated Recurrent Unit (BiGRU), are now commonly integrated into many recent NER systems to help with feature extraction. The use of these models has consistently shown strong performance across various languages and applications (Goyal et al., 2021). Both BiLSTM and BiGRU are effective in drawing out hidden representations by grasping contextual semantic features, which boost the process of identifying named entities (Xu et al., 2019). For these NER models frameworks to be completed, the Conditional Random Fields (CRF) is incorporated as the final layer. This CRF layer takes charge of assigning labels to each word in the sequence based on the highest global probability linked to that word (Sun et al., 2022). Notably, the BiLSTM-CRF combination is recognized as a leading deep learning model frequently adopted in the development of NER systems (Li et al., 2020).

The work presented by Huang et al. (2015) is highlighted as the first to demonstrate the advantages of using the bidirectional LSTM combined with CRF for tagging sequences in natural language processing tasks, including the named entity recognition task. When tested on

the CoNLL-2003 dataset, the performance of the model introduced by Huang et al. (2015) surpassed other models available at that time. The use of the BiLSTM model in their approach ensures that both past and future details from the sentence sequence are captured. Adding the CRF on top of the BiLSTM optimizes the output sequences, considering the relations between neighboring labels in the sequence.

On the other hand, an earlier approach using a different kind of recurrent neural network (RNN) was introduced by Yang et al. (2016). They proposed the BiGRU-CRF model for sequence labeling tasks, and this model also showed strong results in several tasks, like POS tagging, named entity recognition, and chunking across multiple languages. The model didn't just use GRUs to capture word-level morphological and contextual details, but it also utilized the BiGRU to produce character embeddings (Yang et al., 2016). Their model achieved an impressive F1-score of 91.20% on the CoNLL-2003 English dataset for the named entity task.

In another study by Lample et al. (2016), the BiLSTM-CRF model was utilized for both word and character embeddings. This approach used the BiLSTM model specifically to generate character embeddings, capturing the morphological nuances of words. This research further solidified the effectiveness of the BiLSTM-CRF model, producing top-tier performance results. Additionally, it highlighted the advantages of using character embeddings in building NER models. This is particularly beneficial for languages rich in morphological variations and for representing rare out-of-vocabulary words (OOVs).

The emergence of the transformer architecture, as presented by Vaswani et al. (2017), marked a significant change in the way the NER task was approached. As detailed by Vaswani

et al. (2017), this architecture employs a self-attention mechanism to process data, which is unlike the sequential processing used in recurrent neural networks. This change allows for more efficient management of long-distance dependencies in text sequences. Among the transformer models, the Bidirectional Encoder Representations from Transformers, or BERT was the first to achieve state-of-the-art results across various NLP tasks, including named entity recognition (Devlin et al., 2019).

2.3 Named Entity Recognition in Food Domain

This section offers a thorough literature review encompassing two specialized but interconnected domains: the identification of food names and the recognition of dish names, both achieved through Named Entity Recognition (NER) methodologies. Within the larger context of natural language processing, these topics have seen significant development, but they each bring their own unique challenges and objectives. The primary goals of this literature review are: first, to trace the advancements and innovations that have marked these specific domains, and second, to identify existing gaps and limitations that need further exploration.

2.3.1 Named Entity Recognition to Detect Food Mentions

This section is organized into three parts, each focusing on a different method for creating Named Entity Recognition (NER) systems aimed at identifying references to food. This approach is beneficial because the evolution of NER systems in the food area mirrors the progression seen in other fields. It starts with rule-based methods, then advances to deep learning techniques, and finally to ensemble models.

2.3.1.1 Rule-Based NER Systems. One of the first papers in the NER domain for detecting food names is the presented work by Eftimov et.al (2017) which proposes a rule-based named entity recognition (NER) method called DrNER for extracting dietary recommendations from scientific literature. DrNER combines two approaches: terminological-driven NER and rule-based NER. Its objective is to extract three main entities from evidence-based dietary recommendation text, namely food, nutrient, and quantity/unit. To construct the terminological part of the system, a dictionary is created for each entity type. The dictionary for the food entity, which is the primary focus of our research, is developed based on the University Centre for Computer Corpus Research on Language (UCREL) Semantic Analysis System (USAS) online English semantic tagger. The food entities are categorized into two groups: food and farming, and life and living things.

The DrNER system consists of two stages:

- Stage 1: Entity mention identification involves pre-processing the text and applying various NLP tasks such as sentence segmentation, chunking, word-level tokenization, and POS tagging. The chunk tokens are tagged in the BIO (Beginning, Inside and Outside) format. However, in some cases, an entity mention may consist of more than one noun phrase. To enhance entity detection, three ad-hoc chunking rules are applied using predefined Boolean rules. These rules are based on the linguistic features of dietary recommendations.
- Stage 2: Entity mention classification focuses on dietary information and utilizes text syntactic analysis. Also, the rules used for classification are based on the semantic meaning of the entity mentions.

To evaluate DrNER, a test corpus was manually created comprising 100 documents. Fifty documents were extracted from websites that were scientifically validated, while the remaining fifty were obtained from scientific publications. The corpus was manually annotated by domain experts. For the evaluation of food entities, the results are as follows: the number of true positives is 539, the number of false positives is 5, and the number of false negatives is 25.

The key findings of the paper Eftimov et al. (2017) demonstrate that rule-based NER methods can effectively extract dietary recommendations from the scientific literature. Furthermore, to enhance performance, it is highly recommended to employ a set of rules based on the linguistic and semantic features of dietary recommendations. However, one significant limitation of this research is that the drNER methodology is restricted to extracting dietary recommendations solely from scientific literature and cannot handle dietary recommendations found in other types of text, such as news articles or social media posts.

On the other hand, another surveyed paper introduces FoodIE, another rule-based named entity recognition system that focuses on extracting food entities only (Popovski et al., 2019). The paper presents a rule-based NER method specifically designed for extracting food entities from unstructured text. FoodIE employs a set of rules based on the linguistic and semantic information of food entities. Similar to other named entity recognition systems, text-processing steps are necessary for the successful detection of the entities.

FoodIE comprises the following phases:

1. **Recipe text pre-processing:** This phase involves cleaning the text by removing quotation marks, consolidating white spaces, performing transliteration, and converting fractional values into real numbers.

2. **POS tagging:** Two methods, the USAS semantic tagger and coreNLP library, are used for POS tagging. However, according to Popovski et al. (2019), neither method achieves perfect tagging. Therefore, a post-processing step is implemented to refine the tag set using linguistic rules.

3. **Semantic tagging of food names:** This phase forms the core of the FoodIE system and is based on four simple linguistic rules derived from the food entities extracted from the "food and farming" category of the UCREL Semantic Analysis System (USAS) framework.

4. **Food-named entity recognition:** In this phase, words tagged as food tokens in the previous step are considered, and another set of linguistic rules is applied to identify food entities that span multiple words (chunks).

The performance of the proposed method is evaluated on a dataset of food-related text. Two datasets were created from different sources of food recipes: the first consists of 200 recipes extracted from the *Allrecipes*¹ website, and the second includes 1000 recipes extracted from the *MyRecipes*² website. The system's evaluation was manually conducted by two domain experts. The first expert manually annotated the datasets with the food chunks, while

¹ <https://www.allrecipes.com/>

² <https://www.myrecipes.com/>

the second expert cross-referenced the manually obtained chunks with those extracted using the FoodIE system. The evaluation of FoodIE on both datasets consistently yielded close values for evaluation metrics: 97% precision, 94% recall, and 96% F1 measure.

Furthermore, to assess the robustness of FoodIE, a comparison between its performance and the drNER (Eftimov et al., 2017) system was conducted by comparing their accuracies in detecting food entities from the same ten sentences. FoodIE demonstrated more precise results, suggesting that it can be utilized as a post-processing tool to enhance the final performance of drNER in detecting food entities. According to Popovski et al. (2019), the performance of FoodIE is dependent on the quality of both POS and semantic taggers. Therefore, improving the taggers' quality is recommended to enhance the system's performance.

The key findings of the paper confirm that rule-based NER methods can effectively extract food entities from unstructured text. Moreover, the performance of rule-based NER methods can be enhanced by employing a set of rules based on the linguistic and semantic features of food entities. However, the method has limitations in extracting food entities expressed in specific ways according to the provided rules. For instance, it cannot extract food entities expressed as questions or negative statements. Additionally, the method is restricted to extracting food entities within a specific domain, such as recipes or food blogs, and cannot handle food entities found in other types of text, such as news articles or social media posts.

On the other hand, the lack of annotated corpora specifically tailored for food-named entity recognition tasks is the primary motivation behind the creation of the *FoodBase* corpus (Popovski et al., 2019). The *FoodBase* corpus serves as a benchmark dataset for various

machine learning tasks and is expected to contribute to the development of food embedding spaces required by predictive analysis models.

In contrast, an examination of existing annotated corpora prior to the *FoodBase* corpus reveals several limitations. For instance, the r-FG corpus is constructed based on only 266 Japanese recipes, while the CURD corpus relies on 300 annotated recipes and 350 unannotated ones, utilizing an undetailed annotation scheme that provides only general food entities.

To construct the *FoodBase* annotated corpus, 1000 recipes were selected from the popular website *Allrecipes*. The recipes were chosen from five categories on the website: 'Appetizers/Snacks', 'Breakfast/Lunch', 'Dessert', 'Dinner', and 'Drinks'. The creation of the corpus was followed by the annotation step, which employed the Hansard food semantic tags. The primary Hansard semantic tag category used was 'Food and Drink' consisting of three subcategories: 'Food', 'Production of food-Farming', and 'Acquisition of Animals for Food-Hunting'. Additionally, two other categories, 'Animals' and 'Plants' were employed. The first three semantic tag categories and their subcategories resulted in around 219 tags utilized for annotating the *FoodBase* corpus.

To proceed with annotation, the FoodIE NER system (Popovski et al., 2019) was employed to locate food entities. An additional step was added to the end of the FoodIE pipeline to annotate each extracted food entity using FoodIE with the Hansard corpus semantic tags. This annotation process followed the scheme where if the token did not belong to the 'Food and Drink' category, the 'Animals' or 'Plants' categories were used for annotation. If the entity did not fit into the other two categories, it was tagged as 'Food', a subcategory of the top-level hierarchy of the 'Food and Drinks' category.

A manual evaluation was conducted to assess and fix the semantic annotations. During this phase, missing semantic tags were added to food entities, while incorrect tags were removed. Consequently, a gold standard corpus consisting of 1000 recipes was created, with 12,844 annotated food entities, including 2,105 unique ones. Additionally, based on the performance of the extended FoodIE in annotating food entities, a much larger version of the *FoodBase* corpus was created, encompassing 21,790 recipes extracted from the same five categories as the gold standard. This larger version annotated 274,053 food entities, with 13,079 unique ones. However, the larger version of the *FoodBase* corpus was not manually checked and evaluated like its 1000-recipe counterpart. The researchers of Popovski et al. (2019) conducted various statistical comparisons between the two versions of the *FoodBase* corpus: the curated gold standard and the un-curated version. The analysis revealed that different distributions, such as words per recipe, food entities per recipe, and semantic tags assigned per food entity, exhibited similar trends in both corpora. However, due to the inclusion of a large number of recipes, the distributions in the un-curated version appeared much smoother.

It is worth noting that the *FoodBase* corpus adopts the BiC Format as its chosen annotation format. This XML-based format was selected to facilitate the sharing of text data along with the annotations.

The key findings of the paper Popovski et al. (2019) highlight the value of the *FoodBase* corpus as a valuable resource for advancing research on food information extraction and food-related natural language processing tasks. However, a limitation of the corpus is that

it is exclusively annotated for food entities in recipes. Future research could expand the corpus to include other types of food-related text.

Furthermore, it is noteworthy to highlight the results of a comprehensive survey conducted on named entity recognition (NER) systems for food entities, as detailed in Popovski et al. (2020). It is important to note that this survey focused on the literature up until 2020, comparing the performance of four NER methods in the food domain. The first method, FoodIE (Popovski et al., 2019), has been previously analyzed and discussed in this section of the Thesis. The second method is the NCBO (National Center for Biomedical Ontology) annotator, which is part of the BioPortal software. The NCBO annotator serves as a web service for extracting and annotating biomedical ontology concepts in unstructured text provided by the user (Jonquet et al., 2009). It is used to automatically annotate datasets based on the selected ontology.

The researchers examined three food-related ontologies using the NCBO NER: FoodOn (Griffiths et al., 2016), OntoFood (Boulos et al., 2015), and SNOMED CT (Donnelly 2006). Each run of the NCBO annotator on one of these ontologies is treated as a distinct NER system. It is worth mentioning that Popovski et al. (2020) emphasizes the limited availability of food ontologies, and it is not surprising to find variations in the recognition of food mentions among different ontologies.

To evaluate the four NER systems, their outputs were compared to the gold standard annotated food corpus, *FoodBase* (Popovski et al., 2019). Given that the comparison focuses on NER systems, the evaluation metrics employed are true positive (TP), false positive (FP), and false negative (FN). Since the interest lies in a single class (food entity or not), true

negatives (TN) are not considered part of the evaluation metrics. Additionally, an additional metric called "Partial" is used to track incomplete extracted food entities, in cases where the NER system fails to recognize all tokens (words) that belong to a food entity.

The analysis reveals that the FoodIE system outperforms the other three NER systems in all evaluation metrics. Specifically, FoodIE exhibits the highest number of true positives and the lowest number of false positives and false negatives. This can be attributed to the fact that the domain coverage of the three ontologies indicates that numerous entities in the recipes are considered food entities, but they are absent from the ontologies. Furthermore, FoodIE's utilization of semantic information makes it more representative of the food domain. Therefore, the evaluation metric values for FoodIE are as follows: 96.05% (F1 score), 97.80% (precision), and 94.37% (recall). Moreover, the evaluation process demonstrates that the utilization of the OntoFood ontology with the NCBO annotator yielded the poorest results, confirming that this particular ontology inadequately represents the food domain. The evaluation metrics for NCBO using OntoFood are as follows: 32.62% (F1 score), 85.48% (precision), and 20.16% (recall).

2.3.1.2 Deep Learning NER Systems. The first research effort to apply deep learning methodologies for food-named entity recognition was presented by Cenikj et al. (2020). According to Cenikj et al. (2020), their proposed NER system is the first to leverage the FoodBase corpus Popovski et al. (2019) in exploring machine learning methods for food entity detection. The survey conducted by Cenikj et al. (2020) on various NER systems reveals that neural network architectures offer the most promising results. Consequently, the BuTTER system is developed as a food NER model based on the Bidirectional Long Short-Term Memory (BiLSTM) and Conditional Random Fields (CRF) layers.

To evaluate the performance of the proposed system, three pre-trained word representation models: Word2Vec, GloVe, and FastText are utilized, alongside a custom word embedding model known as lexical embeddings. The researchers experimented with four different pre-trained word embedding representations, including variations with and without handling Out of Vocabulary (OOV) words. For the pre-trained models, OOV words are processed by converting Arabic numerals into a textual format, splitting OOV words based on contained punctuation, and generating word representations by averaging the embeddings of the substrings. Zero vectors are used when the substrings are not present in the vocabulary list. As for the custom-generated lexical embeddings, experiments were conducted with and without lemmatization.

Based on the described word representation models, a total of eight representation models for data are obtained:

- (1) Three models of pre-trained word embeddings without handling OOV words.

- (2) Three models of pre-trained word embeddings with handling OOV words.
- (3) One custom-trained word embedding without lemmatization.
- (4) One custom-trained word embedding with lemmatization.

It is important to note that for the Glove embedding model, five variations were experimented with, and the study records the average results obtained from these variations. The five variations according to RaRe-Technologies (2018) are: *glove-twitter-25*, *glove-twitter-100*, *glove-twitter-200* which were trained on Twitter tweets and have dimensions of 25, 100 and 200 respectively, and *glove-wiki-gigaword-50* and *glove-wiki-gigaword-300* which were trained on Wikipedia and Gigaword datasets with 50 and 300 dimensions respectively.

To compare the different word embedding models, all three pre-trained models and the custom-trained word embeddings are of dimension 300. As for the neural network architecture of the proposed NER system, BuTTER, the BiLSTM, and CRF layers are expected to yield better results in extracting food entities. The design consists of five layers: input and embedding layers to incorporate word embeddings, a BiLSTM layer, a fully connected time-distributed layer, and a CRF layer. The outputs of the CRF layer provide probability values for words being outside, inside, or at the beginning of a food entity. Consequently, the NER task can be treated as a classification problem with three classes: B-Food, I-FOOD, and O.

The BiLSTM-CRF model is not the only experimental model evaluated in the research study. Another model, known as the Char-BiLSTM-CRF model, is assessed by incorporating character embeddings along with the word embeddings generated by each word representation model. The Char-BiLSTM-CRF model includes additional layers such as input and embedding

layers to create character embeddings and a time-distributed LSTM layer to process character embeddings before concatenating the results with word embeddings. This comprehensive representation of tokens at both word and character levels is then fed into the main BiLSTM and CRF layers.

The evaluation of the BuTTER system is conducted on the *FoodBase* corpus using stratified 5-fold cross-validation. The choice of 5 folds aligns with the five classes of different recipes in the *FoodBase* corpus. The macro-averaged F1 score is used as the evaluation metric for each of the five folds.

The results demonstrate that BuTTER achieves an accuracy of 94.6%, outperforming other reported NER methods in the literature. This result holds true regardless of whether character embeddings are utilized or not. The lexical model achieves the highest F1 macro score and requires the fewest epochs to converge. Specifically, the lexical model reports an F1 score of 0.94, while the BiLSTM-CRF architecture reports F1 scores of 0.91, 0.90, and 0.90 for GloVe, Word2Vec, and FastText, respectively. As for the Char-BiLSTM-CRF model, the average F1 scores are 0.95, 0.92, 0.91, and 0.92 for the lexical, GloVe, Word2Vec, and FastText models, respectively. Among the three pre-trained models, GloVe achieves the best F1 macro scores for both model architectures, scoring 0.91 (BiLSTM-CRF) and 0.92 (Char-BiLSTM-CRF). Additionally, Word2Vec exhibits the fastest convergence, while FastText requires a larger number of epochs. Furthermore, a slight increase in F1 macro scores is observed when using the proposed handling of OOV mechanism with the three pre-trained word embedding models. Similarly, the use of lemmatization with the lexical model yields a slight increase in performance compared to the non-lemmatized approach.

On the other hand, one limitation of BuTTER is that it is trained solely on a recipe dataset for detecting food names. Future research could expand BuTTER to be trained on more diverse datasets encompassing text from news articles, customer reviews, social media posts, etc. It is important to mention that our research builds upon the baseline experiments conducted with the BuTTER system.

Another limitation of BuTTER is its inability to identify the semantic types of food entities. For example, it is trained to detect food mentions without classifying or identifying the semantic types of the entities, such as ingredients, dishes, or cuisine. In Chapter 4, an in-depth examination of the experiments conducted with the BuTTER system is presented. This section provides a comprehensive analysis of the system's performance, including the evaluation of different word representation models and the impact of incorporating character embeddings along with the results of our replication for these experiments.

The authors of the research study Diwan et al. (2020) put forward an innovative technique for representing and modeling the ingredients and instructions within cooking recipes, based on named entities. This research was prompted by the need for a system that can encapsulate both semantic and syntactic attributes of recipes. Existing methods, as per Diwan et al. (2020), lack efficacy or adaptability when applied to a range of recipes. By applying a named entity-based approach, the authors tackle these shortcomings and offer a robust representation of recipes.

In handling the ingredients portion, the authors perceive it as a catalog of ingredients along with associated characteristics such as quantity, temperature, and processing status. The

primary goal of the research is to devise a strategy capable of transforming these features into graphical illustrations for the recipes.

The principal methodology employed involves Named Entity Recognition (NER), used by the authors to pinpoint crucial components of an ingredient section. These components are seen as the tags for detection by the NER system, and they include Ingredient Name, Unit, Quantity, Processing State, Size, Dryness/Freshness state, and Temperature. The selection of these attributes followed a thorough manual examination of descriptions within the USDA's Standard Legacy Database, a highly esteemed resource for nutritional data.

To carry out the named entity recognition assignment, the authors utilize the Stanford NER Tagger (Finkel et al., 2005). To optimize the NER model's effectiveness, a two-stage process was applied. First, pre-processing was conducted on the ingredient phrases to eliminate stop words, lemmatize all words, and convert the text to lowercase. This was aimed at ensuring accurate classification of similar ingredient terms which might vary due to plurality, capitalization, or hyphenation. Next, an inventive strategy was devised to assign the Parts of Speech (POS) tags of the phrases. The Stanford POS Twitter model was employed given that ingredient listings often lack a comprehensive grammatical structure and bear resemblance to tweets in their lexical structures. The output was then vectorized, depending on the frequency of each POS tag.

The authors then organized clusters within this vector space to group ingredient phrases with analogous lexical structures using the K-means algorithm. From these clusters, phrase subsets were selected to form a training set for the NER tagger. This technique was

perceived as generating a representative training set derived from various data sources and cuisines.

A total of 118,000 recipes were compiled from two major sources: *AllRecipes* and *Food.com*³. The NER models were trained and tested on three derived datasets namely, Recipes from *AllRecipes*, Recipes from *Food.com*, and the Recipes merged from both sources.

The findings revealed that the model trained on the composite dataset yielded the best results, with F1 scores of 0.97, 0.95, and 0.96 when evaluated on the *AllRecipes* dataset, *Food.com* dataset, and the combined dataset respectively. From the 118,000 recipe dataset, the model was able to extract 20,280 unique ingredient names, demonstrating its extensive applicability potential in computational food science and related fields.

One of the drawbacks of this methodology is its inability to create graphical representations of recipes presented in diverse formats. Future studies could expand the method to accommodate recipes written in various styles. Another identified limitation is that the presented F1 scores for the NER system do not specify the performance of detecting each of the seven tags (ingredient attributes). This omits a detailed insight into the system's ability to detect each class individually.

However, the authors' method is pioneering as it treats ingredients and instructions as separate sections, each with unique attributes and characteristics. This perspective could be influential in the ongoing progress of recipe data modeling and semantic analysis. Furthermore, the study suggests that this technique could enhance the performance of other

³ <https://www.food.com/>

recipe-related natural language processing (NLP) tasks such as translating recipes into different languages.

The research paper Stojanov et al. (2021) investigates a series of Named Entity Recognition (NER) models collectively referred to as the FoodNER system. This system relies on contextual BERT embeddings to facilitate food-named entity recognition. Three BERT language models, namely BERT-Standard, BioBERT-Standard, and BioBERT-Large, have been fine-tuned for the NER task, targeting the detection of five distinct semantic resources. The purpose of these fine-tuned models is to execute the detection task across these semantic resources. Table 2-1 encapsulates the summary of the best results for NER task.

Table 2-1

Results Summary of the NER Tasks

Semantic Resource	Best Macro Average F1-score
Food verses non-food	94.31(BERT)
Closest Hansard food semantic tags	78.96 (BioBERT Large)
Parent Hansard food semantic tags	76.26 (BioBERT Large)
FoodOn semantic tags	78.13 (BERT)
Systematized Nomenclature of Medicine Clinical Terms food semantic tags (SNOMED CT)	76.01 (BERT)

Note. The table demonstrates the best recorded macro average scores of several NER models in detecting food mentions from different semantic resources. Adapted from “A fine-tuned bidirectional encoder representation from transformers model for food named-entity recognition: Algorithm development and validation”, by R. Stojanov, G. Popovski, G. Cenikj,

B. K. Seljak, & T. Eftimov 2021, *Journal of Medical Internet Research*, 23(8), e28229.

<https://www.jmir.org/2021/8/e28229>. Copyright 2021 by the authors.

The aforementioned language models have been fine-tuned utilizing the gold standard food-annotated corpus, *FoodBase* (Popovski et al., 2019). The performance of the system was evaluated against the BuTTER baseline.

The primary insights derived from this study include:

- The food classification model, tasked with distinguishing between food and non-food items, surpassed the performance of the rule-based FoodIE system (Popovski et al., 2019) with F1-score of 94.31, achieving results comparable to those of the BuTTER NER system (Cenikj et al., 2020) where the best scored F1 score of 94.64 is achieved by BuTTER when using a BiLSTM-CRF model utilizing the lexical embedding model as the input to the model.
- It is noteworthy that these diverse models can be evaluated online via the FoodViz⁴ (Stojanov et al., 2020) platform. This website enables users to choose any NER model and supply their own text to be automatically annotated by the selected model.
- The findings provide support for the effectiveness of fine-tuned BERT models in detecting food names. Nevertheless, future research should explore the

⁴ <http://foodviz.env4health.finki.ukim.mk/#/recipes>

performance of these models on alternative corpora and the incorporation of additional features into these models.

An important paper addressing the detection of food entities in the unstructured text of social media is Lee et al. (2021). Central to this paper is the mission of augmenting public awareness surrounding food wastage, which mainly stems from misunderstandings related to food expiry date interpretations. The study attempts to extract mentions of food from tweets posted by users on Twitter, subsequently mapping these mentions to corresponding entities within the FoodKeeper dataset (Qamar, 2018). Comprising 509 distinct food entities, the FoodKeeper dataset is enriched with supplementary details associated with each entity, encompassing cooking methodologies, storage techniques, and other pertinent information. Consequently, once the NER model identifies and associates a tweet with a specific food entity from the dataset, a responsive tweet is automatically generated, spreading relevant information about the said food item, and emphasizing sustainable consumption practices.

To construct the named entity recognition (NER) system, a training corpus was formulated leveraging the 509 unique food names from the FoodKeeper dataset. These entities were utilized as search parameters for the Twitter API, leading to the aggregation of 40,000 tweets that housed mentions of these food entities, which then constituted the training corpus.

Subsequent to this, a series of preprocessing measures were executed on the dataset. This encompassed tokenization, part-of-speech tagging, and sentence segmentation facilitated through the employment of a dependency parser. The entirety of these preprocessing operations was conducted using the SpaCyV2 open-source library, which fundamentally operates on a convolutional neural network. In contrast, the more recent SpaCyV3 model for

NER is grounded in a transformer-based architecture. Additionally, word embeddings integral to the training process were generated through tok2Vec, setting the stage for training the named entity classifier on the annotated corpus. As a result of these rigorous processes, two distinct NER classifiers were trained to identify food mentions.

Upon the identification of food entities in tweets by the NER models, corresponding details for these entities were sourced from the *FoodKeeper* dataset. This enabled the system to promptly send out tweets containing the relevant information, amplifying the intended awareness among Twitter users.

Both NER models underwent an evaluation phase on a set of randomly collated tweets, which were subjected to manual annotation. The efficacy of the NER model can be encapsulated by three key metrics: with precision recorded at 0.96, recall at 0.52, and an F1 score of 0.68.

Building upon the foundational work presented in Lee et al. (2021), the research outlined in Chenze et al. (2022) primarily seeks to enhance the suboptimal recall rate observed in the NER system. To address this, a novel methodology is proposed specifically for the recognition of food-related named entities within tweets. A main component of this innovative approach is the incorporation of the snowball technique, employed specifically for the detection of food mentions within the provided tweets.

The training corpus initially prepared by Lee et al. (2021) serves as the data foundation for training the SpaCyV3 named entity recognition model in this extended research. It's essential to note that this particular NER model, being underpinned by a transformer

architecture, distinguishes food mentions using a binary classification system: categorizing mentions as either 'food' or 'not food'.

The snowball technique is applied iteratively. In the initial iteration, before the NER model is trained, a subset of the training corpus is chosen. At the same time, a list of food-related keywords, derived from the FoodKeeper dataset to represent diverse food mentions, is checked. These keywords are then matched against the selected portion of the corpus. Based on the frequency of these keywords, tweets are ranked, with those containing the most prevalent keywords ascending to the top. Subsequently, the top ten tweets from this ranked list are exploited to train the NER model.

Following training, the model's efficacy is assessed on a validation dataset to find out the improvements in the primary metrics, specifically precision, and recall. Concluding this training iteration, the preliminary model is established. The procedure then continues by employing another subset of the remaining training corpus. Within this subset, the frequency of keywords associated with food mentions is matched. The tweets are then ranked accordingly, and the top ten tweets from this ranking are employed to further refine the NER model. This iterative process persists until no further keywords are detected, resulting in the completion and readiness of the final trained model.

The enhancement derived from the snowball method was evaluated under various hyperparameter settings, notably the number of initial keywords employed at the onset of the process. The findings reveal a notable increase in the NER model's performance, with metrics reaching a value of 0.80 for precision, recall, and F1 score. Notably, the deployment of the

snowball technique has led to a significant boost in the recall metric, elevating it from its previous 0.52 to an impressive 0.80.

In the context of preparing annotated datasets for food mentions detection, Wróblewska et al. (2022) revealed a comprehensive dataset named TASTEset, designed specifically for recognizing food-related entities within recipes. This dataset is a collation of 700 unique recipes, containing 3,788 ingredients and over 13,000 diverse entities. These entities have been gathered from four distinctive culinary platforms: AllRecipes, Food.com, Tasty.co⁵, and Yummly.com⁶. TASTEset covers a range of nine entity categories such as ingredient names, quantities, units of measurement, culinary methods, physical attributes of ingredients, taste aspects, intended use, and ingredient parts. The primary objective of this dataset is to function as a reference point for Named Entity Recognition (NER) models that focus on extracting food-related information from recipes.

The creation of TASTEset was driven by the evident need for a robust and high-quality dataset tailored for food entity recognition within recipes. The challenges in this field are multiple and complex; food entities mentioned in recipes often suffer from ambiguity and the terminology used to describe these entities can significantly vary. According to Wróblewska et al. (2022) prior to TASTEset, existing datasets tasked with this specific problem were mostly small-scale or inadequately annotated. TASTEset responds to these deficiencies by delivering a dataset of significant scale and superior quality, tailored specifically for recognizing food-related entities in recipes.

⁵ <https://tasty.co/>

⁶ <https://www.yummly.com/>

TASTEset, a curated dataset, benefits from manual annotation carried out using the BRAT annotation tool (Stenetorp et al., 2012). It encompasses an array of entity types with attribute values. The spectrum of entities extends from straightforward ingredients such as "bread", to more complex ones like "2 lbs sweet potatoes, peeled and cut into 1/8 inch thick slices".

In their paper, the authors also assess several cutting-edge Named Entity Recognition (NER) models on the TASTEset dataset, incorporating architectures like BERT and LUKE. Across most categories, BERT-based models showcased superior performance. However, when identifying the FOOD entity, LUKE demonstrated a slight edge with an average F1-score of 0.915. It's also significant to note that the LUKE architecture is fine-tuned on the RoBERTa-base transformer model.

The results indicate that the deep learning model founded on the BERT architecture outperforms others, closely followed by the BERT-CRF model. The authors conclude that TASTEset proves to be an invaluable asset for advancing and evaluating models that recognize food-related entities. Furthermore, they highlight that the benchmarks incorporated in TASTEset can be employed to contrast the performance of various models and recognize the most effective strategies for food entity recognition in recipes. However, further research is recommended to measure these models' performance on different corpora and to integrate more features into the models for enhanced performance.

In another scholarly endeavor, the paper Fan and Hunter (2023) employs BERT embeddings in conjunction with the BiLSTM-CRF architecture to detect not only food names but also other pertinent entities within food recipes. The core objective of this research is to

advocate for a framework capable of formulating flow graphs that coherently outline the step-by-step processes basic to cooking recipes. Behind this framework are two fundamental components: the named entity recognition system, which is tasked with detecting various entities from a recipe, and the relation extraction module, which functions to interrelate the identified entities cohesively.

As per the assertions made in Fan and Hunter (2023), the presented research is pioneering in its proposition of a framework that can accurately craft a comprehensive flow graph interpreting English culinary recipes. Concerning the NER component, a profound deep learning model, the BiLSTM-CRF, is thoroughly trained on the 300-r recipe corpus. This training aims to locate a variety of entities, categorized under 10 distinct tags, which prominently include mentions of food. To ensure accurate validation, a 10-fold cross-validation technique is implemented. This involves training the model on a dataset comprising 270 recipes while earmarking 30 recipes exclusively for testing and validation purposes.

It is important to note the utilization of the BERT language model, which precedes the BiLSTM-CRF model in the process. BERT is employed to generate word representation vectors for the recipes, thereby enriching the input data fed into the BiLSTM-CRF model.

When evaluated to detect various entities, the NER model exhibited commendable performance with precision, recall, and F1 score metrics registering at 94.4%, 93.4%, and 94.1% respectively. However, when the focus narrows to the detection of food mentions specifically, the model's performance showcases even sharper detection. The precision stands at an impressive 96.4%, recall increases to 97.3%, and the F1 score solidly aligns at 96.9%.

2.3.1.3 Ensemble Models for NER Systems. The study presented in Perera et al. (2022) argues that despite significant progress in the realm of biomedical named entity recognition (BioNER), limited attention has been paid to the recognition of food and dietary constituents. Consequently, seven BioNER models, including dictionary-based, Conditional Random Fields (CRF), hybrid (FoodCoNER), and deep learning models such as Bert (Bidirectional Encoder Representations from Transformers), BioBert (Bidirectional Encoder Representations from Transformers for Biomedical Text Mining), Roberta (Robustly Optimized BERT Pre-training Approach) and Electra (Efficiently Learning an Encoder that Classifies Token Replacements Accurately), are examined for food entity detection in this paper. The results revealed the hybrid model, combining the dictionary-based and the CRF approaches, surpasses the other models regarding performance and processing time.

One notable contribution of the paper is its comparative study between the analyzed models and other food NER systems, including BuTTER (Cenikj et al., 2020), NCBO annotator (Popovski et al., 2020), and FoodIE (Popovski et al., 2019), all of which are discussed earlier in this section of the thesis. The paper investigates seven models for food entity detection and incorporates the *FoodBase* dataset for training the CRF model and the four deep learning models. Meanwhile, for the dictionary-based model, an expansive dictionary, housing 20,000 entities from varied sources, is created for exhaustive coverage of food names and dietary constituents. The dictionary consists of 20,000 entities collected from three different resources: Food-BioMaker Ontology (FOBI) (Castellano-Escuder et al., 2020), Food database (FoodDB, n.d.), and Food database (Food Data Central) (Haytowitz & Pehrsson, 2018) after being manually processed to make sure of avoiding repetitions and spelling

mistakes. The types of entities are diverse among scientific terms, generic food and flavor terms, chemical compound names, and phytochemical and nutrition terms.

The dictionary-based model comprises three steps for feature extraction: text cleaning, assigning parts-of-speech (POS) labels to tokens and semantic cleaning. It also includes a matching criteria definition step to match terms and dictionary items. This step is divided into direct matching with dictionary items and partial matching, the latter ensuring that variant expressions of a directly matched term are not overlooked. While the model does not correct mismatches due to spelling errors at this stage, the post-processing step aims to mitigate potential false positives and traces previously identified named entities in different text sections.

The Conditional Random Fields (CRF) model leverages the *FoodBase* dataset (Popovski et al., 2019) for its training phase. At the point of research execution, this dataset represented the gold standard corpus annotated explicitly for food names. To enrich the model, POS tags, dependency relations, and additional contextual information were incorporated. The latter category includes information such as the tokens immediately preceding and following each instance, the POS tags of the two tokens before and after the current token, and the dependency types of the current token along with its direct dependents. The FoodCoNER hybrid model employs two parallel processes: dictionary feature extraction and CRF model feature extraction and evaluation. The inclusion of the dictionary process as a feature allows a reduction in the feature space from 12 to 8. The dictionary features are then utilized in the CRF model feature extraction phase to evaluate the FoodCoNER's performance.

The last four models, Bert, BioBert, RoBerta, and Electra, are deep-learning models fine-tuned for the NER task using the *FoodBase* dataset.

One key contribution of the Perera et al. (2022) paper is the detailed experiment and analysis it provides. Various trials were conducted to measure the impact of dictionary size and quality on model performance and to evaluate different data configurations.

Key findings from the study include:

- FoodCoNER, along with the pre-trained deep learning models, exhibited the highest F1 score of approximately 0.974.
- The learning times for FoodCoNER and the CRF models were significantly shorter than the other models where the fastest deep learning model, Electra, is 43 times slower than FoodCoNER.
- Compared to other systems, FoodCoNER showed superior performance in terms of food entity extraction, particularly with limited dataset sizes and when running time is crucial.
- Table 2-2 presents the F1 scores for detecting the three tags while utilizing the hybrid model, FoodCoNER.

Table 2-2

The F1 Scores for the Three Classes Detected by the FoodCoNER Model

Tag	F1 Score		
	Full FoodBase	Entities Appeared	Entities appeared Once or

	Dataset	Once are Removed	Twice are Removed
B-ingredient	0.9668	0.9698	0.9695
I-ingredient	0.9575	0.9631	0.9646
O	0.9947	0.9952	0.9954

Note. The table demonstrates the F1 scores of the three tags; the three tags include B, signifying the beginning of the food token, I, denoting any food name that follows B, and O, representing all other tokens that are not classified as food entities. Adapted from “Comparison of text mining models for food and dietary constituent named-entity recognition” by N. Perera, T. T. L. Nguyen, M. Dehmer, & F. Emmert-Streib, 2022, *Machine Learning and Knowledge Extraction*, 4(1), 254-275. Copyright 2022 by the authors.

- The following summarizes the performance of the seven examined models against other prominent food Named Entity Recognition (NER) systems based on their F1 Score: 0.974 for the hybrid model, Roberta, and BioBERT, 0.973 for Bert and Electra, 0.968 for the CRF model, 0.957 for the dictionary-based model, 0.639 for the NCBO annotator (Jonquet et al., 2009), 0.961 for FoodIE (Popovski et al., 2019) and for BuTTER (Cenikj et al., 2020).

However, the study acknowledges that its reliance on a relatively small dataset for evaluation could potentially affect the results. Overall, the paper represents a significant contribution to food and dietary constituent NER, introducing a new, efficient model that outperforms other deep learning models. FoodCoNER is highly recommended for use for food entity extraction when a small dataset size is available and the running time is essential with the expectation of the highest performance compared to other proposed systems.

The study detailed in the paper Komariah and Sin (2022) introduces a pioneering methodology for Named Entity Recognition (NER) aimed at extracting food-related entities and associated attributes from a recipe database. This method combines the predictive results of three recurrent classifiers: Recurrent Neural Network (RNN), Gated Recurrent Unit (GRU), and Long Short-Term Memory (LSTM) to form an innovative ensemble NER system, the Recurrent Network-based Ensemble (RNE). This constitutes the first known use of a deep-learning ensemble approach in the food domain.

An additional contribution of the paper is the development of an annotated dataset tailored for food entities, which has been crafted for the training and evaluation of the RNE. Named the Food Ingredient NER (FINER), the dataset has been populated by leveraging a collection of 64,782 recipes extracted from the widely-used website *AllRecipes*. The paper by Komariah et al. (2023) details the full implementation of the deep learning ensemble model used to generate the FINER dataset.

The FINER dataset has been annotated with five entities: ingredient, product, quantity, unit, and state. In order to accommodate multi-word entities, the IOB2 format has been utilized. The dataset comprises a total of 1,397,960 words in 181,970 sentences, annotated with 11 different tags in IOB format. Approximately one-third (32.23%) of named entities pertain to ingredient attributes (B-ingredient and I-ingredient). The word representation employs pre-trained Glove embeddings of 300 dimensions. In the interest of training and evaluation, the dataset is divided into an 80:20 ratio.

To develop the models, the three recurrent networks, Bi-RNN, Bi-GRU, and Bi-LSTM, were trained independently on the FINER dataset, incorporating the pre-trained Glove

word embeddings through a word embedding layer. Each model's predicted output was fed into an ensemble concatenation layer to generate the final prediction. It is worth noting that the paper Komariah and Sin (2022) presents comprehensive evaluation metrics (precision, recall, and F1 score) for each model: Bi-RNN, Bi-GRU, Bi-LSTM, and the RNE, encompassing each of the 11 tags. The F1 score for both types of models (unidirectional and bidirectional) is recorded, and the confusion matrix for each model is also provided.

The research yielded several key results. For example,

- The bidirectional versions of the three models demonstrated superior performance over the unidirectional versions in the NER task, improving the F1 score by 0.2-0.5% in the context of the proposed ensemble network, RNE.
- The Bi-LSTM model architecture was found to be the most effective single model following the RNE.
- The Bi-RNN model reported the least impressive performance, a finding attributed to the simplicity of its implementation.
- The Bi-LSTM model exhibited the shortest training time across all models and was faster than the RNE model by an average of 74.3%. This time difference is to be expected, given the additional resources required to train and maintain an ensemble model.

In accordance with our primary focus on predicting food entities, we have concentrated on two tags: B-ingredient and I-ingredient as demonstrated in Table 2-3.

Table 2-3

The Results of Detecting Food Entities by the Four Experimented Models

Tag	Bi-RNN			Bi-GRU			Bi-LSTM			RNE		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
B-ingredient	0.948	0.971	0.960	0.954	0.974	0.964	0.956	0.971	0.964	0.958	0.977	0.968
I-ingredient	0.970	0.969	0.966	0.978	0.969	0.974	0.971	0.971	0.971	0.979	0.974	0.977

Note. The table illustrates the precision (P), recall (R), and F1 scores associated with the four models under consideration: Bi-RNN, Bi-GRU, Bi-LSTM, and RNE. Adapted from “Enhancing food ingredient named-entity recognition with recurrent network-based ensemble (RNE) model” by K. S. Komariah, & B. K. Sin, 2022, *Applied Sciences*, 12(20), 1-19.

<https://doi.org/10.3390/app122010310>. Copyright 2022 by authors.

In terms of the study's limitations, the RNE model was evaluated using a relatively small dataset, a factor that may have influenced the results. Nevertheless, this study represents a significant contribution to the NER field, particularly within the context of food ingredients and related attributes, as it pioneers an efficient ensemble model in the food domain NER.

Addressing the challenge of extracting and linking food-related entities from scientific literature is the primary motivation behind the paper by Cenikj et al. (2022). Experimental outcomes reveal that attempts to employ existing Named Entity Recognition (NER) systems, which were originally designed for food recipes, have been ineffective when transposed to scientific textual datasets. Consequently, the authors of Cenikj et al. (2022) introduced a novel NER approach tailored for the identification of food mentions within the scientific literature.

This method capitalizes on the fine-tuning of four transformer-based architectures, namely BERT(Bidirectional Encoder Representations from Transformers), RoBERTa (Robustly Optimized BERT pretraining Approach), BioBERT (Biomedical BERT), and SciBERT (Scientific BERT), using a curated database. This database encompasses scientific texts sourced from the abstracts of 500 research articles in PubMed, specifically annotated with food terminologies for this initiative.

To assess the efficacy of the proposed method for the NER task, an ensemble model was developed that combines the outputs of existing NER models specifically designed for food-named entity recognition. This ensemble model termed the "Food Voting Scheme" operates on a voting mechanism that aggregates the class predictions from each constituent model. The ensemble comprises several specialized food NER systems, namely BuTTER (Cenikj et al., 2020), FoodNER (Stojanov et al., 2021), FoodDB nonscientific, and FoodDB scientific. Notably, the FoodDB model employs a dictionary-based approach, leveraging a straightforward string matching strategy to pinpoint food names derived from the FoodDB database.

The operational mechanism of the ensemble model is outlined as follows:

1. Should an entity be identified by the scientific variant of the FoodDB model, it is immediately recognized as a legitimate food entity.
2. In scenarios where this condition is not met, an extracted entity from any other model is authenticated as a valid food entity, provided:
 - a. It is identified by at least two models without any inclusion or omission of terms.

b. Alternatively, the FoodNER model can associate this entity with one among the external semantic repositories: FoodOn, SNOMED CT, Hansard Closet, or Hansard Parent.

Before the start of fine-tuning the transformer models, a series of experimental evaluations were conducted on the distinct NER models for their competency in food entity recognition. This empirical testing was executed on a set of 500 manually annotated scientific abstracts. The rigorous manual annotation ensured each entry was cross-referenced with corresponding semantic tags exist within the Hansard Taxonomy.

The macro-average F1 scores for the NER systems BuTTER, FoodNER, and Food Voting Scheme are 0.42, 0.51, and 0.58, respectively. It's evident that the performance of the NER systems diminishes when evaluated against the scientific corpus. Furthermore, the ensemble model, despite its comprehensive approach, yielded unsatisfactory results on this corpus.

In contrast, the evaluation metrics derived from fine-tuning the transformer models for the NER task demonstrate a notable superior performance. Specifically, the BioBERT model emerges preeminent with a median macro F1 score of 0.90. The other transformer models have the same performance, registering a consistent performance metric of 0.89 for the same evaluation criterion. On the other hand, for the Named Entity Linking (NEL) task, the BioBERT model remains unparalleled. It achieves median macro-averaged F1 scores of 0.66, 0.43, and 0.58 when associating the extracted food entities with the respective semantic datasets of Hansard, FoodOn, and SNOMED CT.

2.3.2 Named Entity Recognition to Detect Dish Mentions

To our understanding, the pioneering work in the area of Named Entity Recognition (NER) for identifying dish names within online consumer reviews was first undertaken in the master's thesis by Tareq (2019). This investigation underscores the critical importance of examining customers' text reviews on digital platforms such as social media. These text reviews are invaluable data that can offer great benefits for both the consumer and business sectors. A focal point of such examinations is the technique of Named Entity Recognition (NER), with a special emphasis on detecting particular entities, such as names of dishes. In confronting the inherent obstacle posed by the scarcity of pre-labeled data sets specifically tuned for dish names, Tareq's study employs a lookup table, which is systematically compiled using the WordNet dictionary. The principal objective behind this innovative methodology is to minimize the necessity for manual labeling efforts to the greatest extent feasible.

In this research, the dataset from Yelp served as the primary source of information. Each customer review was examined, and all nouns within these reviews were treated as potential named entities. To classify what we refer to as "positive samples", a lookup table was employed, which had been generated based on a predefined dictionary. To extend the scope of entity recognition, especially for those entities not labelled in the initial data, techniques from Positive Unlabelled Learning (PU Learning) were integrated into the analysis.

The model isn't solely reliant on the noun in isolation; it also takes into consideration the contextual setting in which the noun appears. Specifically, the model analyzes the words that come both before and after the noun in question, along with the Part of Speech (POS) tag associated with each word.

To ensure data quality and eliminate repetition, any entities that were mentioned multiple times across various reviews were addressed. The average representation for these repetitive entries is calculated so that each named entity is uniquely represented in the dataset. This strategy is aimed at reducing data redundancy while maintaining the integrity and reliability of the results.

In this research, a labeled Yelp dataset of 100,000 restaurant reviews was generated through a multi-step process. The first step involved the creation of a lookup dictionary, which was built by pulling words representing edible food from the WordNet database, resulting in a list of 3,480 terms. Next, the reviews underwent preprocessing and text-cleaning procedures. Each cleaned review was then tokenized and assigned appropriate Part of Speech (POS) tags. Every noun in the review was treated as a potential dish name. In addition, the four tokens preceding and following each noun were noted, along with their respective POS tags.

The third step involved utilizing the lookup dictionary to determine whether each noun appeared in the list of food terms. If a match was found, the noun was labeled with a '1'; if not, it was labeled with a '0'. Following this process, approximately 1.9% of the dataset's tokens received a positive label '1', while the remainder were held unlabeled.

In the fourth step, Positive Unlabeled Learning (PU Learning) techniques were employed to estimate the likelihood of each word in the dataset serving as a named entity, specifically a dish name. The algorithm considered each labeled noun and calculated the probability that the eight adjacent words (four before and four after the noun) might also constitute part of the dish name.

The implementation of PU learning enhanced the model's performance. By taking into account both context and POS tags for each noun, the accuracy rate rose to a satisfactory 67%. Furthermore, the number of identified named entities increased by roughly 15% compared to using just the lookup dictionary. This framework has the advantage of being adaptable and could potentially be applied in domains beyond the identification of food and dish names, underscoring its flexibility as an approach not confined to any specific algorithm.

The research paper Syed and Chung (2021), to our knowledge, represents the most recent contribution to the task of dish name entity recognition within academic literature. This paper advances a domain-adapted BERT-based Named Entity Recognition (NER) approach, tailored for recognizing dish names within a dataset comprising customer reviews. In this section, we explore the findings of this paper for the purpose of comparing them with the outcomes of our research. Moreover, the annotated food menu dataset developed by Syed and Chung (2021) was employed in our comprehensive experimental study.

As stated in Syed and Chung (2021), a significant motivation behind this study was the encouraging outcomes exhibited by the use of contextualized BERT embeddings in various NER systems. Consequently, the core of the proposed NER system rests on the application of BERT word embeddings. This approach is based on generating BERT embeddings in two distinct methods before feeding these embeddings as input to the deep learning model. These methods encompass:

1. Additional training of the original BERT model using an unlabelled customer reviews dataset, effectively achieving domain adaptation of the BERT model.

This approach employs unsupervised learning on unlabelled Yelp reviews to ensure a superior contextual representation of dish entities.

2. Fine-tuning the BERT model on a food menu dataset for the conventional NER task, whereby domain adaptation is accomplished via supervised learning with a small, labeled dataset of dish names.

Subsequently, the BERT embeddings generated from these two versions were supplied as input to three NER network models for further training: the softmax layer fine-tuning model, the BiLSTM+CRF NER model, and the BiLSTM-Att+CRF. Notably, the input was not solely confined to BERT word embedding representations. Part of Speech (POS) tags and character embeddings also played a role in shaping the input to the various models.

Prior to the conduct of the study, ablation tests were conducted to evaluate the influence of different inputs on the performance of the deep learning model. The inclusion of POS tags and character embeddings as input features, complementing the word embeddings, was observed to yield optimal results.

To train these deep learning models, an annotated dataset comprising dish names was created, and extracted from Yelp customer reviews. As per Syed and Chung (2021), this annotated dataset is the first of its kind in the existing literature. The dataset consists of 10,765 annotated sentences, subdivided into train, test, and validation subsets for training and evaluation purposes. Throughout our thesis, we refer to this dataset as *MenuNER* dataset.

In terms of architecture, the BiLSTM-CRF model is composed of two stacked BiLSTM layers, each featuring 512 hidden units, followed by a CRF layer for prediction. An

additional variant of this model, the BiLSTM-Att-CRF, employs a multi-head attention layer post-BiLSTM stack, but preceding the CRF layer.

Key findings of this study include:

- The technique of domain adaptation, achieved through further pretraining of the BERT language model, can be leveraged to enhance the performance of BERT-based NER models within the food domain, enhancing dish name detection.
 - Domain-adapted BERT embeddings led to superior performance results across all three experimental models when compared with the baseline-generated BERT-base embedding.
 - The BiLSTM-CRF model's results exceeded those of other NER models, including the regular Softmax fine-tuning and the BiLSTM-Att-CRF model.
- Table 2-4 includes the most prominent results achieved by the BiLSTM-CRF model for both types of BERT embeddings:

Table 2-4

Best Performance Results of Syed and Chung (2021) Models

BiLSTM-CRF NER Model	Precision	Recall	F1-Score
Bert-Base-Embeddings	90.3	91.3	91.8
Bert-Domain Embedding	92.7	92.3	92.5

Note. The best recorded results in literature for food names NER detection are in bold.

Adapted from “MenuNER: Domain-adapted BERT based NER approach for a domain with

limited dataset and its application to food menu domain”, 2021, *Applied Sciences*, 11(13).

<https://doi.org/10.3390/app11136007>. Copyright 2021 by the authors.

Further discussion of the experimental settings and the description of the *MenuNER* dataset can be found in Section 3.2.2 of the thesis.

In conclusion, the results demonstrated that the approach proposed by Syed and Chung (2021), utilizing a further pre-trained BERT language model on unlabeled customer reviews, can yield the highest F1 score of 92.5% and surpasses the F1 score of 91.8% achieved by the baseline BERT model.

2.4 Deep Learning Neural Networks

This section provides a detailed exploration of the diverse deep learning models employed to build the experimented models within the study. Initially, an explanation is provided for two employed variants of Recurrent Neural Networks (RNNs): namely, Long Short-Term Memory (LSTM) models and Gating Recurrent Units (GRUs). Lastly, an introduction is presented for the Conditional Random Field (CRF) and Convolutional Neural Networks (CNNs).

2.4.1 Recurrent Neural Network (RNN)

RNNs are a type of artificial neural network that is designed mainly for processing sequences of data by maintaining internal states that capture information from previous inputs. Opposite to the popular feedforward neural networks (FFNNs), the RNN architecture allows the demonstration of the dynamic temporal behavior for a time sequence as the processing of input sequences is based on using their internal states (Ali et al., 2018). The difference

between a regular neural network and a recurrent network can be explained by the output of each. For example, the well-known formula for calculating the output of a regular neuron given the input sequence at time t x_i , weight matrix W_i , and the bias b_i where the activation function σ is:

$$y_i^t = \sigma(W_i x_i + b_i)$$

While the formula for the output in the case of using RNN network is adjusted to feed back the output of the previous time step $t-1$. Therefore, the formula will be adjusted to:

$$y_i^t = \sigma(W_i x_i + U_i y_i^{t-1} + b_i)$$

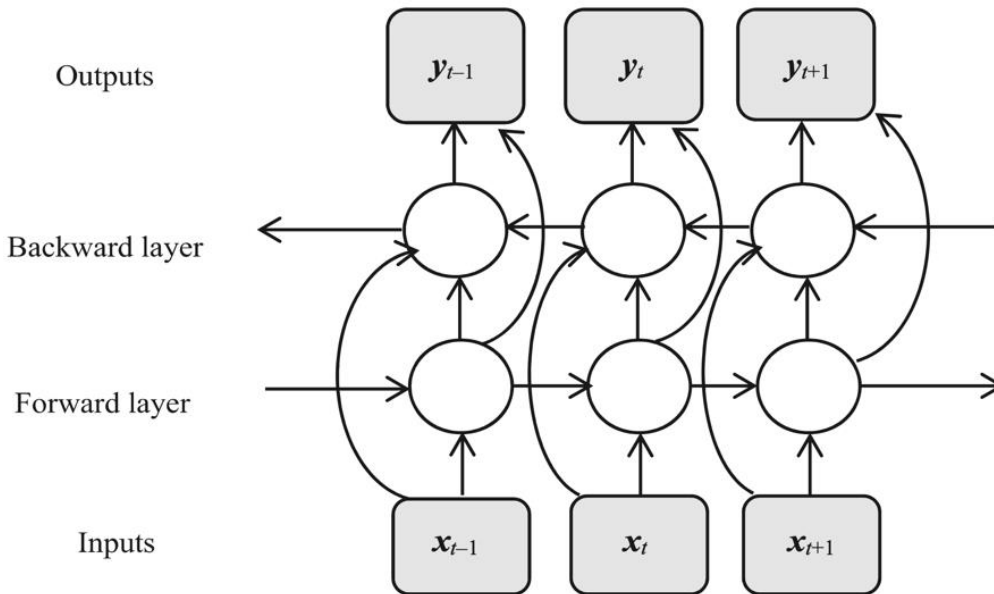
Therefore, for each timestep, the data flow from left to right (default unidirectional) carrying the past information. The ability of RNNs to retain previous sequence information has driven advancements in their application for sequence labeling tasks.

The simple RNN model consists of three layers; the input layer, the hidden layer, and the output layer. The unique property of an RNN relies on the hidden layer which is made of recursive connections that allows propagating the past information from the input data. The hidden layers are also called memory units since they are storing information (Li et al., 2015).

The RNNs models can be run in two modes: forward (unidirectional) and bidirectional. In the bidirectional RNN, the output is not only established based on the past information ($N-1$), but also the future information ($N+1$) is considered (Khan et al. 2020). Figure 2-1 depicts the representation of a bidirectional RNN model:

Figure 2-1

Schematic Representation of a Bidirectional RNN Model



Note. The figure demonstrates the schematic representation of a bidirectional RNN model.

From “Deep recurrent neural networks with word embeddings for Urdu named entity recognition”, by W. Khan, A. Daud, F. Alotaibi, & S. Arafat, 2020, *ETRI Journal*, 42(1), 90-100. Copyright 2020 by ETRI Journal.

In certain situations, considering both previous and upcoming contexts can be advantageous. Bidirectional RNNs analyze sequences from both ends, keeping distinct hidden states for elements from the past and those yet to come.

On the other hand, the RNN suffers from a serious drawback represented by the gradient-vanishing and exploding problem. The RNN is trained based on the Backpropagation Through Time (BPTT) which is a gradient-based approach, and this is the root cause of the

vanishing gradient problem. The vanishing gradient problem occurs when gradients become very small as they are propagated back through time in RNNs. This can lead to the network's inability to effectively capture long-range dependencies in sequences, preventing their ability to learn and generalize from sequence data. The vanishing gradient problem becomes especially pronounced for many sequence-to-sequence tasks, such as machine translation or speech recognition, where understanding the entire context is crucial (Hochreiter & Schmidhuber, 1997).

To address this problem the work of Hochreiter and Schmidhuber (1997) proposed the robust framework Long-Short-Term-Memory (LSTM) to address the vanishing gradient problem that the original RNN is suffering from.

2.4.2 Long Short-Term Models (LSTM)

The paper Hochreiter and Schmidhuber (1997) introduced the LSTM neural network model, made to address long-term dependencies and the vanishing gradient issue seen in recurrent neural networks (RNNs). In 1997, Hochreiter and Schmidhuber presented the LSTM design to tackle the challenges faced by standard RNNs, which struggled with learning over long sequences. The LSTM has a unique structure of memory cells, which consists of three multiplicative units: the input, output, and forget gates. These gates control how information enters and leaves the memory cell, helping to keep important information and discard what's irrelevant and not needed. The three gates are used as follows:

- **Input gate:** Determines the extent to which a piece of information should be stored in the memory cell.

- **Forget gate:** Decides on which proportion of historical past information should be discarded.
- **Output gate:** Controls which proportion of the input information to be transferred to the next time step.

The gates introduced in the LSTM enable it to choose what information to remember or discard. This feature helps it learn and store patterns from extended sequences which is the basic challenge for standard RNNs. Additionally, the design of the LSTM addresses the vanishing gradient problem often seen in regular RNNs. In these RNNs, as gradients are sent backward in time, they undergo repeated multiplications, leading them to either become very small (vanish) or very large (explode). However, with the LSTM's gating system and constant error pathways, gradients can move backward with less decay, promoting more consistent learning over extended periods. Hochreiter and Schmidhuber (1997) in their research presented tests that prove the efficiency of LSTMs. They highlighted that LSTMs are superior to regular RNNs in scenarios where understanding long-term patterns is curial. This research paved the way for more exploration and utilization of the LSTM approach. In the subsequent years, LSTMs became an essential element in deep learning, especially for tasks involving sequences, such as language modeling, machine translation, and speech recognition. This growth emphasized their key role in advancing neural network designs.

The LSTM model processes a sequence of input vectors $x_1, x_2, \dots, x_n$ and produces a sequence of hidden states vectors $h_1, h_2, \dots, h_n$ that corresponds to the context representation vector h_t . Each vector in the latter sequence represents the features learned from the corresponding word w_i at each respective time step t in the input.

The following formulas express the mechanism of updating an LSTM unit as presented by Panchendrarajan and Amaresan (2018) where the element-wise sigmoid function is σ , the element-wise product is $\odot$, the input gate is i , the forget gate is f and output gate is o and the cell vector is c .

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o)$$

$$h_t = o_t \odot \tanh(c_t)$$

Moreover, the weights matrix (W) and the biases (b) as indicated in the formulas are the parameters to learn during training.

When analyzing a sequence of words, the knowledge of both past and future inputs at a given time enables the effective use of features from both the right and left directions. This is denoted as bidirectional LSTM as the input sequence is inputted to a forward LSTM and a backward LSTM to allow capturing the left and right contextual information of the word. The final representation of a word comes from the concatenation of the forward and backward representations.

2.4.3 Gating Recurrent Unit (GRU)

The Gated Recurrent Unit (GRU) represents a modification of the traditional Recurrent Neural Network (RNN) designed to address the vanishing gradient issue and handle long-term

dependencies more effectively. The GRU offers an alternative to the challenges inherent in the original RNN structure and, notably, provides a less complex implementation than the Long Short-Term Memory model (LSTM). Comprising just two gates, the reset and update gates, the GRU controls the flow of information within the hidden units (Cho et al., 2014a).

For a sequence input vector x_t at time-step t , the final output of the hidden neuron of GRU model s_t is calculated as:

$$s_t = (1 - u_t) \odot s_{t-1} + u_t \odot \tilde{s}_t$$

where the following formulas are used for the calculations:

$$u_t = \sigma(W_u x_t + z_u s_{t-1} + b_s)$$

$$r_t = \sigma(W_r x_t + z_r s_{t-1} + b_s)$$

$$\tilde{s}_t = \tanh(W_s x_t + r_t \odot (z_s s_{t-1}) + b_s)$$

Note that $\tilde{s}_t$ formula represents the current content of the hidden state, the computation of the update gate is done using the formula of u_t , and the calculation of the reset gate is represented by the r_t formula. Also, $\odot$ denotes the element-wise product and W and z represent the weights vector.

According to the formulas,

- The **reset gate** of the GRU model is a sigmoid function that takes the input and the previous hidden state as inputs. It decides which parts of the previous hidden state to forget or to ignore. Therefore, if $r_t=0$, then the information in the

current hidden state is ignored, otherwise if $r_t=1$, then all information is considered and carried with the current input.

- Regarding the GRU's **update gate**, it is also implemented as a sigmoid function to decide *how much* of the previous hidden state should be preserved and *how much* of the new candidate information should be added to the new hidden state.

These two gates collaborate to shape the behavior of the GRU cell and direct the flow of information across the recurrent layers. The candidate hidden state, which is determined by the reset gate and current input, is combined with the previous hidden state based on the update gate's output to produce the new hidden state. This new hidden state is then used for predictions or passed to the next time step in the sequence.

For every time step in the sequence, this procedure recurs, enabling the GRU to capture temporal relationships and trends within the data. The final hidden state can be used for making predictions at that time step or passed on to subsequent time steps in the sequence (Cho et al., 2014b).

Based on its simpler design, the time taken for training a GRU model is relatively less than the time taken by LSTM model as it is training fewer parameters (Chung et al., 2014).

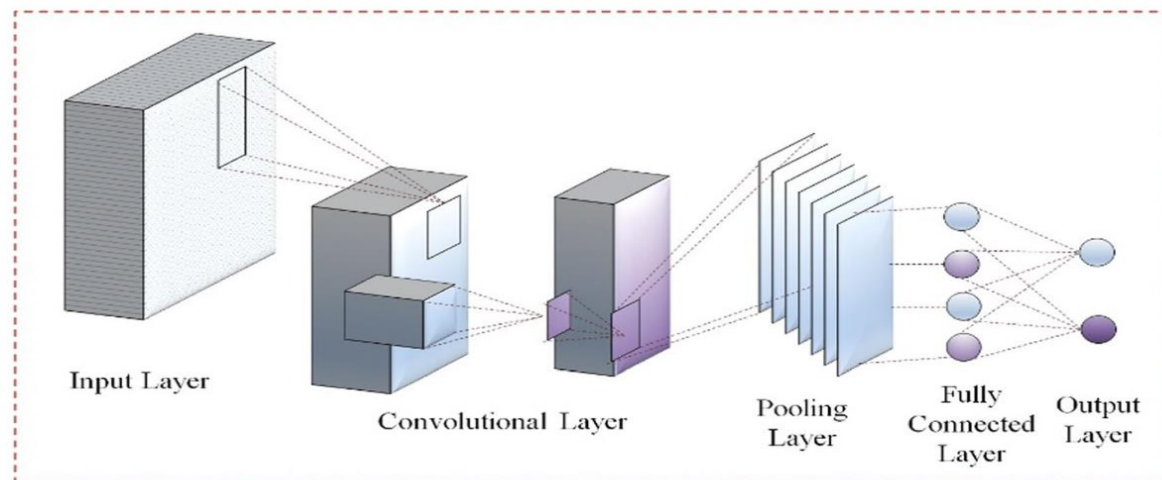
2.4.4 Convolutional Neural Models - (CNN)

Originally designed for image processing and computer vision tasks, Convolutional Neural Networks (CNNs) have achieved substantial success in those domains. More recently, CNNs have attracted increasing interest in the field of Natural Language Processing (NLP). When applied to Named Entity Recognition (NER) tasks, CNNs have shown notable

enhancements compared to traditional methods. Among the diverse range of architectures explored, CNNs have exhibited surpassing outcomes, often outperforming alternative architectures like Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) models (Strubell et al., 2017).

A standard Convolutional Neural Network (CNN) consists of several types of layers, including an input layer, convolutional layers, pooling layers, and fully connected layers. In the context of image processing, the input layer typically receives pixel values from the image. However, when applied to a Natural Language Processing (NLP) task, such as Named Entity Recognition (NER), the input layer is fed with the numerical representations, or word vectors, of the sentence sequence (Zhang et al., 2015). These word vectors are then passed to the convolutional layer, where a collection of filters or kernels slide across the input sequence to identify relevant features.

Subsequent to the convolutional layer, the pooling layer plays a crucial role in the dimensionality reduction of the extracted features. It retains only the most significant features from short sequences of neighboring words, thereby condensing the representation into a more manageable form. This dimensionality reduction performed by the pooling layer serves to accelerate computational processes. Following the pooling layer, the fully connected layers are tasked with classifying the extracted, condensed features into appropriate labels or categories, as detailed in the work of Simran et al. (2020). A graphical representation of the architecture of a CNN layer can be seen in Figure 2-2.

Figure 2-2*Convolutional Neural Network Layer Architecture*

Note. The figure demonstrates the architecture of the convolutional neural network layer.

From “Impact of word embedding models on text analytics in deep learning environment: A review”, by D. S. Asudani, N. K. Nagwani, & P. Singh, 2023, *Artificial Intelligence Review*, 56, pp.10345-10425. Copyright 2023 by Artificial Intelligence Review.

Convolutional Neural Networks (CNNs) present a range of distinct benefits when employed in Named Entity Recognition (NER) tasks. Among these, their computational efficiency stands out as a particularly noteworthy advantage. In contrast to Recurrent Neural Networks (RNNs) or Long Short-Term Memory (LSTM) models, which handle sequences in a temporally dependent fashion, CNNs harness the power of parallelism. During the convolution and pooling stages, they can carry out multiple operations simultaneously. As a result, this inherent parallel processing capability considerably shortens both training and inference

durations, positioning CNNs as a speedier alternative in the domain of NER (Strubell et al., 2017).

An additional benefit of Convolutional Neural Networks (CNNs) resides in their robustness against overfitting. This is largely attributed to the pooling layers, which are instrumental in reducing the dimensionality of the feature set. When combined with a relatively fewer number of model parameters compared to other architectures, this dimensionality reduction enables CNNs to generalize more effectively to previously unseen data. Furthermore, CNNs excel in the hierarchical extraction of features due to their multi-layered structure. This hierarchical design allows CNNs to detect increasingly complex patterns as the data progresses through the layers. Starting from the identification of basic shapes and progressing to more complicated, high-level features, CNNs prove particularly skillful at capturing both syntactic and semantic information, an attribute that is especially advantageous in Named Entity Recognition (NER) tasks (Zhang et al., 2015).

Despite their numerous advantages, Convolutional Neural Networks (CNNs) are not without limitations, particularly when utilized for Named Entity Recognition (NER) tasks. A notable drawback is their limited capability to effectively capture long-range dependencies between words, which can be essential for accurate entity identification in Natural Language Processing (NLP) tasks. While Long Short-Term Memory (LSTM) models are specifically designed to manage such long-range contextual dependencies, CNNs are fundamentally local operators. Consequently, they may struggle to incorporate broader contextual information that can be crucial for accurate NER. Nonetheless, it is imperative to acknowledge that ongoing research efforts are being directed toward mitigating this limitation. Several approaches have

been proposed, including the development of modern CNN variants and configurations designed to capture more expansive contexts. These may involve utilizing larger receptive fields, stacking multiple convolutional layers, or even experimenting with hybrid models that combine CNNs and LSTMs, thereby aiming to harness the strengths of both architectures (Zhang et al., 2015).

An additional constraint associated with Convolutional Neural Networks (CNNs) pertains to their high sensitivity to various hyperparameters. These include but are not limited to, the choice and number of filters, the architecture of the layers, and the types of activation functions employed. The settings for these hyperparameters can substantially influence the effectiveness of the model, often necessitating rigorous tuning to optimize performance for each specific Named Entity Recognition (NER) task at hand (Zhang & Wallace, 2017). Consequently, while CNNs undoubtedly provide an efficient and powerful alternative to more traditional computational approaches, it is crucial to exercise caution and perform due thoroughness in evaluating these intrinsic limitations when considering their application to NER tasks.

2.4.5 Conditional Random Field (CRF)

A Conditional Random Field (CRF) is a statistical modeling technique frequently employed in sequence tagging tasks, including but not limited to part-of-speech (POS) tagging and named entity recognition (NER) (Lafferty et al., 2001). Numerous named entity recognition (NER) systems, particularly those developed using deep learning frameworks, leverage CRFs as a tag decoding layer. This CRF layer is typically positioned as the final layer in the model's architecture, with succeeding layers designed to capture contextual information,

such as BiLSTM and CNN layers (Li et al., 2020). Within the academic literature, various NER models incorporating CRFs for sequence tagging have been proposed. For example, research by Huang et al. (2015), Qiu et al. (2019), and Asghari, Sierra-Sosa and Elmaghraby (2022) have utilized a CRF layer on the top of a BiLSTM layer to assign labels to text sequences. Similarly, studies conducted by Collobert et al. (2011), Strubell et al. (2017), and Sornlertlamvanich and Yuenyong (2022) have integrated a CRF layer following a CNN layer for the same sequence labeling purpose.

The growing use of Conditional Random Fields (CRFs) in labeling named entities can be attributed to their ability to account for dependencies across an entire tag sequence while tagging individual words. For instance, in a labeling task involving food items, a CRF model can capture specific tag dependencies, such as the fact that an 'I-FOOD' token cannot directly follow an 'O' token but must instead follow a 'B-FOOD' token (Sun et al., 2022). This is achieved by considering both the features of the word targeted for labeling and those of its adjacent words. As a result, the inclusion of a CRF layer in a neural network architecture provides the advantage of explicitly capturing tag dependencies, leading to improved accuracy in named entity detection tasks (Li et al., 2020).

The output sequence $x_1, x_2, \dots, x_n$ of a BiLSTM or CNN layer is considered as the input sequence to the CRF layer. The target of the CRF layer is to give the corresponding label to each input in the sequence as $y_1, y_2, \dots, y_n$. Therefore, the CRF score of the sequence can be calculated using (Wang & Su, 2022):

$$s(X, y) = \sum_{i=0}^n A_{y_i, y_{i+1}} + \sum_{i=1}^n P_{i, y_i}$$

where the estimate of moving from label y_i in location i to y_{i+1} in location $i+1$ in the sequence is denoted as $A_{y_i, y_{i+1}}$. Also, the estimate that the mark is labeled as y_i at location i of the sequence is denoted as P_{i, y_i} . Therefore, the estimate of the label sequence Y is calculated in the following formula where $\hat{y}$ corresponds to the true label sequence:

$$p(y|X) = \frac{e^{(X, y)}}{\sum_{\hat{y} \in Y_X} s(X, \hat{y})}$$

To conclude, in our study we have evaluated several NER deep learning models that have utilized different combinations of different deep learning layers: BiLSTM, BiGRU, CNN and CRF. The details of all evaluated deep learning models are in Section 3.4.

2.5 Language Models and Word Embeddings

Language models, especially those that utilize word embeddings, are core elements in many Natural Language Processing (NLP) applications. Their function is to represent words in a way that machine learning algorithms can efficiently process and interpret. Word embeddings can grasp semantic and syntactic details about words, which is extremely beneficial for various NLP tasks such as text classification, sentiment analysis, named entity recognition, machine translation, etc.

The following points are some of the advantages of word embeddings:

- **Semantic and Syntactic Information:** Word embeddings are a dense representation of words that capture the contextual meaning of words. They are able to capture relationships between words, both in terms of semantics (meaning) and syntax (grammatical role) (Mikolov et al., 2013b). For example, the embeddings can capture

that "king" is to "queen" as "man" is to "woman" by performing mathematical operations on the vectors.

- **Dimensionality Reduction:** Traditional bag-of-words models or TF-IDF (Term Frequency - Inverse Document Frequency) models use a sparse representation of the words, where each word is represented by a vector of the size of the entire vocabulary. Word embeddings, however, use a dense representation, significantly reducing the dimensionality of the input data, which can be beneficial for machine learning models (Asudani et al., 2023).
- **Transfer Learning:** Word embeddings trained on large amounts of data can be reused across different tasks, which is a form of transfer learning. This can improve the performance of models, especially when the amount of task-specific data is limited (Howard & Ruder, 2018).
- **Handling Out-of-Vocabulary Words:** Embedding models like FastText can generate embeddings for words that are not present in the training corpus, addressing the issue of out-of-vocabulary words by breaking words into smaller units (character n-grams) and leveraging the internal structure of the words. This feature is particularly useful when dealing with languages that have a rich morphology, or in cases when dealing with misspelled words (Bojanowski et al., 2017).
- **Modeling Sequential Information:** With embeddings as part of larger models like RNNs or transformers, sequential information, and dependencies in the text can be modeled, which is important in many NLP tasks (Vaswani et al., 2017).

- **Improving Model Performance:** Word embeddings have been shown to improve the performance of various NLP tasks, as they can capture the relationships between words and the context in which they are used, which is crucial for understanding natural language (Mikolov et al., 2013a).

Therefore, word embeddings play a crucial role in the current state-of-the-art performances achieved in various NLP tasks. They help bridge the gap between the human understanding of language and the numerical representation required by machine learning algorithms.

In literature, numerous techniques and language models have been proposed for word embedding training. However, for the purpose of our study on named entity recognition, we specifically concentrate on two prominent models: the Neural Probabilistic Language Model and the Contextual Representation Model. These models have been selected due to their relevance and significance in our research.

2.5.1 Neural Probabilistic Language Model

Neural Probabilistic Language Models, often referred to as predictive models, try to anticipate a word based on its surrounding context, or inversely, by employing neural networks (Asudani et al., 2023). These models rely on the distributional hypothesis. This theory asserts that words appearing in similar contexts generally convey similar meanings. A notable example of these models is Word2Vec, which presents two distinct architectural designs: the Continuous Bag of Words (CBOW) and Skip-Gram. Within the CBOW framework, the model anticipates a target word utilizing its contextual words, whereas the

Skip-Gram design anticipates context words from a designated target word. A significant advantage of such models lies in their proficiency to encapsulate complex semantic and syntactic interrelations among words, subsequently representing them in more condensed vector formats (Mikolov et al., 2013b).

At the core of these models is their capability to produce dense vectors for each word. Essentially, these models craft a continuous vector space in which semantically analogous words converge to nearby points. An added pivotal merit of these models is their ability at determining analogical relationships amongst words, exemplified by relationships like 'king' to 'queen' paralleling 'man' to 'woman'. GloVe (Global Vectors for Word Representation), an alternative predictive model, combines the strengths of both count-based and predictive models. It binds statistical data from a corpus to cultivate word representations (Pennington et al., 2014).

In the following sections, more exploration of three exemplary neural probabilistic models utilized in the study is provided: Word2Vec, GloVe, and FastText.

2.5.1.1 Word2Vec. Mikolov et al. (2013a) introduced the Word2Vec model, a paradigm shift in embedding words as high-quality distributed vector representations. These vectors, commonly referred to as word embeddings, encapsulate rich semantic information derived from textual datasets. Furthermore, these embeddings preserve various syntactic and semantic relationships among words.

The Word2Vec model, based on neural network methodologies, has two primary architectures to derive continuous vector representations of words from extensive datasets: Continuous Bag of Words (CBOW) and Skip-gram. Mikolov et al. (2013a) argued that these models when compared against preceding techniques anchored in various neural network

designs, present marked advancements in accuracy while remarkably reducing computational costs. While the attempt to represent words as continuous vectors has a long history, with numerous architectures and training paradigms proposed over time, many of these architectures were computationally expensive. Mikolov and his colleagues sought to build upon these foundational architectures by emphasizing an efficient approach to learning word vectors through a simpler model.

Mikolov et al. (2013a) offer a comprehensive exploration into distributed representations of words derived from neural networks, drawing attention to their enhanced efficacy in comparison to traditional methodologies like Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA). Their discussion centers around two principal models: the Feedforward Neural Net Language Model (NNLM) and the Recurrent Neural Net Language Model (RNNLM).

The NNLM, characterized by its input, projection, hidden, and output layers, has its computational demands primarily concentrated at the hidden and output layers. On the other hand, the RNNLM, conceptualized as an evolution of the NNLM, efficiently mitigates some of the latter's constraints. It is architecturally simpler, comprising only the input, hidden, and output layers, with the bulk of its computational complexity rooted in the hidden layer.

Beyond these two models, the paper also sheds light on the innovative utilization of the DistBelief distributed framework. This large-scale infrastructure facilitates the training of more expansive models, ensuring commendable precision across large datasets. An essential element in DistBelief is its capability to simultaneously operate multiple iterations of a

singular model, employing mini-batch asynchronous gradient descent coupled with an adaptive learning rate procedure, thereby simplifying the training process considerably.

Mikolov et al. (2013a) introduced two pioneering models for obtaining distributed representations of words, both designed with an emphasis on reducing computational complexity. These models, founded upon the principles of neural network language models, are Continuous Bag-of-Words (CBOW) and Continuous Skip-gram. For the Continuous Bag-of-Words (CBOW), the model seeks to predict the likelihood of a word based on its surrounding context. In this architecture, word vectors from the context (which could be words before and/or after the target word) are averaged, effectively projecting them to a common position. This design choice deliberately omits the sequence in which these context words appear. Using this averaged vector, the model employs a log-linear classifier to estimate the current word. In contrary to CBOW, the Skip-gram model employs the current word to anticipate context words that fall within a specified window around the current word. The model takes in the current word and subsequently deploys a log-linear classifier to predict or generate context words. The main goal is to determine surrounding words within a predefined proximity (both preceding and succeeding) of the input word. Both methodologies provide distinct advantages, tailored to specific applications or dataset characteristics.

To assess the efficacy of the word vectors in capturing both semantic and syntactic information, an evaluation was conducted employing a comprehensive test set encompassing five semantic questions and nine syntactic questions. This test set comprised 8,869 semantic questions and 10,675 question queries, all of which were derived from hand-curated pairs of similar words. For a question to be accurately addressed, the evaluation mandates the retrieval

of the exact word in question, without considering synonyms or related terms as valid responses.

The evaluation encompassed various model architectures: RNNLM, NNLM, CBOW, and Skip-gram. These were trained on a corpus sourced from Google News, comprising six billion tokens. The vocabulary was restricted to the one million most frequent words. The primary objective was to compare the word embeddings produced by these models based on their semantic and syntactic precision.

The findings indicated that increasing the dimensions or supplementing more training data yielded incremental enhancements. Notably, both the CBOW and Skip-gram models, when trained on a subset of the Google News dataset, surpassed the performance of other assessed models. In summary, the study underscored the significance of the training architecture, dataset size, and word vector dimensionality in determining the precision of the generated word vectors. Amongst the evaluated models, CBOW and Skip-gram emerged as particularly efficacious.

Building on the work of Mikolov et al. (2013a) to enhance the quality of trained word embeddings and expedite the training process using the continuous Skip-gram model, further advancements were discussed in Mikolov et al. (2013b). This subsequent paper introduced several extension strategies for refining the original Skip-gram model. Notably, the authors proposed a method for subsampling frequent words, which not only accelerated the training process, achieving speeds two to ten times faster than prior rates but also improved the quality of representations for less common words in the vocabulary. Certain words, such as 'the' and 'is', appear with high frequency in most texts. However, their recurrent presence often doesn't

contribute as much contextual significance as rarer words do. When generating word embeddings, this dominance by frequently occurring words can overshadow the learning from less common but potentially more contextually informative terms.

Addressing this challenge, the authors proposed a subsampling method during the model's training phase. This method probabilistically discards each word in the training set based on a formula that factors in the word's frequency in the corpus. Consequently, common words like 'the' face a higher likelihood of being omitted compared to scarcer words. This approach shifts the training emphasis towards the rarer, and typically more contextually meaningful, words. Upon implementing this subsampling strategy, the authors observed marked enhancements in the quality of word vectors, both in terms of their semantic and syntactic attributes.

The "negative sampling" technique serves as an efficient training approach, particularly beneficial for vast datasets. In the foundational Word2Vec models, the Skip-gram model faced computational challenges when predicting the subsequent word in a sentence based on the current word. This is mainly because it had to calculate probabilities for every word in the vocabulary. When dealing with extensive vocabularies, this becomes notably resource-intensive.

Negative sampling addresses this issue by reframing the learning problem. Instead of predicting the next word based on the current one, the technique transforms the objective into a binary classification challenge. Here, the Skip-gram model's goal shifts from the prediction of next word based on the current one to distinguishing the actual word pairs (termed positive samples) from randomly selected, incorrect combinations (denoted as negative samples). By

doing so, the model can efficiently focus on a subset of the entire vocabulary during each training iteration, significantly reducing computational overhead.

In the domain of word embeddings, the ratio of negative samples to positive samples is a crucial hyperparameter. Depending on the dataset's size, this ratio can vary. For smaller datasets, the norm is to opt for 5-20 negative samples for every positive sample. Conversely, for larger datasets, a range of 2-5 negative samples per positive sample is more typical. Negative sampling's chief advantage lies in its ability to considerably trim down the computational demands of training, enabling the model to feasibly process training data and handle vast vocabularies with ease.

Expanding on the enhancements introduced in Mikolov et al. (2013b), the paper also addresses the representation of phrases within a corpus. A notable drawback of the original Word2Vec model is its singular focus on individual words as the primary learning units. This narrow perspective overlooks the intricate semantic interrelationships between certain words that frequently co-occur as phrases, creating specific meanings. Examples of these are 'New York' or 'United States'. In real-world contexts, these terms aren't just independent words but represent unique entities or concepts. Recognizing and embedding such phrases enhances the richness and applicability of the learned embeddings.

Addressing the challenge of encapsulating semantic relationships inherent in multi-word expressions, Mikolov et al. (2013b) proposed an innovative technique designed to detect phrases within textual data. At its core, this algorithm scours the training corpus for bigrams, or pairs of words, that frequently appear, signaling their status as a distinct phrase. The strategy relies on a specifically crafted scoring function that quantifies the association strength

between words. This function assesses the likelihood of two words forming a phrase based on individual word frequencies compared to their combined occurrence frequency.

When such phrases are located, they're encapsulated as singular tokens, and the model's training progresses. Importantly, this mechanism is recursive: after treating bigrams as individual entities, the algorithm can then pair them together to detect even larger cohesive units like trigrams or four-grams. This recursive layering ensures that the model can adapt to and capture the semantics inherent in complex, multi-word constructs.

By adopting this methodology, the model extends its capability to learn meaningful representations for not just isolated words, but also compound expressions. This encapsulates a broader spectrum of semantic and contextual nuances. Demonstrative results presented by the authors affirm the heightened efficacy of this technique, with a marked enhancement in the reliability of the derived word vectors across diverse semantic and syntactic challenges.

2.5.1.2 Glove (Global Vectors for Word Representation). GloVe (Global Vectors for Word Representation), as introduced by Pennington et al. (2014), stands as a word embedding technique developed through unsupervised learning to produce vector representations of words. Distinctively, GloVe strives to incorporate both the global statistical attributes and the local semantic details of a textual corpus. This approach addresses the shortcomings of prior models that leaned predominantly towards one aspect, neglecting the other. For instance, the LSA (Latent Semantic Analysis) technique, while it is good at exploiting global statistical information from the corpus, often results in suboptimal embeddings due to its limitations in the word analogy task. On the other hand, the Skip-gram model, despite its commendable performance in word analogy tasks, operates within a local context window. This orientation tends to overlook the broader co-occurrence statistics of the corpus, thereby not fully capitalizing on its comprehensive statistical potential.

GloVe operates on the foundational principle of co-occurrence matrices. These matrices are instrumental in quantifying the frequency with which words co-appear within a specified window across a text corpus. The term "global vectors" in GloVe emphasizes the model's unique ability to directly learn global statistical information from the corpus when formulating embeddings.

Essentially, GloVe forms a large matrix using this co-occurrence data. Within this matrix, each element presents the likelihood that word i appears in proximity to word j throughout the entire corpus. Following this, the GloVe model employs a weighted least squares methodology to minimize the difference between two values: the dot product of the vectors representing a word pair, and the logarithm of their co-occurrence frequency. This process ensures that the resultant embeddings are representative of both the global statistical properties and complex semantic relationships among words in the corpus.

GloVe's approach produces word vectors that capture various dimensions of a word's meaning, ensuring that semantically similar words are positioned closely within the vector space. This attribute proves invaluable for many natural language processing (NLP) applications, granting models the capacity to understand the semantic associations among diverse words.

The evaluation results for the GloVe model proved its superior performance over baseline benchmark models in several domains, including word analogy, word similarity, and named entity recognition. Such results prove GloVe's proficiency in capturing complex semantic and syntactic relations between words. To evaluate its efficacy across these tasks, GloVe was trained on five extensive corpora of varying sizes. These corpora were sourced

from the 2010 and 2014 Wikipedia dumps, Gigaword 5, a combined dataset from Gigaword 5 and the 2014 Wikipedia dump, and the massive Common Crawl, encompassing one billion, 1.6 billion, 4.3 billion, 6 billion, and a staggering 42 billion tokens, respectively. The Stanford tokenizer was utilized to tokenize these corpora and to convert the text to lowercase. From this, a vocabulary set, limited to the top 400,000 frequent words, was curated. In terms of the word analogy task, GloVe's performance was markedly superior to its competitors. An observation by Pennington et al. (2014) emphasized that GloVe's efficiency scales with corpus size, achieving significant enhancements even with a large corpus comprising 42 billion tokens. Nonetheless, it's worth noting that the expansion in corpus size doesn't guarantee improved outcomes across all tested models in the study.

In assessing word similarity, the procedure involves standardizing each feature across the vocabulary followed by computing the cosine similarity. The efficacy of this method is evaluated using Spearman's rank correlation coefficient, comparing the derived score with human evaluative judgments. Despite training on a corpus that's less than half the size of what was used for CBOW, GloVe's performance surpasses other models in this task.

For the Named Entity Recognition (NER) task, which employs a Conditional Random Field (CRF) based model, GloVe's performance is superior to all the competing models across various evaluation metrics with the only exception being the CoNLL test set. This conclusion of the findings verifies the efficiency of GloVe vectors in downstream NLP applications, mirroring the efficacy previously attributed to neural vectors.

Pennington et al. (2014) conducted a detailed evaluation to compare the performance of GloVe against Word2Vec. This comparison considered a group of essential parameters,

including vector length, the size of the context window, corpus size, vocabulary dimensions, and training duration.

GloVe's training duration is primarily influenced by the number of training iterations, while, for Word2Vec, the number of training epochs is the most relevant factor. An interesting aspect of Word2Vec's coding design is its restriction to a single epoch. As an alternative work to this limitation, the study explored the possibility of varying the number of negative samples. This is based on the understanding that adding more negative samples effectively increases the number of training words the model is exposed to, therefore, acting similarly to additional training epochs.

However, the study underscored an interesting observation: augmenting the negative samples in Word2Vec beyond a count of 10 resulted in diminished performance. This performance degradation was claimed to stem from the negative sampling technique's inability to approximate the desired probability distribution.

In summarizing the evaluation outcomes, the evidence pointed toward GloVe's superiority over Word2Vec when parameters like corpus, vocabulary, window size, and training duration were held constant. GloVe not only achieved superior results in lesser training time but also obtained the top performance overall, irrespective of training speed. This places GloVe in a favorable position, suggesting its efficiency for training word embeddings compared to Word2Vec within the tested parameters (Pennington et al., 2014).

2.5.1.3 FastText. FastText was introduced as a novel method for learning word representations through the work of the Facebook AI Research lab team (Bojanowski et al., 2017). This innovative method generates word embeddings by considering character-level information of words, more specifically their internal structures or "subwords". The

researchers proposed a model architecture that treats each word as a bag of character n-grams, and the words are represented as the sum of the vector representations of their constituent n-grams.

This methodology addresses some of the limitations in the existing word embedding techniques that treat each word as an individual entity and do not take into consideration their morphological richness. In the context of languages that are morphologically complex, where words may have many inflected forms such as Spanish language, the proposed approach shows significant promise. It takes into consideration the many forms and inflections, which would typically be a challenge for conventional models to accurately capture. By representing words as a sum of these character n-gram vectors, the model is able to generate word representations that capture this morphological information.

Another advantage of the FastText embedding is the effectiveness of its approach in dealing with out-of-vocabulary words that are rarely available in the training corpus. When words are broken down into character n-grams, it becomes possible for the model to generate reasonable representations for words that were not available in the training corpus. This is a noteworthy advancement as it addresses a common issue in natural language processing where the least occurring words can often become noise in the data and are inaccurately represented.

FastText word embeddings are generated by a modified version of the Word2Vec model: the skip-gram model. The skip-gram model is extended by treating each word as a group of character n-grams, allowing it to capture information about subword units. To train this model, FastText uses a negative sampling objective which formulates the problem as a binary classification task - given a pair of words, the model needs to determine if they occur in

the same context or not. Also, the model can generate vector representation for the words that don't exist in the training dataset by averaging the embeddings of its n-grams.

According to Bojanowski et al. (2017), the method for generating the FastText embedding is evaluated using nine different languages after training the FastText and other baseline models (CBOW and Skip-gram) on datasets extracted from the Wikipedia dump. The evaluation was carried out to check the performance on different tasks such as word similarity and word analogy. For the word similarity task, the evaluation was done using Spearman's rank correlation to compare human judgment and vector representation cosine similarity. It is found that the model, which uses subword information, outperformed the baselines on almost all datasets, demonstrating the advantage of using subword information. It is also found that character n-grams were especially beneficial for languages like Arabic, German, and Russian, due to their complex grammatical structures. Lastly, their approach excelled on the English Rare Words (RW) dataset, suggesting that character-level similarities can aid in learning good word vectors for rare words.

On the other hand, the FastText approach has been assessed on word analogy tasks, using datasets in English, Czech, German, and Italian. Questions involving words not in their training corpus were excluded. The results showed that morphological information substantially improved performance on syntactic tasks, outperforming baseline models. However, it did not assist with semantic questions and even reduced performance for German and Italian. The choice of character n-gram length was found to be closely related to this effect, with optimal n-gram size reducing semantic analogy degradation. Also, the FastText

model's improvement over baselines was more notable for languages like Czech and German as they are considered as rich morphological languages.

Overall, the approach followed for generating FastText embedding highlights the importance of utilizing subword information in generating word vectors. This approach is beneficial in various natural language processing tasks and can be applied to other machine learning tasks where text morphological content is crucial.

The methodologies introduced by Mikolov et al. (2013a) and Mikolov et al. (2013b), encompassing the Word2vec models (CBOW and Skip-gram), along with the techniques presented by Pennington, Socher and Manning (2014), such as negative sampling and frequent word subsampling, have acquired widespread adoption in later studies. Presently, they serve as bases for a lot of natural language processing (NLP) tasks and applications (Asudani et al., 2023).

2.5.2 Contextual Representation Model

In the domain of Natural Language Processing (NLP), contextual representation models have garnered considerable attention due to their ability to generate word representations that are sensitive to context. Contrary to traditional word embedding methods such as Word2Vec and GloVe, which generate static, context-independent embeddings for each word, contextual representation models produce dynamic embeddings. These dynamic embeddings vary based on the surrounding words in a given sentence or document. Such models are trained on extensive datasets, enabling them to grasp the word usage in varying contexts. As a result, these models have proven to be exceptionally effective in a broad

spectrum of NLP tasks, ranging from text classification and sentiment analysis to machine translation (Devlin et al., 2019).

One noteworthy advantage of contextual representation models is their adaptability. After the initial training phase, these models can be fine-tuned with a modest amount of task-specific data, thereby aligning the model's capabilities with the requirements of the task at hand. This fine-tuning process is both time-efficient and cost-effective, making these models a valuable asset for diverse NLP applications (Radford et al., 2019).

Among the notable instances of contextual representation models is BERT (Bidirectional Encoder Representations from Transformers), a model developed by Google. BERT utilizes a transformer-based architecture and considers the entire context of a word by examining the words that precede and follow it. This bi-directional context awareness allows BERT to generate word representations that are highly specific to the context, thereby enhancing its effectiveness in various NLP tasks (Devlin et al., 2019).

Another significant contribution to this field is the GPT (Generative Pretrained Transformer) series of models, including GPT-2 and GPT-3, developed by OpenAI. These models are autoregressive, which means they generate word representations by taking into account all the preceding words in a text sequence (Radford et al., 2019).

Additional variants of Contextual Representation Models exist, such as ELMo (Embeddings from Language Models). ELMo creates context-sensitive embeddings by leveraging the internal states of a deep bidirectional language model (Peters et al., 2018).

In the subsequent section, the focus will shift to an in-depth exploration of BERT, which not only stands as a prominent example of contextual representation models but is also a key component in our research study.

2.5.2.1 BERT (Bidirectional Encoder Representations from Transformers). In the study by Devlin et al. (2019), the authors, affiliated with Google, present a comprehensive introduction to BERT (Bidirectional Encoder Representations from Transformers). This innovative method for pre-training language comprehension models has subsequently contributed to substantial advancements in the state of the art across an extensive array of natural language processing tasks.

At the heart of BERT's innovation, as presented in the study by Devlin et al. (2019), is its utilization of a bidirectional transformer architecture. Unlike preceding language models, which typically process text sequences in a unidirectional manner (either left-to-right or right-to-left), BERT considers the entire sequence of words simultaneously from both directions. This simultaneous consideration of context from preceding and following words offers a deeper understanding of each word within its context. Consequently, this bidirectional approach allows BERT to develop a more comprehensive representation of language.

The study by Devlin et al. (2019) sheds light on the emerging effectiveness of pre-training language models to enhance a wide spectrum of natural language processing (NLP) tasks. These tasks range from sentence-level tasks, such as determining the relationship between sentences (natural language inference) and rephrasing sentences (paraphrasing), to token-level challenges like identifying specific entities in a text (named entity recognition) and deriving answers to specific questions (question answering).

Typically, the integration of pre-trained language representations into these NLP tasks follows one of two predominant strategies: the feature-based approach or the fine-tuning approach. In the feature-based methodology, exemplified by models like ELMo (Embeddings from Language Models), task-specific architectures are designed wherein pre-trained representations are utilized as supplementary features. This is different from the fine-tuning methodology, exemplified by the Generative Pre-trained Transformer (GPT). Here, only a limited set of task-specific parameters are introduced. Instead of creating a new architecture for the task, the model tweaks or "fine-tunes" the parameters of the pre-trained model to adapt to the downstream tasks.

However, a significant challenge associated with both strategies is the inherent unidirectional processing of standard language models. Such a linear approach can pose complications, especially for sentence-level tasks. Additionally, when applied through the fine-tuning strategy, this unidirectional nature may adversely affect the model's performance on token-level tasks.

BERT's bidirectional approach addresses the shortcomings associated with the unidirectional methods. Unlike traditional models which process words in sequence, BERT's transformer encoder reviews the entire word sequence concurrently. This allows the model to understand a word by considering the context provided by its surrounding words. At the heart of this process is the transformer's attention mechanism. This mechanism measures the importance of each word in a sequence relative to the word being examined, ensuring that the model correctly interprets the context in which a word appears. To effectively train BERT and generate concise vector representations for each word within a sentence, the researchers

devised an innovative task known as the masked language model (MLM). In this task, certain words in a sentence are deliberately masked or hidden. The model's challenge is to predict these masked words based solely on their surrounding context. The MLM allows the generated word representation to integrate context from both the preceding and succeeding words in the sentence, a process not possible in left-to-right model training. This deep bidirectional pre-training offered by BERT outperforms traditional unidirectional pre-training methods.

BERT's Masked Language Model (MLM) approach distinguishes itself by focusing on predicting parts of the input that are intentionally obscured or masked. In particular, 15% of all WordPiece tokens within a sequence are selected at random to be masked. This contrasts with other methods where the aim is often to reconstruct an entire input sequence.

However, a unique challenge arises with MLM: during the pre-training phase, a [MASK] token is used to hide certain words, but this token doesn't show up during the fine-tuning phase. This discrepancy could hinder the model's performance. To counter this potential issue, the developers of BERT implemented an approach for masking. Specifically, when a token is selected to be masked, it doesn't always get replaced with the [MASK] token. Instead, it takes on the [MASK] token in 80% of instances, is replaced by a random token in 10% of instances, and remains unchanged in the remaining 10%. Despite these variations, the model's objective stays the same: predict the original token. Such a diverse masking strategy ensures that BERT remains robust and facilitates truly bidirectional learning.

Furthermore, BERT incorporates a “Next Sentence Prediction” task (NSP) during its pre-training phase, designed to enhance its ability to comprehend the relationship between pairs of sentences. This understanding is essential for various natural language processing

tasks. To shed light on the Next Sentence Prediction (NSP) task: it involves presenting the model with two sentences, termed A and B. In 50% of instances, sentence B genuinely succeeds sentence A in the original text, and is thus labeled “IsNext”. Conversely, in the remaining cases, sentence B is randomly chosen from the corpus and does not directly follow sentence A, earning it the label “NotNext”. This exercise equips the model with the discernment to differentiate between coherent and incoherent sentence pairs, an ability that becomes invaluable for specific tasks, notably Question Answering (QA) and Natural Language Inference (NLI).

It's worth noting that the original BERT model derives its pre-training from an extensive corpus encompassing two major sources the BooksCorpus, and English Wikipedia with 800M and 2,500M words respectively.

The primary sizes of BERT models are introduced by Devlin et al. (2019) with the given specifications in the Table 2-5:

Table 2-5

BERT Models

BERT Model	Number of Layers (L)	Number of Hidden Units (H)	Number of Self- Attention Heads (A)	Total Parameters
BERT Base	12	768	12	110 million
BERT Large	24	1024	16	340 million

Note. Adapted from “BERT: Pre-training of deep bidirectional transformers for language understanding”, by J. Devlin, M. W. Chang, K. Lee, & K. Toutanova, 2019, In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (Vol. 1, pp. 4171-4186). Copyright 2019 by the authors.

Subsequent research endeavors, both from independent scholars and institutional teams, have given rise to a variety of BERT derivatives. These adaptations primarily focus on optimizing the architecture for computational efficiency or tailoring it for specialized tasks. Noteworthy among these are:

- **DistilBERT (Distilled Bidirectional Encoder Representations from Transformers):** This variant is approximately half the size of the original BERT-Base model, featuring around 66 million parameters. Developed by Sanh et al. (2019), DistilBERT serves as a streamlined version of the original BERT, designed to maintain a substantial proportion of its predecessor's performance capabilities while offering advantages in speed and resource requirements.
- **TinyBERT:** A further scaled-down model, TinyBERT is even more compact than DistilBERT. It is engineered chiefly for resource-efficient deployment, particularly in environments with limited computational resources (Jiao et al., 2020).

Both DistilBERT and TinyBERT navigate the complicated balance between model size and performance efficacy. These models present resource-efficient alternatives that preserve a considerable extent of the original BERT's functional capabilities.

2.6 Character Embeddings

Deep learning has indeed revolutionized the field of Named Entity Recognition (NER), delivering markedly superior performance by autonomously extracting features directly from the data. One critical component that significantly contributes to this elevated level of performance is the incorporation of character embeddings. The utilization of character-level embeddings provides the distinct advantage of learning task-specific and domain-specific representations. These embeddings have proven to be particularly advantageous in the context of morphologically rich languages, where they can capture intricate linguistic details. Moreover, they offer a compelling solution for addressing challenges related to out-of-vocabulary words, a common issue in tasks like part-of-speech tagging (Kim et al., 2016).

In contrast to word embeddings, which furnish vector representations for whole words, character embeddings provide these representations at a much more granular, character-by-character level. This finer level of granularity comes with several unique benefits. One of the most prominent advantages is the effective management of out-of-vocabulary (OOV) words, defined as words that were not part of the original training data. Traditional word embedding techniques such as Word2Vec (Mikolov et al., 2013a) or GloVe (Pennington et al., 2014) are inherently constrained by their predefined vocabularies, which limits their ability to adequately represent words not found in their training sets. On the contrary, character embeddings empower the model to construct vector representations for any word by examining its individual characters. This effectively sidesteps the limitations imposed by the OOV issue, enabling the model to generalize better to previously unseen words (Kim et al., 2016).

Secondly, character embeddings are particularly skilled at capturing the subtle morphological information that resides in certain languages. Morphologically rich languages often employ a combination of roots, prefixes, and suffixes to infuse words with intricate meanings. When utilizing character embeddings, the model can dissect the structure of a word, thus enabling it to differentiate between entities that share the same morphological root but are modified by different prefixes or suffixes. This capability is invaluable for languages characterized by complex morphological structures, and it substantially aids in the accurate identification of various forms of named entities (Augustyniak et al., 2021).

Thirdly, the incorporation of character-level representations provides the model with a heightened level of adaptability across a diverse range of languages. This is especially useful for languages that may not have extensive libraries of pre-trained word-level embeddings. Research conducted by scholars such as Lample et al. (2016) has substantiated the efficacy of character embeddings in settings where resources for specific languages are limited, thereby attesting to their broad applicability and universality.

Fourthly, the fine-grained nature of character embeddings plays a pivotal role in recognizing sub-word information, a feature that proves invaluable for identifying abbreviations, acronyms, and even newly coined terms. Named entities frequently encompass these specialized forms of language, and their unique characteristics can be more precisely captured when utilizing character-level embeddings. Research conducted by Ma and Hovy (2016) has provided empirical evidence to demonstrate how this granular capability can be harnessed for comprehensive end-to-end sequence labeling tasks, which explicitly includes the challenges associated with Named Entity Recognition.

Additionally, the incorporation of character embeddings alongside traditional word embeddings serves to significantly augment the model's capacity for contextual understanding. This multi-layered representation captures a broader range of syntactic and semantic cues, thereby facilitating more accurate recognition of named entities. Typically, these character-level embeddings are input into specialized neural network layers tailored for specific kinds of information processing. For example, Character-level Convolutional Neural Networks (CharCNNs) are adept at learning spatial hierarchies among the characters, while recurrent structures like Long Short-Term Memory networks (LSTMs) are designed to capture temporal dependencies between characters. The output from these specialized layers is then combined with the word-level embeddings. The result is a synthesized, enriched representation that is channeled into the core Named Entity Recognition (NER) model for further processing, as evidenced by research conducted by Reimers and Gurevych (2017).

In conclusion, character embeddings play an indispensable role in the performance of deep learning-based Named Entity Recognition systems. They tackle numerous challenges such as handling OOV words and capturing morphological and sub-word information. The feature richness that character embeddings add is not just incremental but often transformational, setting new benchmarks in NER tasks.

2.7 POS and Syntactic Information

The role of parts of speech in natural language processing is particularly important due to the wealth of information these tags offer about a word and its adjacent terms. Incorporating Part-of-Speech (POS) tags when training deep learning models for Named Entity Recognition (NER) tasks has the potential to considerably boost both the accuracy and efficiency of the

resulting model. Although deep learning architectures such as Recurrent Neural Networks (RNNs) and Transformer models are inherently capable of identifying complex patterns in data, the inclusion of POS tags serves to enrich the model's understanding with an added layer of syntactic and semantic details. This augmented data dimensionality can further optimize the performance of these advanced NER systems (Syed & Chung, 2021).

Firstly, POS tags serve as a valuable aid for a model in understanding the contextual use of a word within a sentence, thus enhancing its capacity for accurate entity recognition. For example, identifying a word as a proper noun can guide the model toward categorizing it as a specialized named entity, such as a company name, rather than mistaking it for a generic common noun. This level of discernment becomes especially critical in eliminating ambiguity when the same word can function as both a common and proper noun, as indicated in studies such as those by Cai et al. (2019).

Secondly, POS tags act as a powerful tool for disambiguation. Given that words in natural languages can possess multiple meanings based on their contextual placement in a sentence, a POS tag can offer crucial insights into the intended meaning of a word. When integrated with deep learning technologies, this additional layer of information can substantially boost the accuracy of named entity recognition. Furthermore, POS tags contribute to enriching the set of features available for training the model. While existing word embeddings like Word2Vec or GloVe are effective in capturing semantic details, they generally fall short in providing syntactic ones. Adding POS tags to the set of inputs introduces this missing syntactic layer, resulting in a more comprehensive and richer representation of the input data. Research, such as that by Güngör et al. (2018), has shown that

models trained with this augmented set of features tend to outperform their counterparts that lack such features.

Moreover, efficiency is another key benefit of incorporating POS tags. Deep learning models are often resource-intensive, particularly when they need to process vast datasets. The syntactic guidance provided by POS tags can help these models to generalize more effectively from smaller datasets. This can reduce the computational resources needed, thereby potentially accelerating the training process and increasing overall efficiency, as highlighted by Cai et al. (2019).

Finally, the utilization of POS tags can significantly contribute to reducing errors in Named Entity Recognition. These tags offer an extra layer of data that can be invaluable in minimizing both false positives and false negatives. This becomes particularly useful in the context of complex sentences, where the immediate linguistic environment may not provide enough clues for accurate entity recognition. Named entities are frequently proper nouns that occupy specific syntactic roles within sentences. Identifying these roles can thus serve as an additional validation step in the entity recognition process, as demonstrated by research such as that by Yi et al. (2023).

In conclusion, the integration of POS tags into deep learning models for NER not only enriches the feature set but also significantly boosts the model's performance by aiding in contextual understanding, disambiguation, and generalization. Therefore, POS tags are not simply optional but are a powerful tool for enhancing the robustness and accuracy of Named Entity Recognition systems.

It's important to note that in the scope of our study, the initial baseline model we employed, specifically a BiLSTM-CRF model, did not include Part-of-Speech (POS) tags as part of its input features. In contrast, one of our significant contributions lies in incorporating these POS tags into the text reviews, which serve as input to all the models we experimented with. Our findings indicate that this incorporation led to a substantial improvement in model performance. Detailed results of the conducted experiments are in Chapter 4 of the Thesis. For a deeper understanding of the meaning of the POS tags utilized, further information can be found in Appendix A.

2.8 Research Gaps

The literature review conducted on Named Entity Recognition (NER) techniques specifically targeting food mentions reveals a diverse set of approaches that have been explored and evaluated within the food domain. Rule-based methods, for instance, have been adopted in systems such as DrNER (Eftimov et al., 2017) and FoodIE (Popovski et al., 2019). Dictionary-based approaches were investigated by researchers like Popovski et al. (2020), while Tareq (2019) applied a machine learning technique known as Positive Unlabeled learning for detecting dish names.

Furthermore, deep learning models have also gained prominence in the development of NER systems for food entities. Researchers like Cenikj et al. (2020), Stojanov et al. (2021), Syed and Chung (2021), Cenikj et al. (2022), and Fan and Hunter (2023) have employed such models in their work. More specifically, they've employed BiLSTM-CRF models for the NER task. Beyond custom-developed solutions, some studies have leveraged existing general-purpose NER systems for detecting food names. For example, Diwan et al. (2020) utilized the

Stanford NER Tagger, whereas Lee et al. (2021) and Chenze et al. (2022) implemented SpaCY for this purpose.

Lastly, the literature also documents the utilization of ensemble models in the construction of NER systems tailored for the food domain. One notable example is FoodCoNER, developed by Perera et al. (2022), which melds dictionary-based feature extraction with Conditional Random Fields (CRF). Similarly, the RNE model proposed by Komariah and Sin (2022) incorporates a deep learning ensemble consisting of BiLSTM, BiRNN, and BiGRU layers. The Food Voting Scheme model presented by Cenikj et al. (2022) employs a more holistic approach, amalgamating the BuTTER model (Cenikj et al., 2020), FoodNER (Stojanov et al., 2021), and both scientific and non-scientific FoodDB dictionaries.

Conversely, a majority of the models explored in the literature have been primarily evaluated using food recipe datasets. Researchers such as Popovski et al. (2019), Cenikj et al. (2020), Diwan et al. (2020), Stojanov et al. (2021), Perera et al. (2022), and Komariah and Sin (2022) have followed this approach. In contrast, the DrNER model was constructed using text from dietary recommendations, and Cenikj et al. (2022) trained and evaluated their model on a dataset comprising scientific texts.

Moreover, a smaller subset of studies has ventured into the realm of unstructured text data sourced from social media platforms. Specifically, Lee et al. (2021) and Chenze et al. (2022) focused on extracting food mentions from Twitter posts. Tareq (2019) and Syed and Chung (2021), on the other hand, concentrated on detecting dish names in Yelp reviews.

Additionally, an analysis of the literature on named entity recognition systems designed for identifying dish names in customer reviews highlights limited but noteworthy

contributions. Two research studies stand out in this regard. Tareq (2019) implemented the Positive Unlabeled learning algorithm in their study. Syed and Chung (2021) took a two-pronged approach using Bert embeddings. Initially, they used Bert embeddings as inputs to a BiLSTM-CRF deep learning model, achieving an F1 score of 91.8. Subsequently, they conducted further training of the Bert model on Yelp customer reviews in an unsupervised fashion. The newly generated embeddings were then utilized as inputs to a BiLSTM-CRF model, which yielded the highest recorded F1-score of 92.5 in the existing literature.

In summary, the existing body of literature on named entity recognition in the food domain reveals several gaps and limitations that warrant further exploration:

- **A predominance of food recipe datasets:** Although various models have been proposed for named entity recognition within the food domain, a substantial portion of these focus on extracting food entities specifically from food recipe datasets.
- **Limited focus on customer reviews:** Only a small number of research studies, specifically Tareq (2019) and Syed and Chung (2021), have explored the task of detecting dish names within the context of restaurant customer reviews.
- **Narrow scope of deep learning models:** While recent research has shown an inclination towards employing deep learning models for NER tasks in the food domain, there is an observed lack of experimentation with different types of deep learning models, such as Convolutional Neural Networks (CNN).

- **Absence of domain adaptation studies:** Notably, no studies in the surveyed literature have explored the efficacy of applying domain adaptation from food name detection in recipes to dish name detection in customer reviews.
- **Scarcity of annotated datasets:** The field suffers from a notable absence of standardized, annotated datasets that focus on dish name detection. During the course of this thesis, only one such dataset was found, which was shared by Syed and Chung (2021).
- **Computational intensity of best-performing model:** The model that achieved the best result for detecting dish names from customer reviews, with an F1-score of 92.5, did so at the cost of significant computational resources. This model, developed by Syed and Chung (2021), required further training of a Bert model on customer reviews, resulting in a complex deep-learning architecture that involved fine-tuning over 40 million parameters during its training phase.

These identified gaps and limitations serve as a guide for future research directions, calling attention to unexplored areas and challenging issues in the domain of named entity recognition for food mentions.

Chapter 3 : Research Methodology

Chapter 3 is organized into four main sections, each serving a specific purpose. At the beginning, the annotation process of our dataset, *ReviewsDB*, is detailed, and a comprehensive overview of its characteristics is presented. Also, the characteristics of the other two utilized datasets, *FoodBase* and *MenuNER* are demonstrated. The primary evaluation metrics employed in our study and the optimization and regularization techniques adopted by the deep learning models for the NER task are explained. Finally, the Chapter concluded by meticulously outlining the different architectures of the models experimented in our study.

3.1 Our *ReviewsDB* Dataset

At the time of starting our research, the absence of a suitable gold standard annotated corpus specifically tailored for training machine learning algorithms to detect dish names encouraged us to create and annotate our own dataset. To construct the dataset, 1000 Yelp reviews were randomly selected from the Yelp database to form the dish name annotated dataset namely *ReviewsDB*.

The semi-manual annotation process is completed in two primary stages:

1. Initial automatic annotation leveraging the WordNet dictionary.
2. Manual review and annotation focused specifically on identifying the missed dish names within the dataset and correct mistakes in previous annotation step.

The annotation scheme aligns with the *FoodBase* corpus annotation in terms of class names. However, the annotation specifically targets dish names rather than general food

names. Consequently, food names unrelated to dish names are excluded and annotated with the "O" class. The classes used in the annotation are as follows:

- **B-FOOD:** marks the beginning of a dish name.
- **I-FOOD:** represents the inside or part of a dish name.
- **O:** indicates a token that is not part of a dish name.

Our primary *ReviewsDB* dataset, consisting of 1000 reviews, exhibits a significant class imbalance, similar to the *FoodBase* dataset. The percentage of dish names within the entire dataset is 6.2%, encompassing both B-FOOD and I-FOOD classes. However, it is important to highlight that our annotated dataset is smaller in size compared to *FoodBase* and *MenuNER* datasets.

For the experiments we're conducting, we divide the *ReviewsDB* dataset into five separate folds to enable cross-validation. We initially explore two distinct data splitting methodologies: an 80/20 split and a 90/10 split.

80/20 Split:

- **Training Set:** comprises 800 reviews, which is 80% of the dataset.
- **Testing Set:** includes 200 reviews, accounting for the remaining 20% of the dataset.

90/10 Split:

- **Training Set:** consists of 900 reviews, making up 90% of the dataset.
- **Testing Set:** contains 100 reviews, which represent the remaining 10%.

Our goal in utilizing these two different splitting methods is to identify which approach provides the most accurate and reliable outcomes for the models we are evaluating, particularly given the imbalanced nature of our dataset. After running identical experiments using both the 80/20 and 90/10 splits, we found that the 90/10 division yielded slightly superior performance results compared to the 80/20 option. As a result, we have chosen to adopt the 90/10 split for all subsequent experiments in our study.

The following table, Table 3-1, demonstrates the distribution of dish names within each fold of *ReviewsDB* dataset.

Table 3-1*Dish Names Distribution in Each Fold of Our ReviewsDB Dataset*

Dataset		Total no. of Tokens	Dish/Part of Dish Name Tokens	Percent of Dish/Part of Dish Name Tokens
Fold 1	Train 1	111,556	7,083	6.35%
	Test 1	12,100	765	6.32%
Fold 2	Train 2	109,572	7,009	6.40%
	Test 2	14,085	839	5.96%
Fold 3	Train 3	111,628	7,041	6.35%
	Test 3	12,029	807	6.71%
Fold 4	Train 4	110,878	6,971	6.29%
	Test 4	12,779	877	6.86%
Fold 5	Train 5	111,342	7,156	6.43%
	Test 5	12,315	692	5.62%

It is worth noting that all experiments conducted for the proposed NER models adhere to a 5-fold cross-validation approach. The models were trained on a training set denoted as Train_n and evaluated on a corresponding test set Test_n , where $n \in \{1,2,3,4,5\}$. This methodology ensures consistent and reliable results across the models, providing confidence that their performance remains stable. By training the models on five distinct folds, we can ensure that their deployment in production settings will yield similar performance outcomes.

On the other hand, dataset creation involves several stages to prepare it for the study. The initial stage consists of extracting restaurant reviews exclusively from the Yelp database.

Subsequently, the second stage entails the cleaning and preprocessing of the reviews, employing standard techniques such as tokenization and POS tagging. Detailed information regarding the Yelp database and the extraction process can be found in Appendix A.

We applied the following preprocessing steps to the extracted reviews:

1. Removal of line breaks (new lines '\n') from the reviews, if present.
2. Development of a custom food dictionary for automatic annotation of food names in the reviews. The approach suggested by Tareq (2019) was followed to create this dictionary. The dictionary is constructed based on WordNet, an extensive lexical database for the English language. WordNet organizes English words, including nouns, verbs, adjectives, and adverbs, into sets called synsets, with each set representing a distinct concept. Synsets primarily exhibit a synonymy relationship among their constituent words. Definitions and examples are provided for each word within a synset, with words potentially belonging to multiple synsets if they have different meanings (Princeton University, 2010).

Furthermore, it is important to note that nouns synsets in WordNet exhibit two major relationships: hyperonymy and meronymy. Hyperonymy, also known as the super-subordinate or ISA relation, links specific concepts to more general ones. For instance, a specific term like "bed" is associated with the broader concept of "furniture" (bed is a furniture) (Princeton University, 2010). On the other hand, meronymy represents a part-whole relationship among synsets, connecting parts or components to the larger whole they belong. For example, "finger" is a meronym of "hand" (Fellbaum, 2005).

By exploring WordNet for food names, three synsets were identified, with two of them related to edible food (Synset '*food.n.01*' and Synset '*food.n.02*'), while the third synset (Synset '*food.n.03*') pertains to non-edible food and encompasses synonyms associated with providing mental stimulation for reasoning and thinking (Fellbaum, 2005).

To utilize WordNet synsets, the NLTK (Natural Language Toolkit) toolkit in Python was employed for various operations enabling the creation of a food dictionary (NLTK, 2023). These operations involved accessing and extracting food nouns from the two edible synsets by iterating through the synonyms within each set and extracting nouns that exhibit a hyponym relationship denoted by "is a". As a result, 2,177 nouns were extracted from Synset '*food.n.01*' and 1,621 nouns were extracted from Synset '*food.n.02*'. The extracted nouns underwent preprocessing steps, including the removal of single-character words and duplicates, resulting in a single dictionary encompassing a total of 3,480 food items. The words were sorted in ascending order and converted to lowercase. Additionally, punctuation symbols were replaced with space characters. The final dictionary comprises food items consisting of single words (e.g., "apple") or multiple words (e.g., "apple pie").

3. Tokenization of the reviews and tagging each token with its corresponding part-of-speech (POS) tag using the NLTK library. This stage leads to the dataset containing two columns: Token and POS tag.

4. Creation of a new column named "is_food". Each token is iterated over, and if it represents a non-noun part of speech, it is assigned a value of 0. If the token is

a noun, it undergoes a check to determine if it matches any entry in the lookup dictionary. If a match is found, the value of the "is_food" feature for that token is set to 1; otherwise, it is assigned a value of 0.

5. Manual review of the dataset, involving the following actions:

- Verification of whether the automatically annotated food nouns also correspond to dish names.
- Annotation of any dish names that were missed during the initial annotation step. In this stage, the "is_food" feature is transformed into the "Class" name, which includes three possible classes: B-FOOD, I-FOOD, or O.

6. Second review of dataset annotation is conducted to ensure that the labelling process is correct and consistent.

7. Division of dataset into five folds using the splitting option 90/10.

3.2 Other Datasets

In the course of our research, we employed another two distinct datasets for our experiments: the *FoodBase*, developed by Popovski et al. (2019); the *MenuNER* dataset, specifically crafted for recognizing dish names in customer reviews by Syed and Chung (2021). The subsequent sections provide comprehensive details of the dataset and a comparison between the two customer review datasets is detailed.

3.2.1 FoodBase Dataset

The *FoodBase* corpus, as presented in the paper by Popovski et al. (2019), represents a significant contribution to the field of Natural Language Processing (NLP) with a specific focus on food entities. The dataset provides a comprehensive resource for researchers interested in food-related NLP tasks. One of the primary objectives of the *FoodBase* corpus is to facilitate the annotation of food entities, which has broad applications ranging from health informatics to recommender systems and more.

The dataset comprises a diverse set of food-related entities annotated to ensure high accuracy. What sets this corpus apart is its robust coverage of various types of food items, including but not limited to fruits, vegetables, meats, and dairy products. This diversity allows for a comprehensive understanding of food-related terms, thereby enabling more accurate NLP models focused on this domain. It is highly curated, taking into consideration different aspects of food items such as common names, scientific classifications, and even culinary usage.

In addition to the above features, the *FoodBase* corpus is structured in a way that makes it highly usable for machine learning models. The annotations are provided in a standardized format, making it easier to integrate the dataset into various NLP pipelines. This dataset doesn't just stop at naming food items; it also contains important contextual information that could be beneficial for semantic understanding. Such context might include how a particular food item is commonly used, its nutritional aspects, or its cultural relevance, providing an extra layer of detail that most similar databases lack.

The precisely curated version of the *FoodBase* dataset comprises 1,000 recipes extorted from *Allrecipes* website, each of which has undergone extensive processing steps to

accurately extract and annotate food entities using Hansard food semantic tags. For the purpose of our experiments, we employ the gold standard version of the *FoodBase* corpus, which is both readily accessible and conveniently formatted for immediate use in our analytical framework.

The annotated *FoodBase* dataset is partitioned into two distinct subsets: one for training and another for testing. Within the training subset, a noteworthy feature is that 11.04% of the annotated tokens are specifically dedicated to food names, which underscores the dataset's inherent class imbalance. This skewed distribution stems from the dominance of the 'O' class as compared to the 'B-FOOD' and 'I-FOOD' classes. To clarify, each class designation represents the following:

- B-FOOD: denotes the beginning of a food name.
- I-FOOD: indicates a token that is part of the food name.
- O: represents a token that falls outside the boundaries of any food name.

Importantly, the dataset is already tokenized and meticulously structured into two columns: 'Token' and 'Class Type' with the latter indicating one of the three class labels (B-FOOD, I-FOOD, or O).

3.2.2 MenuNER Dataset

The *MenuNER* dataset, as described by Syed and Chung (2021), is derived from Yelp restaurant customer reviews. The reviews are extracted and annotated using the WebAnno2⁷ tool, following the IOB (Inside, Outside, and Beginning) format for annotation. The dataset is divided into three subsets: Train (5,517 sentences), Valid (3,037 sentences), and Test (2,211 sentences) datasets. Table 3-2 offers a more detailed analysis, revealing the number of unique dish names and partial dish names present in each dataset, along with the total count as illustrated in the work of Syed and Chung (2021).

Table 3-2

Number of Dish Names/Part of Dish Names in the Original MenuNER Dataset

Dataset Features	Train	Valid	Test
Unique Dishes Names	1,027	637	516
Total Dishes Names	2,626	1,424	1,106

Note. Adapted from “MenuNER: Domain-adapted BERT based NER approach for a domain with limited dataset and its application to food menu domain”, 2021, *Applied Sciences*, 11(13).

<https://doi.org/10.3390/app11136007>. Copyright 2021 by the authors.

⁷ <https://webanno.github.io/webanno/>

In the dataset, each sentence is separated by whitespace to distinguish it from the next sentence. Additionally, each dataset consists of four features as follows:

- **Token:** represent the individual tokens.
- **POS tag:** the part of speech tags and there are seven values appeared in the datasets: ['O', 'B-NP', 'B-VP', 'I-VP', 'I-NP', 'I-PP', 'B-PP']
- **Chunk tag:** the more systematic information are augmented withing this tag and there are 44 values within the datasets: ['RB', 'VBP', 'PDT', 'PRP', 'VB', '\$', '""', 'NNPS', 'CD', 'VBG', 'LS', 'UH', ':', 'RBR', 'RP', '`', 'RBS', '#', 'IN', 'VBZ', 'POS', 'FW', '!', 'VBD', '(', 'JJS', 'DT', 'NN', 'JJR', 'NNS', 'JJ', ',', 'WRB', 'MD', 'WP', 'SYM', 'TO', 'NNP', 'VBN', ')', 'PRP\$', 'EX', 'CC', 'WDT']
- **Food Tag (Class):** the label of each entity is one of three tags: B-MENU, I-MENU, or O.

In the three datasets, it is observed that approximately 5% of the annotated tokens correspond to dish names or parts of dish names, while the remaining 95% represent non-dish tokens. This distribution highlights the relatively small proportion of dish-related terms in comparison to the overall dataset as shown in Table 3-3

Table 3-3*Original Class Distribution in Original MenuNER Datasets*

Class	Train	Test	Valid
B-MENU	3.06%	2.95%	3.39%
I-MENU	1.92%	1.94%	2.03%
O	95.02%	95.11%	94.58%

This detailed description of the *MenuNER* dataset sheds light on its composition and provides essential information regarding the number of sentences, unique dish names, and overall dish name occurrences within each dataset subset.

To facilitate our experiments using a 5-fold cross-validation methodology, we opted to merge the Train, Valid, and Test datasets into a single comprehensive dataset.

Additionally, each dataset was further divided into five folds, consisting of five training folds and five testing folds. This division into folds allows for comprehensive evaluation and validation of the model's performance across different subsets of the data. In each fold, 90% of the sentences are designated for training the Named Entity Recognition (NER) model, while the remaining 10% are reserved for testing and performance evaluation.

Based on the Table 3-4, it is evident that the distribution of dish names within our created folds is closely aligned with the distribution in the original dataset introduced by Syed and Chung (2021). Specifically, approximately 5% of tokens represent dish names, while the remaining 95% pertain to non-dish names.

Table 3-4*Dish Names Distribution in Each Fold of MenuNER Dataset*

Dataset		Total No. of Words	Dish/Part of Dish Name Words	%Dish/Part of Dish Name Words
Fold 1	Train 1	151,270	7,621	5.0%
	Test 1	16,770	845	5.0%
Fold 2	Train 2	150,514	7,571	5.0%
	Test 2	17,526	895	5.1%
Fold 3	Train 3	152,509	7,739	5.1%
	Test 3	15,531	727	4.7%
Fold 4	Train 4	150,754	7,420	4.9%
	Test 4	17,286	1,046	6.1%
Fold 5	Train 5	150,837	7,645	5.1%
	Test 5	17,203	821	4.8%

To ensure compatibility with our existing code, the following preprocessing steps were implemented for each fold:

1. Each token was assigned to its respective sentence by introducing a new column called "Sentence Number" which indicates the sentence to which each token belongs.
2. Extra whitespace between consecutive sentences was removed.

3. The class labels were renamed to align with those employed in earlier experiments, specifically as B-FOOD, I-FOOD, and O.

3.2.3 Comparative Analysis of Customer Review Datasets

The following characteristics are shared between the two customer reviews datasets: *ReviewsDB* and *MenuNER*, utilized in our experiments:

1. **Source:** Both datasets were originally compiled by extracting customer reviews of restaurants from the Yelp platform.
2. **Folding Structure:** Each dataset is partitioned into five folds, facilitating the use of 5-fold cross-validation for model evaluation.
3. **Data Split:** In each of the five folds, both datasets employ a 90/10 split. Specifically, in the *MenuNER* dataset, 90% of the sentences are allocated for training, while the remaining 10% are used for testing. Similarly, in the *ReviewsDB* dataset, 90% of the reviews are designated for training purposes, and the remaining 10% are set aside for testing.

Conversely, there are several noteworthy differences between the two datasets, *MenuNER* and *ReviewsDB*:

1. **Dataset Size:** *MenuNER* comprises a total of 168,804 tokens when aggregating its original training, validation, and testing sets. In contrast, the 1000 reviews in *ReviewsDB* amount to 123,658 tokens. Consequently, the token count in each of the five folds of *MenuNER* exceeds that in *ReviewsDB*.

2. **Class Distribution:** In each fold of *MenuNER*, only 5% of the tokens are representative of either complete dish names or parts thereof, while the remaining 95% of tokens do not relate to dish names. In the *ReviewsDB* dataset, an average of 6.4% of labeled tokens signify either full dish names or their components, leaving 93.6% of the tokens to represent other types of information.

3. **Text Preprocessing:** While constructing the *ReviewsDB* dataset, all words in the Yelp reviews were converted to lowercase during preprocessing. On the other hand, *MenuNER* preserves the original casing of the words, without applying any such transformations.

4. **Specificity of food items:** *MenuNER* often includes more traditional or specialized dishes like naan, curry, and tandoori whereas *ReviewsDB* tends to feature more general and Western food items like ice cream, pizza, and fries.

3.3 NER Model Evaluation and Optimization

In this section, the focus is on two crucial components of the Named Entity Recognition (NER) model developed using deep learning techniques: namely, model evaluation and model optimization. The first section provides a comprehensive discussion of the evaluation metrics utilized for assessing the models' performance and the cross-validation technique that has been employed for validation purposes. The second section delves into various optimization techniques and regularization methods that were implemented to mitigate the risk of model overfitting.

3.3.1 Model Evaluation

Deep learning architectures, including Recurrent Neural Networks (RNNs), Long Short-Term Memory Networks (LSTMs), and Gated Recurrent Unit Networks (GRUs), have been extensively utilized in the development of Named Entity Recognition (NER) systems. However, the implementation of these models constitutes merely one dimension in a complex landscape. Another dimension of equal importance is the rigorous evaluation of these models to ensure they meet the desired standards of accuracy, reliability, and efficiency. The significance of model evaluation in this context can be understood through multiple aspects, each of which carries considerable weight for researchers and developers.

First and foremost, the issue of accuracy is of paramount importance in NER systems. Although deep learning models are famous for their predictive competence, they are not protected from pitfalls such as inaccuracies or overfitting. These shortcomings are often intensified by the inherent complexity of these models and the large number of parameters requiring optimization during the training phase. An overfit model may demonstrate impressive performance metrics on the training dataset but is likely to underperform when subjected to previously unseen data.

To mitigate such risks and enhance the model's generalization capabilities, a comprehensive suite of evaluation metrics is crucial. These metrics not only facilitate the quantification of a model's performance but also identify potential areas warranting further investigation or refinement. By employing these evaluation tools, researchers and practitioners can significantly augment the reliability and practical applicability of NER systems, thereby contributing to the broader advancement of the field (Grandini et al., 2020).

Secondly, computational resource optimization is another crucial aspect to consider. Deep learning models are often resource-intensive, requiring substantial memory and CPU (central Processing Unit) or GPU (Graphics Processing Unit) power for both training and inference. Model evaluation can help optimize these resources by identifying how well the model performs relative to its computational requirements. This is particularly significant for real-world deployments where computational resources are not unlimited (Justus et al., 2018).

Thirdly, we can't overlook issues of fairness and ethics. A full review of NER models can show if they have biases in how they recognize things. These biases could keep harmful stereotypes or unfair practices going. For example, a model that mostly learned from data about one group of people might not work well for a broader group. Identifying such biases is the first step toward creating more equitable AI systems (Mehrabi et al., 2021).

Lastly, model evaluation is not just a supplementary phase but an integral part of implementing NER systems that use deep learning. It assures the stakeholders of the model's accuracy, optimizes resources, helps to discover and rectify biases, facilitates comparison between different models, and is instrumental in building user trust. Failure to properly evaluate could compromise any of these critical aspects, leading to less effective and potentially biased systems.

In the next sections, the used evaluation metrics to evaluate our NER deep learning models are detailed in addition to an explanation of the cross-validation technique that has been used in our study to overcome and prevent the evaluated models from overfitting.

3.3.1.1 Evaluation Metrics. Evaluation metrics are essential in Named Entity Recognition (NER) to measure the performance of NER systems and models. The evaluation metrics utilized in our research are the conventional ones used for assessing named entity recognition systems, namely precision, recall, and F1 score.

Before the explanation of these metrics, it is beneficial to clarify the meaning of the key terms TP (True Positive), FN (False Negative), and FP (False Positive) in terms of named entity recognition of dishes names. These terms are illustrated in Table 3-5.

Table 3-5*The Interpretation of TP, FN, and FP*

Key Terms	Meaning
TP (True Positive)	The number of identified <i>entities as</i> dish names (or part of dish names) and they are dish names (or part of dish names). In other words, it is a <u>correct detection</u> by the model. For example, if the sentence is "I ordered a delicious pizza" and the model correctly identifies "pizza" as a dish name, it would be considered a <i>true positive</i>.
FN (False Negative)	The number of identified <i>entities as</i> not dish names (or part of dish names) but they are dish names (or part of dish names). It represents a <u>missed detection</u> by the model. For instance, if the sentence is "The restaurant served a variety of dishes, including spaghetti" but the model fails to identify "spaghetti" as a dish name, it would be a <i>false negative</i>.
FP (False Positive)	The number of <i>identified entities as</i> dish names (or part of dish names) and they are not dish names (or part of dish names). It denotes an <u>incorrect detection</u> by the model. For example, if the sentence is "She ate a delicious apple" and the model mistakenly identifies "apple" as a dish name when it refers to the fruit, it would be a false positive.

To evaluate the performance of the named entity recognition system, precision and recall are calculated based on these values.

Precision measures the proportion of correctly identified dish names out of all the terms identified by the model:

$$\text{precision} = \frac{TP}{(TP + FP)}$$

While recall measures the proportion of correctly identified dish names out of all the actual dish names in the text. In other words, the recall reveals how many of the predicted entities are correct:

$$\text{recall} = \frac{TP}{(TP + FN)}$$

When a trade-off between precision and recall in the evaluation of named entity recognition is required, the F1-score is introduced. The F1-score is a metric that combines precision and recall into a single value. It is calculated using the following formula:

$$\text{F1 - score} = \frac{2 \times \text{precision} \times \text{recall}}{(\text{precision} + \text{recall})}$$

The F1-score ranges between 0 and 1, with 1 indicating the best possible score. A high F1-score signifies a balanced performance between precision and recall. It penalizes models that demonstrate a significant difference between precision and recall.

In the context of evaluating a named entity recognition task, precision, recall, and F1-score are typically computed by comparing the model's predictions to the ground truth

annotations. The model assigns labels to the tokens in a sequence, and the evaluation involves comparing these predicted labels with the true labels provided in the dataset.

In addition to evaluating each individual class using precision, recall, and F1-score, the study also utilizes macro-averaged precision, recall, and F1-score to facilitate a comprehensive comparison of the overall performance across different experimented named entity recognition (NER) models.

The macro-average precision, recall, and F1-score offer valuable insights into the overall performance and efficiency of different NER models, allowing for meaningful comparisons. It is a technique that calculates performance measures for each class individually and then takes the average across all classes, providing an aggregated view. In the evaluation of NER models, macro-averaged precision, recall, and F1-score are commonly employed (Grandini et al., 2020). In addition, to establish a fair comparison between our experimental results and existing NER models, it is essential to utilize the same evaluation metrics that have been reported in previous studies. This ensures consistency and allows for a meaningful and unbiased assessment of the performance across different models. By employing the same evaluation metrics, we can effectively compare the effectiveness and efficacy of our approach with established NER models and draw valid conclusions.

3.3.1.2 Cross Validation. Cross-validation is a widely used technique in machine learning for evaluating the predictive performance and assessing how well it will generalize to unseen data, including Named Entity Recognition (NER) systems. It is typically utilized when data availability is limited. It involves segregating the dataset into training and testing subsets to both develop and evaluate models (Raschka, 2018).

Among the various techniques for cross-validation, k -fold cross-validation stands out for its versatility and robustness. In k -fold cross-validation, the dataset is randomly divided into k equally sized folds with the model being trained on $k - 1$ folds while the remaining fold is used for testing. The process is repeated k times, using each fold once as the testing set, and the performance metrics, such as accuracy or F1-score, are calculated for each iteration and subsequently averaged to provide a reliable estimate of the model's overall performance (Szeghalmy & Fazekas, 2023).

Stratified k -fold cross-validation, a specialized version of k -fold cross-validation, offers a solution to the issue of class imbalance in datasets. Unlike standard k -fold cross-validation, where the folds are created randomly without considering the class distribution, stratified k -fold ensures that each fold has a similar proportion of each target class as the original dataset. This method preserves the class distribution across folds, providing a more accurate and reliable assessment of model performance, particularly when dealing with imbalanced classes (Szeghalmy & Fazekas, 2023).

In stratified k -fold cross-validation, the dataset is initially sorted into groups based on the target classes. Each of these groups is then subdivided into k folds, ensuring that the distribution of classes within each fold closely mirrors that of the entire dataset. Following this, the model undergoes k cycles of training and validation. Unlike standard k -fold cross-validation, the data used for each training and validation cycle is deliberately chosen to maintain a balanced or near-balanced distribution of each target class, thereby providing a more accurate representation of the model's performance (Yates et al., 2023).

The key advantage of using stratified k-fold cross-validation lies in its capacity to yield more reliable performance metrics, especially for datasets with class imbalances. A skewed class distribution can distort performance metrics, often making the model appear more effective than it truly is, particularly if it predominantly predicts the majority class. Stratified k-fold cross-validation addresses this issue by ensuring that each fold is representative of the overall class distribution, thereby facilitating a more robust and credible model evaluation (Yates et al., 2023). However, it is important to note that this method can be computationally expensive, particularly when applied to datasets with a large number of classes or extreme class imbalances. Furthermore, like its standard k-fold counterpart, stratified k-fold cross-validation operates under the assumption that data points are independently and identically distributed, an assumption that may not hold in specialized data types such as time series (Raschka, 2018).

In conclusion, cross-validation remains an integral aspect of model assessment and selection in machine learning, offering a rigorous evaluation framework that has stood the test of time.

The 5-cross validation technique is used to evaluate all experimented models in our study. For either the *MenuNER* dataset or our semi-manual annotated dataset, *ReviewsDB*, the dataset is split into five distinct folds. Each fold holds 90% of the reviews in the dataset to test it on the remaining 10%. A model is trained on a fold in which 10% of it is used as a validation set during each epoch of training. After training the model, it is tested on another unseen fold of data which holds 10% of the reviews. The evaluation metrics are recorded after each iteration. After the fifth round of training and testing, the recorded metrics values are

averaged, including precision, recall, and F1-score. It is important to remember that the 5-fold cross validation experiments are conducted for each experiment model multiple times in which each time a different word embedding model is utilized.

3.3.2 Model Optimization and Regularization

This section is devoted to providing an overview of the optimizers and regularization techniques utilized by the deep learning models in this study. The first section considers two employed optimizers: RMSProp, and Adam. Then, the subsequent section elaborates on the regularization approaches, namely early stopping and dropout.

3.3.2.1 Optimization. Optimization is a crucial process in the training and performance of deep learning models. It involves a tightly coupled interaction with a designated objective function, commonly known as the loss function. The objective is to minimize this function, although, in some instances, it may be desirable to maximize it. Through iterative cycles of optimization, the model fine-tunes its internal parameters, thereby enhancing its predictive accuracy. When this process is executed effectively, the resulting trained model is capable of generalizing well to new, unseen data, making it highly valuable for real-world applications (Goodfellow et al., 2016).

Stochastic Gradient Descent (SGD) and its variants, such as Adam and RMSprop, stand as the de facto optimizers in the deep learning domain. These algorithms dynamically update the model's parameters in a way that incrementally minimizes the associated loss function. Their role becomes particularly important in the context of deep learning, where models often comprise millions, or even billions, of adjustable parameters and are trained on

vast datasets. Conversely, inadequately optimized models can lead to multiple of drawbacks: they may be grossly inefficient, struggle to learn meaningful patterns from the data or consume an unsustainable amount of computational resources and time during the training phase (Kingma & Ba, 2015).

The selection of an optimization algorithm, along with its associated hyperparameters, can exert a profound influence on both the training dynamics and ultimate performance of a deep learning model. Different optimizers traverse the loss function in unique ways, with some being notably good at avoiding local minima, a frequent challenge in the non-convex optimization problems often present in deep learning scenarios (Sutskever et al., 2013). For example, the Adam algorithm combines the strengths of both AdaGrad (Adaptive Gradient Algorithm) and RMSprop, enabling adaptive learning rate adjustments for individual parameters. This approach not only facilitates quicker convergence but also enhances the model's capacity for generalization (Kingma & Ba, 2015).

Optimization serves as a foundational pillar in the architecture of deep learning models. Optimization is important for enhancing model interpretability and robustness as well. Regularized optimization techniques, for instance, can act as preventative measures against overfitting, thereby rendering the model more resilient to data noise (Srivastava et al., 2013).

As deep learning is a swiftly advancing field, an ongoing influx of innovative optimization techniques continues to emerge, each designed to tackle specific challenges inherent in training complex models.

In the following sections, a concise introduction is offered to the two optimization algorithms that were experimentally evaluated in our study: RMSprop and Adam.

3.3.2.1.1 RMSprop Optimizer. RMSprop (Root Mean Square Propagation) is an optimization algorithm specifically designed for the challenges inherent in training deep neural networks. It was first proposed by Hinton et al. (2012) in an online lecture. The algorithm has gained popularity for its efficiency, adaptability, and not being published. However, it is very well-known and many deep learning models have employed RMSprop in their implementation.

It is particularly well-suited for dealing with non-convex optimization problems, which are widespread in deep learning applications.

RMSProp tries to address the problem facing the AdaGrad optimizer where the learning rate goes to zero during the learning process (Sun et al., 2020). The core idea behind RMSprop is to adapt the learning rates during the training process. RMSprop accomplishes this by keeping a moving average of the square of the gradients. This moving average effectively normalizes the gradients, ensuring that each parameter is updated with its specifically adapted learning rate. The key task is to scale down the step size for parameters that are receiving high gradients and scale up for those receiving small gradients. This dynamic adjustment of the learning rate makes RMSprop particularly useful for problems where optimization is required for deep learning models (Sun et al., 2020).

RMSprop has a lot of advantages. Its adaptability allows it to be applicable to a variety of non-convex optimization problems. It usually converges faster than standard stochastic gradient descent algorithms and is less sensitive to the choice of initial learning rate. However, RMSprop was introduced in an online course and therefore lacks thorough theoretical

justification, although it performs well empirically. The algorithm is also sensitive to the choice of its hyperparameters, especially the decay factor β .

RMSprop adjusts the learning rate for each parameter individually and maintains a moving average of the squared gradients for each parameter. By using this moving average, it scales the gradients to normalize them, thereby ensuring that each parameter is updated with an adaptive learning rate.

Mathematically, the RMSprop algorithm can be explained via following formulas (Sun et al., 2020):

- The gradient g_t is computed by the formula: $g_t = \nabla f(\theta_t)$, where f is the loss function with respect to the parameter θ at time step t .
- The moving average of the square gradient, v_t , is updated as follows:

$$v_t = \beta \cdot v_{t-1} + (1 - \beta) \cdot g_t^2$$

where β is the decay factor for the moving average which typically has a value around 0.9

- Therefore, the parameter, θ , is updated using the formula:

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{v_t + \epsilon}} \cdot g_t$$

Where α is the learning rate, ϵ is a small constant to prevent division by zero.

In our research, we initially employed the RMSprop optimizer as a baseline for evaluating the performance of our proposed deep learning models. Subsequently, we transitioned to using the Adam optimizer, which demonstrated superior efficiency by

significantly improving both the learning speed and the overall performance metrics of the models.

3.3.2.1.2 Adam Optimizer. The work of Kingma and Ba (2015) first introduced the adaptive moment estimation optimization algorithm, Adam, as a computationally efficient method for deep learning models. The paper proposes Adam as an algorithm that combines the advantages of the two other well-known optimization methods: Adaptive Gradient Algorithm (Adagrad) and Root Mean Square Propagation (RMSprop).

According to Kingma and Ba (2015), Adam essentially combines the best properties of Adagrad and RMSprop. It computes adaptive learning rates for each parameter and is robust to both sparse gradients and non-stationary objectives. Also, Adam was found to be more efficient than most other stochastic optimization methods in terms of computation time. The extensive conducted empirical tests to validate Adam, including both non-convex optimization problems related to neural networks and convex optimization problems, showed that the algorithm performs well even with its default hyperparameter settings, making it user-friendly for researchers who may not have the time or expertise to fine-tune hyperparameters.

The Adam optimizer is an extension of stochastic gradient descent (SGD) that computes adaptive learning rates for each parameter. Mathematically, it combines elements of both Momentum and RMSprop as per the following formulas (Kingma & Ba, 2015):

- The gradient g_t is computed by the formula: $g_t = \nabla f(\theta_t)$, where f is the loss function with respect to parameter θ at time step t .

- The estimate of the biased first moment of gradients(m_t) and the biased second moment of gradients-RMSprop (v_t) are updated via the formulas:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t ,$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$$

Where β_1 and β_2 are the exponential decay rates for first and second moment estimates

- $\widehat{m}_t$ and $\widehat{v}_t$ are bias corrected first and second moment estimates, respectively.

$$\widehat{m}_t = \frac{m_t}{1 - \beta_1^t}, \widehat{v}_t = \frac{v_t}{(1 - \beta_2^t)}$$

- The computation of the parameter update ($\Delta\theta_t$):

$$\Delta\theta_t = -\alpha \cdot \frac{\widehat{m}_t}{\sqrt{\widehat{v}_t} + \varepsilon}$$

- Therefore, the iterative formula is:

$$\theta_{t-1} = \theta_t - \alpha \Delta\theta_t$$

Where α is the learning rate, the hyperparameter to control the step size of the updates and ε is a small constant to prevent division by zero.

On the other hand, Adam has been observed to be sensitive under certain noisy conditions, which can cause the optimization to diverge from the optimal solution.

The primary Adam algorithm has been researched by several studies and other variants were proposed by modifying the main algorithm, such as Adam with L2 regularization, AdamW, AdamP, etc. (Abdulkadirov et al., 2023).

3.3.2.1.3 Regularization. Deep learning models, particularly neural networks, are susceptible to overfitting, especially when the model is complex and the amount of data is limited. Regularization methods in deep learning are created to prevent overfitting, which happens when a model becomes perfect in performing remarkably well on the training data but struggles to apply the knowledge to novel unseen data. These techniques limit the model from accommodating irrelevant patterns present in the training data. Regularization techniques are commonly employed to mitigate overfitting such as L1/L2 regularization, dropout, early stopping, data augmentation, etc. The choice of regularization techniques depends on the specific problem to be solved, the architecture of the neural network model, and the characteristics of the dataset. Often, a combination of multiple techniques can yield optimal results (Tian & Zhang, 2022).

Our deep learning models primarily employ two techniques: early stopping and dropout. The following sections research and describe these methods.

3.3.2.1.4 Early Stopping. Early stopping is a widely used regularization technique that has the advantage of simplicity and effectiveness. Early stopping involves halting the training process before the model starts to overfit. During the training process, performance metrics such as accuracy, loss, etc. for both the training and validation sets are monitored. Initially, the performance on the validation set improves. However, as the model continues to train and after a certain number of epochs, the model starts to overfit the training data as it becomes increasingly better at fitting it, causing the performance on the validation set to degrade and the validation error to start to increase. Early stopping halts training at this point, preventing further overfitting and preserving the generalization ability of the model (Friedrich et al., 2023).

The basic algorithm of early stopping using a validation dataset works as follows (Prechelt et al., 2002):

1. Split the dataset into three sets: training, validation, and test sets.
2. Initialize the deep learning model and its hyperparameters. Then, start the training process on the training dataset only.
3. Monitor the performance of the predefined performance metric on the validation set regularly for instance, after each epoch. Examples of performance metrics to be evaluated are validation loss, accuracy, F1-score, etc.
4. The early stop criteria should be checked to decide when to stop training. For instance, a stop-training decision is made if the validation loss stops decreasing or starts increasing consistently over a certain number of epochs. For this step to be completed, the early stopping parameters must be set and checked, such as patience and threshold where patience refers to the number of epochs with no improvement after which training will be stopped, and the threshold is used to determine what constitutes an improvement whether the metric should increase or decrease.
5. Once the training stops, the best model is saved.

All utilized deep learning models in our study are set to halt the training process after five consecutive epochs in case the chosen monitored metric, validation loss, did not improve by at least 0.005 over the consecutive five epochs.

3.3.2.1.5 Dropout. Dropout is a simple yet effective technique introduced by Hinton et al. (2012). During the training process, the dropout method works by randomly setting a portion of the neurons or input units in the deep learning model to zero at each training iteration. This simply means effectively dropping out certain neurons during training to prevent the model from becoming heavily reliant on specific neurons, therefore, reducing the possibilities of overfitting.

According to Hinton et al. (2012), randomly dropping out of the deep learning model neurons leads to reduced training time of different networks of large size. The process of dropout works as follows (Srivastava et al., 2014):

1. During the training process, the dropout rate is initialized for the hidden units. The rate denotes the probability with which each neuron will be "dropped out" or deactivated.
2. At each training iteration, randomly deactivate a portion of neurons in accordance with the dropout rate and run the training on the remaining non-dropped-out neurons.
3. Since dropout essentially creates a new "thinned" network at each iteration, the weights are shared among these thinned networks to keep the parameter count reasonable.
4. Therefore, for each training iteration, a new "thinned" network is sampled and steps 2-3 are repeated.
5. During the testing process, the outgoing weights of each neuron are multiplied by the retention probability that was used during the training phase. This is to ensure that the expected output at test time matches that of the training time. The

full network is used for inference, without applying dropout, but with the scaled-down weights from the previous step.

In our study, the dropout technique has been used in all utilized deep learning models by different recurrent neural network layers, BiLSTM and BiGRU. In addition, it is also used by the dropout layer of the utilized convolutional neural network layer.

3.4 Experimented Models

This section provides an overview of the various experiments conducted in our study, as well as the deep learning architectures utilized for building diverse named entity recognition (NER) models. Initially in this section, we discuss the baseline experiments set by Cenikj et al. (2020) that we have replicated in our study. Then, the efficiency of domain adaptation, specifically from detecting food names within dish recipes to recognizing dish names within customer reviews is presented. Subsequent sections detail the training of the BiLSTM-CRF model using the benchmark *MenuNER* dataset, which aims to detect dish names using fixed pre-trained word embeddings. Later, the section discusses the incorporation of Part-of-Speech (POS) tags as part of the model input and explores the use of contextual word embeddings, specifically leveraging BERT, to enhance the models' capabilities further. Finally, we detail the specifics of deep learning models tested for the NER task across two distinct datasets: *MenuNER* and *ReviewsDB*.

3.4.1 Baseline Experiments: Detecting Food Names in FoodBase Dataset

To initiate our work, we replicated the experiments conducted by Cenikj et al. (2020) to set a standard for later comparisons. The original study by Cenikj et al. (2020) aimed to

identify food names in the *FoodBase* corpus using the method of stratified 5-fold cross-validation. For assessment, the macro average F1-score is used. This score is computed by averaging the F1 scores for each individual category over the 5-fold cross-validation process and then averaging those numbers for the three classes (I, O, B). It is worth noting that the dataset has an imbalance: the frequency of the O class is significantly higher than that of the B-Food and I-FOOD classes as earlier mentioned in Section 3.2.1 of the Thesis.

To validate the effectiveness of the method proposed by Cenikj et al. (2020), we meticulously replicated their experiments using the *FoodBase* corpus. While the original study employed 5-fold cross-validation, we limited our tests to a single fold and found that the results closely matched those of the 5-fold procedure. Based on this, we concluded that a single-fold evaluation is adequate for confirming the baseline findings.

We differed from the original study in one aspect: we did not use the authors' technique for handling Out of Vocabulary (OOV) words, instead representing all OOV words with zero vectors. Consequently, we compare our results with the version of the model presented by Cenikj et al. (2020) that also did not use a specific method for handling the OOV words. While the original authors employed the macro average F1-score for evaluation, our primary focus was on analyzing the precision and recall values for the three designated classes: B-FOOD, I-FOOD, and O.

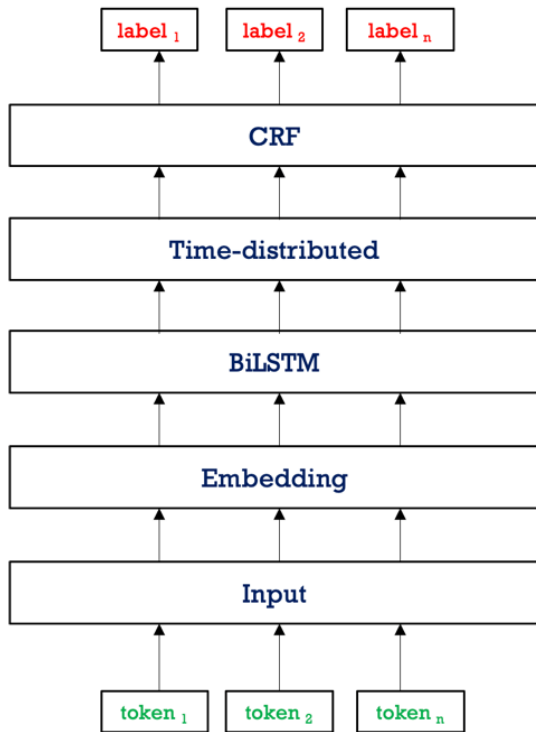
3.4.1.1 Baseline Deep Learning Model Architectures. The baseline model for our study is the BiLSTM-CRF architecture, a deep learning model that has been chosen for its relevance to our research objectives. To comprehensively assess its performance, we conducted experiments with varying inputs, initially following the approach of Cenikj et al. (2020).

For this reason, two distinct models based on their input configurations are considered:

1. **BiLSTM-CRF:** This model features a BiLSTM layer succeeded by a CRF layer. It uses word embedding vectors as inputs, which represent individual words in sentences along with their corresponding labels. This information aids the model in learning contextual relationships for making predictions.

2. **BiLSTM-CRF with CharEmbd:** This model builds on the BiLSTM-CRF architecture by including character embeddings as an extra input. The addition of character-level information is expected to enhance the model's ability to understand more complex structures and patterns within the data.

The different input configurations facilitate a comparative evaluation of the model's effectiveness across diverse settings. As illustrated in Figure 3-1, the architecture of the BiLSTM-CRF model consists of multiple layers, as follows:

Figure 3-1*BiLSTM-CRF Model Architecture*

1. The architecture begins with a single *input layer* that takes in word IDs from a defined vocabulary, which covers all word tokens in the *FoodBase* dataset.
2. These word IDs are then passed through an *embedding layer*, converting them into dense vector representations. One of the pre-trained models, such as Glove, Word2Vec, or FastText, is used for this transformation. The size of these vectors varies based on the specific pre-trained model chosen. It's noteworthy that in the original work by Cenikj et al. (2020), these word embeddings were considered non-trainable and remained fixed during the training. We retain this setting when replicating their model but later experiment with trainable embeddings.

An additional experiment introduces *lexical-300*, an embedding layer initialized randomly rather than with pre-trained embeddings. This layer is trainable, allowing it to learn custom weights that are better suited to the specific characteristics of the *FoodBase* dataset. This approach offers the potential for improved performance and a better understanding of the data.

3. The vectors from the *embedding layer* are subsequently sent to a *BiLSTM layer*, which has 50 hidden units and is used to capture the contextual information from the sequence in both forward and backward directions.

4. The *BiLSTM layer's* output is directed to a *fully connected layer with time-distributed* connections. This arrangement allows the model to effectively handle the sequential nature of the input data.

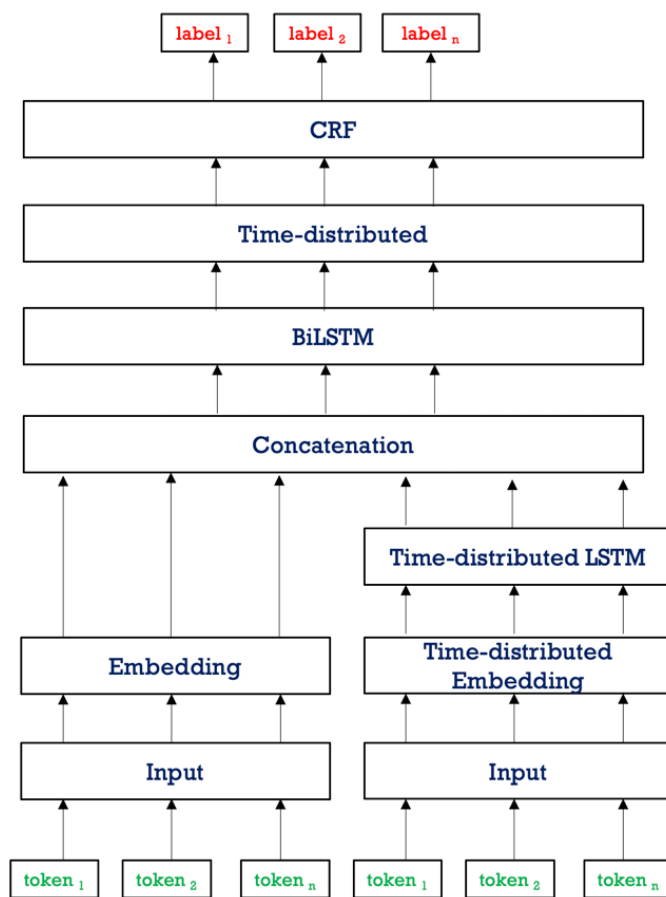
5. Lastly, a Conditional Random Field (CRF) layer is employed for sequence labeling to predict the named entity tags (B-Food, I-FOOD, or O). This layer models the relationship between adjacent labels in the sequence, providing a probability distribution over all possible label sequences. The CRF layer acts on the output of the time-distributed dense layer. The model is compiled using the RMSprop optimizer and the CRF loss function, with the accuracy of the CRF predictions also monitored as a training metric.

This design integrates the advantages of both the BiLSTM and CRF layers. The BiLSTM layer captures contextual information in both directions of the sequence, while the *CRF* layer models the dependencies between adjacent labels. These features synergize to improve the model's precision in predicting class labels.

In contrast, the BiLSTM-CRF with CharEmbd model extends the basic BiLSTM-CRF architecture by incorporating character embeddings as an additional input, as depicted in Figure 3-2. This enhancement is intended to provide a more comprehensive understanding of the data, including its finer-grained structures and patterns.

Figure 3-2

BiLSTM-CRF With CharEmbd Model Architecture



The character embeddings in the BiLSTM-CRF with CharEmbd model are generated through a specialized neural network architecture with the following components:

1. *Input layer* for characters: An input layer is designed to manage character-level data. It accommodates characters based on the maximum sentence length (50) and the calculated maximum word length, thus allowing the model to process characters in the context of individual words.
2. *Character embedding layer*: A *TimeDistributed* wrapper is employed around an embedding layer that maps input characters to dense vectors of a fixed size, which is set at 10 dimensions. This allows the same embedding operation to be independently applied to each word at every time step. The embedding layer starts with random weights and is fine-tuned during training using rmsprop optimizer. Over several epochs, the learned embeddings encapsulate the semantic and contextual information of the characters.
3. *Character-level LSTM layer*: This layer processes the sequence of character embeddings for each word, generating a single output vector per word. Recurrent dropout is used, randomly omitting 50% of the units during training to reduce the risk of overfitting. This setup enables the model to capture character-level features and generate word encodings based on these features.
4. *Concatenation layer*: The outputs from the *character-level LSTM* and the *word embedding* layer are combined using a *concatenate layer*. This forms the comprehensive input that feeds into the main BiLSTM-CRF architecture.

In summary, both the basic BiLSTM-CRF and the extended BiLSTM-CRF with CharEmbd models share a core structure: an embedding layer, a bidirectional LSTM (BiLSTM) layer, and a Conditional Random Field (CRF) layer. These components

collectively equip the models to effectively perform sequence labeling tasks, specifically in the realm of named entity recognition, enabling accurate predictions of entity tags within a word sequence.

3.4.1.2 Experimental Settings. In this study, all experiments were conducted under standardized conditions to ensure consistent and comparable results. The batch size is fixed at 256, and the maximum sentence length is limited to 50 tokens. For the sake of reproducibility, all sources of randomness are eliminated by setting both random and hash seeds to a seed value of 42. This ensures that running the same model multiple times produces identical results.

The *lexical-300* model employed custom-designed word embeddings specifically trained with a dimensionality of 300. This dimensionality is chosen to align with standard word representation models. In contrast, the character-based representations were designed with a lower dimensionality of 20, reflecting the limited complexity of character sets.

Moreover, to prepare the data for model input, each word in the *FoodBase* dataset is assigned a unique numerical ID. To maintain consistent sequence lengths, sentences were padded to a total of 50 tokens, aligning with the dataset's maximum sentence length of 45 tokens.

For character-level representations, the unique characters are identified in the dataset and the words are converted into sequences of character IDs. These sequences are then padded to 20 tokens based on the longest word in the dataset, which consisted of 18 characters, along with the two special tokens: PAD and UNK.

On the other hand, the architecture of the models included *fully connected time-distributed* and *LSTM* layers, both with 50 hidden units. The dropout rate in the *BiLSTM* layer was set at 10%. The activation function in the *time-distributed* layer was the rectified linear unit, *relu*, and the *rmsprop* optimizer is used with a learning rate of 0.001. During training, the validation loss is monitored and if no improvement of at least 0.005 is observed over five consecutive epochs, the training is terminated to prevent overfitting. Additionally, 10% of the training data is allocated for validation to monitor the model's performance during training.

3.4.1.3 Word Embeddings. During this research, seven distinct variations of word embeddings are explored, each is serving as inputs to various deep-learning model architectures. To facilitate a comprehensive understanding of these embeddings, a summarized overview of their key characteristics is presented in Table 3-6 (RaRe-Technologies, 2018).

Table 3-6

Word Embedding Models Used in Experiments

No.	Embeddings Name	Pre-Trained Word Embeddings	Dimension
1	glove_twitter-25	Glove twitter embeddings were generated	25
2	glove_twitter-100	based on 2B tweets, with vocabulary size:	100
3	glove_twitter-200	1.2M words (uncased).	200
4	glove_wiki-50	Glove wiki embeddings were generated based	50
5	glove_wiki-300	on Wikipedia 2014, and Gigaword, with vocabulary size: 400K words (uncased).	300
6	word2vec-300	Word2vec embeddings were generated based	300

No.	Embeddings Name	Pre-Trained Word Embeddings	Dimension
		on Google News dataset, with vocabulary size: 3M words and phrases.	
7	fasttext-wiki-news-300	Fasttext embeddings were generated based on Wikipedia 2017, UMBC webbase corpus, and statmt.org news dataset.	300

Note: The table demonstrates the used word embeddings detailing the text sources used to generate these embeddings, the vocabulary size and the vector dimensions. Adapted from “gensim-data”, by RaRe-Technologies, 2018, <https://github.com/RaRe-Technologies/gensim-data>. Copyright 2018 by the authors.

It is important to highlight that the datasets utilized for training these language models were of considerable amount, which reduced the probability of significant bias impacts. Additionally, the texts incorporated within these datasets covered a wide range of topics, ensuring diversity in the training data.

Furthermore, it is important to note that the focus is solely on extracting and examining word embeddings for words that appeared within the *FoodBase* dataset. This approach allowed us to specifically analyze the relevance and performance of the embeddings in the context of our dataset.

3.4.2 Domain Adaptation from Food Names Detection in FoodBase to Dish Name Detection in Customer Reviews

In the study's second experiment, the performance of the two trained baseline models: BiLSTM-CRF and BiLSTM-CRF with CharEmbd are tested for their ability to identify dish names within manually annotated reviews.

Details about the dataset utilized for testing the models can be found in Section 3.2.1 of the thesis. It is crucial to highlight that the configurations, including pre-trained word embeddings, deep learning model architectures, and experimental settings, were maintained consistently as outlined in the previous section where the baseline models were trained and assessed on the *FoodBase* dataset.

3.4.3 Training the BiLSTM Models on MenuNER Dataset Using Fixed Pre-Trained Word Embeddings

In the third phase of our experimental work, the focus is on the *MenuNER* dataset, which is specifically annotated and provided by Syed and Chung (2021). The decision to use this dataset for testing the baseline models: BiLSTM-CRF and BiLSTM-CRF with CharEmbd is motivated by the importance of benchmarking our results against those of Syed and Chung (2021). This strategy enables a direct and comprehensive comparison of our methodology with their existing work.

An in-depth description of the *MenuNER* dataset can be found in Section 3.2.3 of the thesis. It's worth noting that the pre-trained word embeddings used in this phase are consistent with those utilized in previous experiments, where these embeddings were set as fixed

parameters. Additionally, the architecture of the deep learning models and the experimental settings are maintained as per the baseline models discussed in earlier sections.

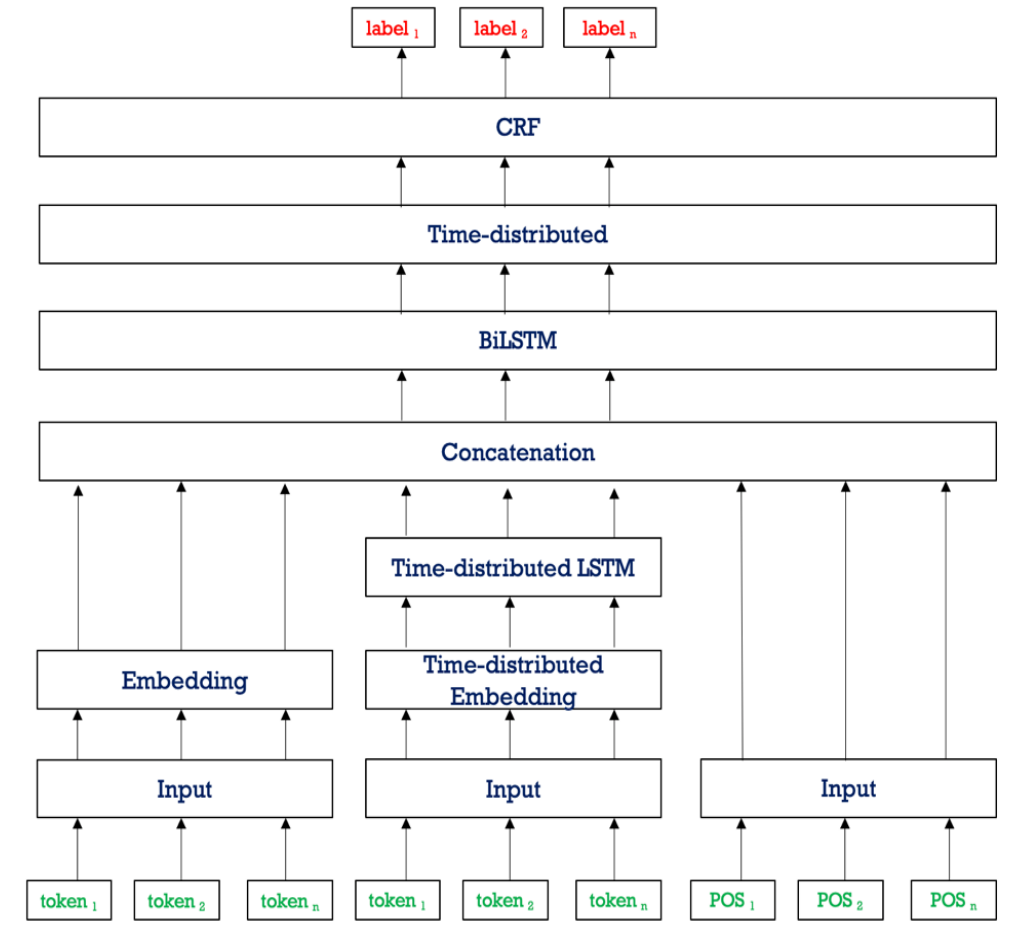
3.4.4 Enhancing BiLSTM-CRF Model Performance by Incorporating the POS Tags and Trainable Word Embeddings

In the fourth stage of our experimental series, we augment the baseline model BiLSTM-CRF with CharEmbd by incorporating Part-of-Speech (POS) tags, represented through one-hot encoding. As a result, the model's input now comprises not only word and character embeddings but also POS and syntactic features. To integrate these POS features into the deep learning model, the 'Chunk Tag' column from the *MenuNER* Dataset is transformed into a one-hot encoded format. This column includes a range of POS tags and other syntactic categories that serve as additional features for the model.

An exhaustive list of these POS tags, with their detailed explanations, is provided in Appendix B of the thesis. Figure 3-3 visually illustrates how the BiLSTM-CRF with CharEmbd model has been adapted to include these POS-based inputs.

Figure 3-3

BiLSTM-CRF With CharEmbd Model Utilizing POS Features as Additional Input



3.4.5 The Utilization of Contextual Word Embeddings (Bert)

In the fifth stage of our experimental series, we explore the impact of incorporating contextual word embeddings, specifically those generated by BERT, on the performance of our baseline models. Two primary considerations arose when preparing to integrate BERT embeddings into the *MenuNER* dataset for our study:

1. Identifying the most effective methodology for generating the embeddings.
2. Choosing the specific version of the BERT model best suited for generating these embeddings.

To address the first issue, we experimented with three distinct strategies for generating BERT embeddings, aiming to determine which would most effectively enhance the performance of our baseline model, the BiLSTM-CRF with CharEmbd. We examined the following approaches:

- **Mean Pooling:** This technique calculates the average output of the hidden units from the BERT model's last four layers, resulting in embeddings with 768 dimensions.
- **Summation:** This approach sums the outputs from the hidden units of the BERT model's last four layers, also producing 768-dimensional embeddings.
- **Concatenation:** This method concatenates the outputs from the hidden units of the last four layers of the BERT model, generating embeddings with a dimensionality of 3,072.

Upon reviewing the experiment's outcomes, we found that the mean pooling strategy slightly outperformed the other methods in terms of the F1 score for the resultant model.

As indicated in Table 3-7, the Bert-MeanPooling approach emerged as the most effective, delivering the highest F1 score when integrated with the BiLSTM-CRF with

CharEmbd model. Given this promising performance, we chose to continue utilizing the mean pooling methodology for generating BERT embeddings.

Table 3-7

BiLSTM-CRF with CharEmbd Model Performance When Using BERT Embeddings.

Bert Embeddings	CharEmbd-BiLSTM-CRF		
	P	R	F1
Bert-MeanPooling	0.91	0.83	0.86
Bert-SumLast4	0.91	0.82	0.85
Bert-ConcatLast4	0.90	0.81	0.85

It's noteworthy that our chosen approach aligns with the methodology employed in the Syed and Chung (2021) study, where BERT embeddings were also generated using mean pooling. This methodological consistency enhances the comparability of our findings with those reported in their work.

Addressing the second concern about selecting the appropriate version of the BERT model, we experimented with two variations: *bert-base-cased* and *bert-base-uncased*. Beyond these, we also examined a hybrid approach that combined both versions. The hybrid configurations are termed *bert-cased-uncased-mean-768* and *bert-cased-uncased-concat-1536*, as detailed in Table 3-8.

Table 3-8*Bert Embeddings Variants*

Bert Version	Embedding Dimension	Description
bert-cased-mean-768	768	bert-base-cased is used to generate the embeddings using mean pooling strategy
bert-uncased-mean-768	768	bert-base-uncased is used to generate the embeddings using mean pooling strategy
bert-cased-uncased-mean-768	768	Combines the generated bert-base-cased and bert-base-uncased embeddings by averaging the embeddings. In this approach, the embeddings of a word in the cased representations are averaged with the embeddings of the uncased representation, resulting in the same number of embeddings, 768. For instance, the embeddings of the words “Pasta”, and PASTA” are averaged with the embeddings of the word “pasta”. This will lead to the same number of embeddings, 768.
bert-cased-uncased-concat-1536	1,536	Combines the generated bert-cased and bert-uncased embeddings by concatenating the embeddings. Here, the embeddings of a word in

Bert Version	Embedding Dimension	Description
		the cased representations are concatenated with the embeddings of the uncased representation, leading to a total dimension of 1536 embeddings for each word in the dataset. For instance, the embeddings of the words “Pasta”, and “PASTA” are concatenated with the embeddings of the word “pasta” each. This will lead to a total dimension of 1536 embeddings for each word in the dataset.

The rationale behind combining embeddings from both *bert-base-cased* and *bert-base-uncased* versions stems from empirical observations during our model testing. It is observed that each version of BERT had a distinct impact on key performance indicators. Specifically, the use of cased embeddings led to an increase in the model's precision, while the uncased embeddings improved the model's recall rate. As a result, integrating both cased and uncased embeddings is expected to concurrently boost both precision and recall, thereby enhancing the model's overall F1 score.

In summary, four distinct BERT embedding variations are explored, along with other pre-trained models such as GloVe, Word2Vec, and FastText. Additionally, the *lexical-300* model is deployed using custom-designed word embeddings to provide a comprehensive evaluation of model performance as illustrated in Table 3-9.

Table 3-9*All Utilized Word Embeddings in the Study*

No.	Embedding Type	Embeddings Name	Dimension
1	Glove	glove_twitter-25	25
2		glove_twitter-100	100
3		glove_twitter-200	200
4		glove_wiki-50	50
5		glove_wiki-300	300
6	Word2Vec	word2vec-300	300
7	FastText	fasttext-wiki-news-300	300
8	Bert	bert-cased-mean-768	768
9		bert-uncased-mean-768	768
10		bert-cased-uncased-mean-768	768
11		bert-cased-uncased-concat-1536	1536
12	Custom embeddings	Lexical-300	300

3.4.6 Experiment with Different Deep Learning Model Architectures

In the sixth phase of our experimental series, the focus is on identifying the best-suited deep learning and word embedding models for dish name detection in customer reviews. A variety of modifications are made to our baseline model, BiLSTM-CRF with CharEmbd, to assess their impact on performance. The key aspects of this phase include:

- **Input consistency:** all experiments maintain a standardized input configuration comprising word embeddings, character embeddings, and Part-of-Speech (POS) features to ensure comparability across different model evaluations.
- **Embedding fine-tuning:** the pre-trained word embeddings are incorporated into the named entity recognition (NER) models and further fine-tuned during training. These embeddings serve as the initialization for the word embedding layer.
- **Embedding diversity:** a total of 12 different word embeddings are explored to rigorously assess the model's adaptability to various linguistic contexts. The 12 embeddings are: *glove-twitter-25*, *glove-twitter-100*, *glove-twitter-200*, *glove-wiki-gigaword-50*, *glove-wiki-gigaword-300*, *word2vec-300*, *fastText-300*, *lexical-300*, *bert-cased-mean-768*, *Bert-uncased-mean-768*, *bert-cased-uncased-mean-768*, and *bert-cased-uncased-conct-1536*.
- **Character embedding consistency:** the methodology for generating character embeddings remains consistent across all models, as outlined earlier in Section 3.4.1.1.
- **Recurrent layer variants:** beyond the BiLSTM layer, it is also experimented with a BiGRU layer and a Convolutional Neural Network (CNN) layer to assess their potential impact on NER model performance.
- **Hidden unit dimensions:** the effect of altering the dimensions of the hidden units across different layers is examined, aiming to find the optimal configuration for enhanced performance.

- **Optimizer Variation:** Two distinct optimizers, rmsprop and Adam, are tested to evaluate their influence on the model's training and performance metrics. This comparison helps clarify how the choice of optimizer can affect overall model efficacy.

This section is divided into three main parts, each focusing on introducing a specific deep-learning architecture proposed for the NER task. First, the introduction of deep learning models utilizing one type of recurrent layers is given. Then, an exploration of deep learning models utilizing different recurrent layers is presented before concluding the section by investigating the deep learning models utilizing both recurrent and convolutional neural network layers.

3.4.6.1 Deep Learning Models Using One Type of Recurrent Layers. The two primary recurrent neural networks employed in this study are the BiLSTM and BiGRU, configured in various ways and their specific implementations are as follows:

- **BiLSTM-CRF Models and 2BiLSTM-CRF Models**

These models are depicted in Figure 3-4 and Figure 3-5. The baseline BiLSTM-CRF model serves as our point of comparison and has been extended into seven different configurations. The variations mainly involve adjusting the number of neurons in the BiLSTM layer, ranging from 25 to 150 neurons and the use of different optimizers, rmsprop and Adam. The 2BiLSTM-CRF Models employ a stack of two BiLSTM layers and come in five configurations with differing numbers of neurons in the BiLSTM layers, namely 25, 50, 75, and 100 and two utilized optimizers.

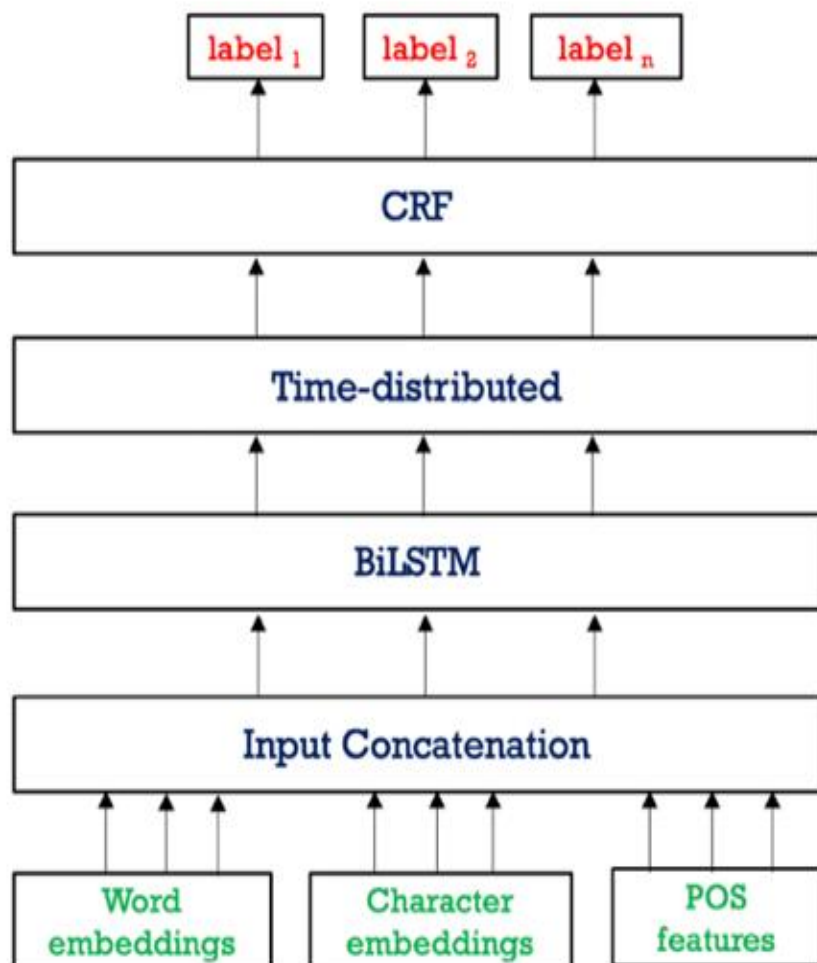
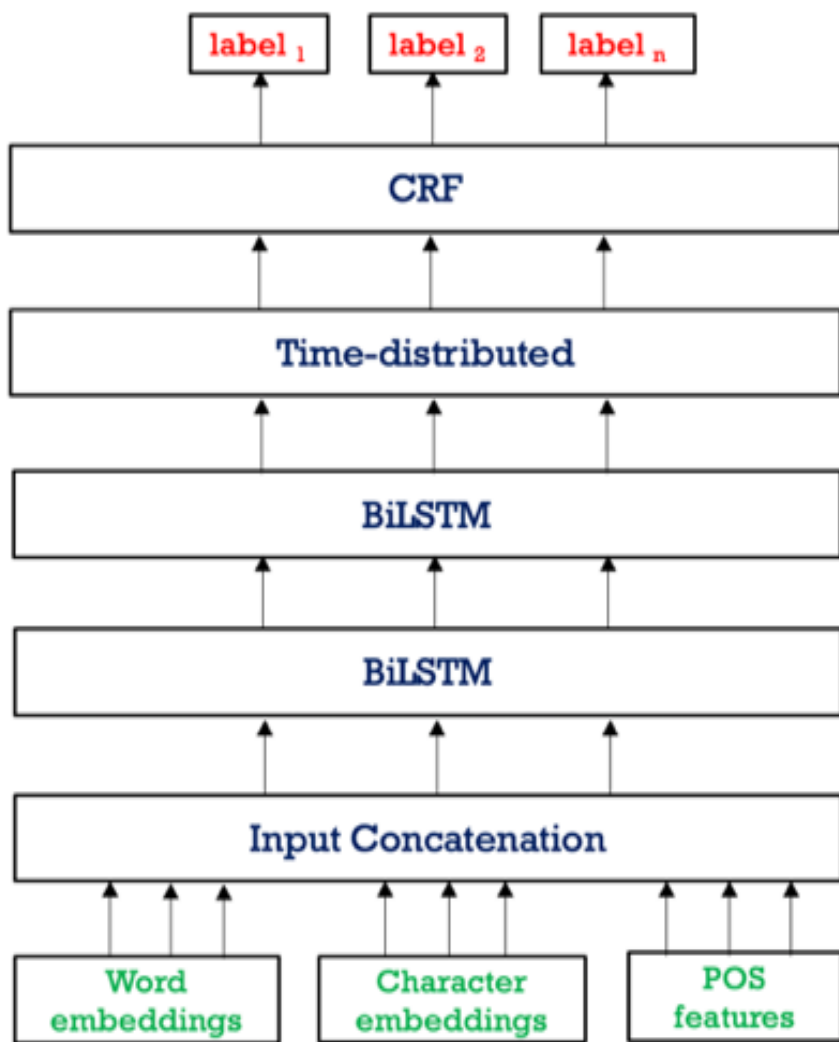
Figure 3-4*1BiLSTM-CRF Model Architecture*

Figure 3-5*2BiLSTM-CRF Model Architecture*

- **BiGRU-CRF Models and 2BiGRU-CRF Models**

These models are illustrated in Figure 3-6 and Figure 3-7. The BiGRU-CRF model has seven distinct configurations. As with the BiLSTM models, the number of neurons in the BiGRU layer varies from 25 to 150. All variants are optimized using Adam, except BiGRU(50N)-CRF which employs once with Adam and another time with rmsprop.

For the 2BiGRU-CRF models, the setup involves stacking two BiGRU layers and comes in five variations based on the number of neurons: 25, 50, 75, and 100 in these layers. Similar to the 2BiLSTM-CRF models, all are optimized using Adam except for one model, 2BiGRU(50N)-CRF, which also is experimented with rmsprop optimizer.

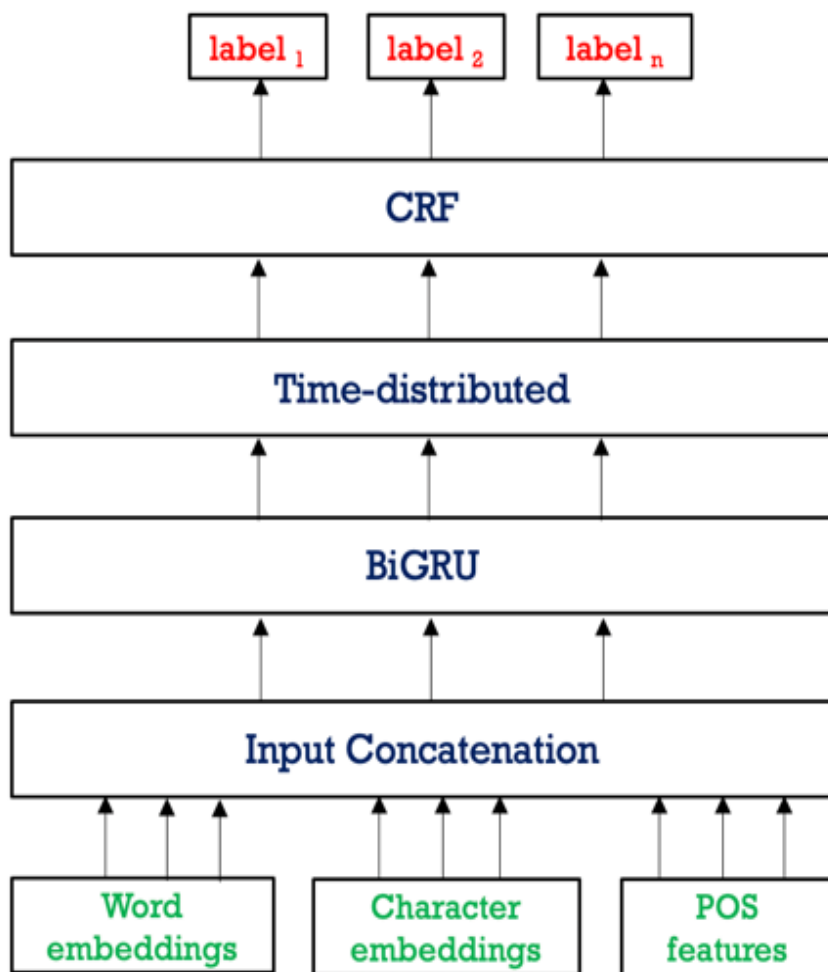
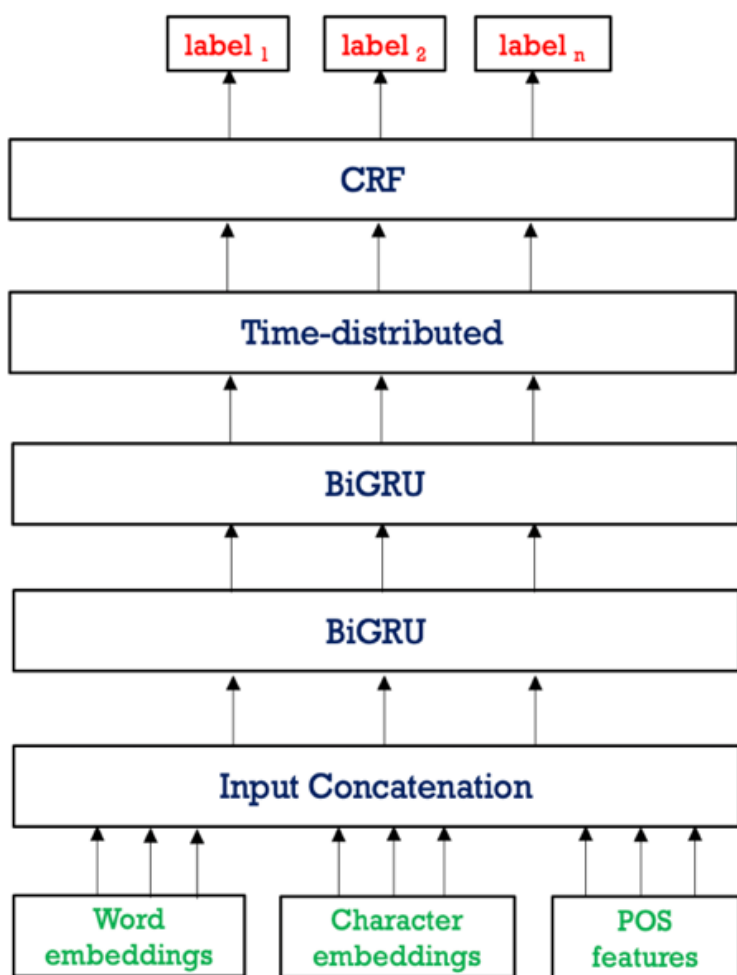
Figure 3-6*1BiGRU-CRF Model Architecture*

Figure 3-7*2BiGRU-CRF Model Architecture*

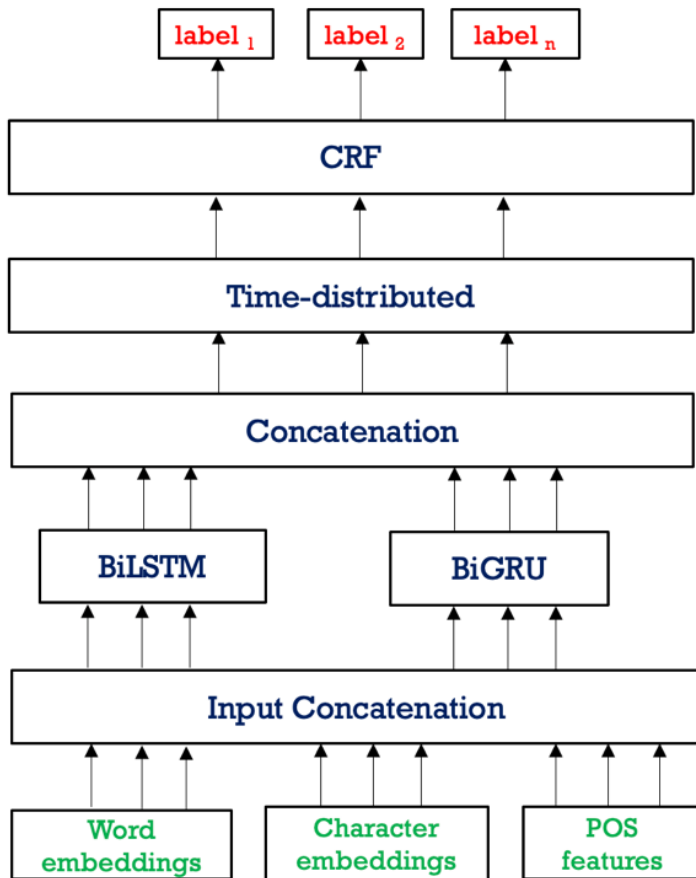
In summary, both BiLSTM and BiGRU models are subjected to extensive experimentation involving various configurations. This allows for a detailed understanding of how changes in architecture and optimization algorithms affect performance.

3.4.6.2 Deep Learning Models Using Recurrent Layers. There are two models that utilize the two types of recurrent layers: BiLSTM and BiGRU and they are:

- **1BiLSTM-1BiGRU-CRF**

As illustrated in Figure 3.8, the input to the hybrid BiLSTM-BiGRU-CRF model navigates sequentially through a BiLSTM layer and a BiGRU layer. The outputs from these two layers are then concatenated and funneled into a time-distributed layer. Finally, a Conditional Random Field (CRF) layer is deployed for sequence labeling.

This composite model exists in five distinct configurations, each differing in terms of the optimizer employed and the number of neurons in the BiLSTM and BiGRU layers. These variations enable a thorough exploration of how these settings impact the model's overall performance.

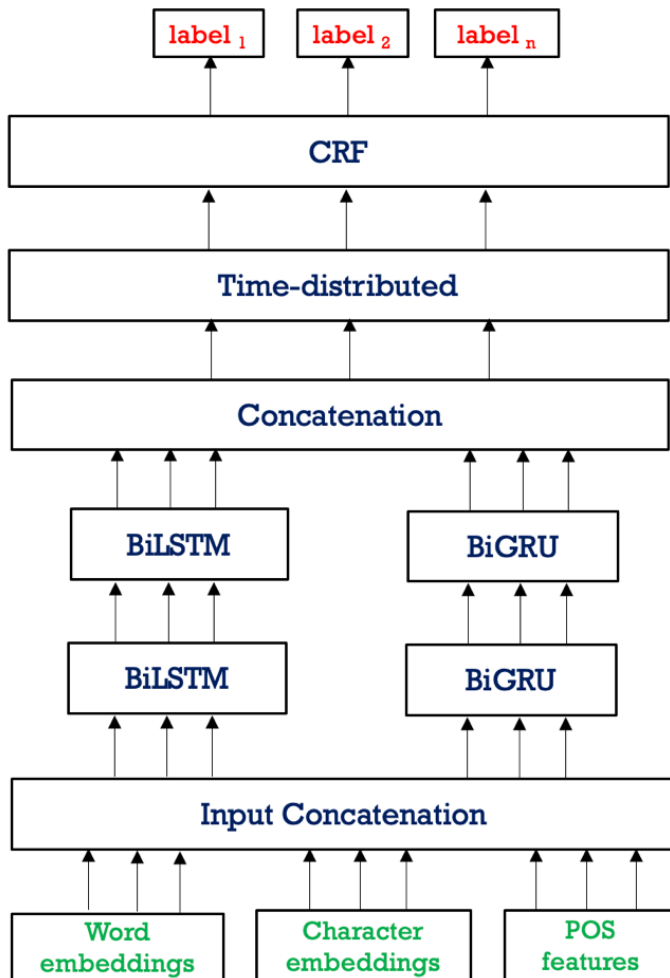
Figure 3-8*1BiLSTM-1BiGRU-CRF Model Architecture*

- **2BiLSTM-2BiGRU-CRF**

As depicted in Figure 3.9, this advanced model employs dual-layered stacks of both BiLSTM and BiGRU networks. Initially, the input sequence is processed through the first layers of both the BiLSTM and BiGRU stacks. The resulting outputs are then used as inputs for the second layers in each stack. Subsequently, the outputs from these second layers are merged using a concatenation layer. This unified sequence then flows through a time-distributed layer before reaching the final layer, the Conditional

Random Field (CRF), which performs the ultimate task of Named Entity Recognition (NER) by assigning appropriate labels to each token in the sequence.

The model is designed in five unique configurations, varying in terms of the choice of optimizer and the neuron count in both the BiLSTM and BiGRU layers. These different setups facilitate an in-depth investigation into how these specific parameters influence the overall effectiveness of the model.

Figure 3-9*2BiLSTM-1BiGRU-CRF Model Architecture*

3.4.6.3 Deep Learning Models Using Recurrent Layers and Convolutional Neural

Network Layers. The third set of models are models that combine recurrent layers and utilizing convolutional neural network layer, as follows:

- **1BiLSTM-1CNN-CRF, 1BiGRU-1CNN-CRF, and 1BiLSTM+1BiGRU-1CNN-CRF models**

As depicted in Figures 3-10, 3-11, and 3-12, our models incorporate a convolutional neural network (CNN) layer in various configurations: first, in combination with a BiLSTM layer; second, with a BiGRU layer; and finally, as part of a multi-layer consisting of BiLSTM, BiGRU, and CNN layers. Each of these architectures is further refined into up to five distinct variants, distinguished by the number of neurons in each layer and the optimizer algorithms employed.

The CNN layer serves to distill the input sequence into a compact feature vector through the following sequence of operations:

1. **Convolutional Layer:** Equipped with 100 filters (or kernels), this layer utilizes a kernel size of 3 and employs the ReLU (Rectified Linear Unit) activation function to introduce non-linearity into the network.
2. **Max Pooling Layer:** This layer employs a pool size of 2 to perform max pooling, effectively reducing the sequence length by half and retaining the most critical information.
3. **Dropout Layer:** To mitigate the risk of overfitting, a dropout layer is added that randomly deactivates 20% of its input units during the training phase.

This CNN segment undergoes a sequence of transformative steps, including filtering, down-sampling, and dropout operations. The resulting feature vector is then fused with the outputs of either the BiLSTM or BiGRU layers, or both, in the configurations. Such a multi-layer, multi-model approach provides a robust mechanism for capturing both the local and global features inherent in the input sequence.

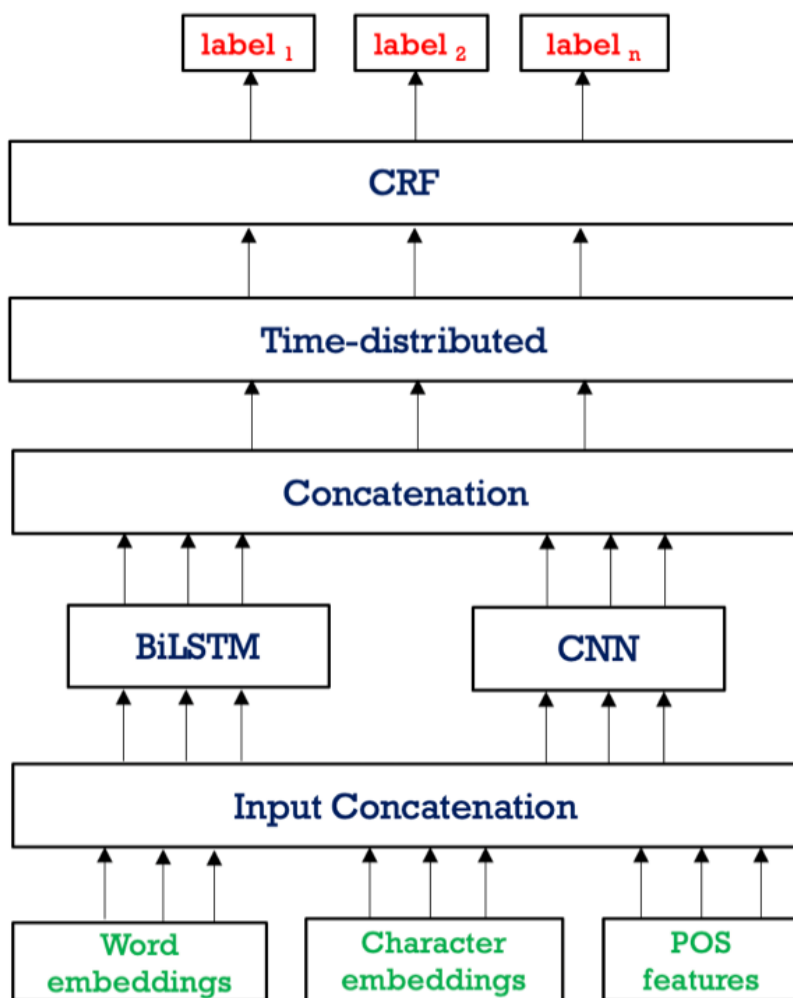
Figure 3-10*1BiLSTM-1CNN-CRF Model Architecture*

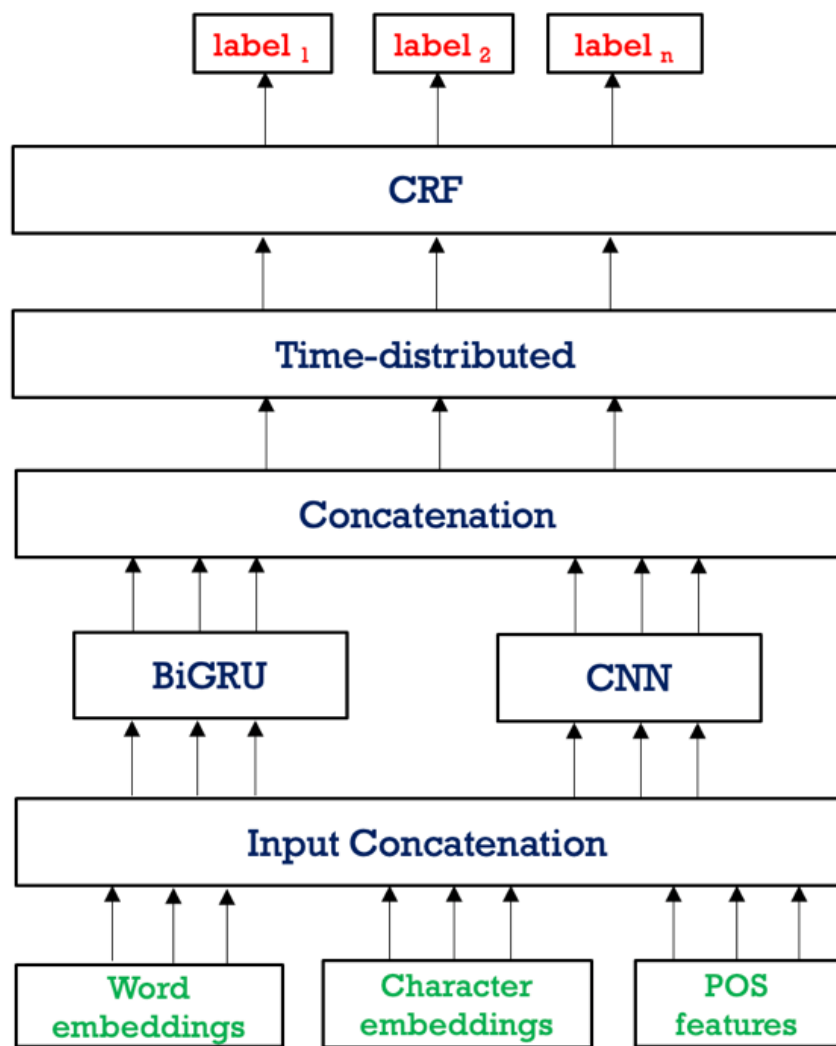
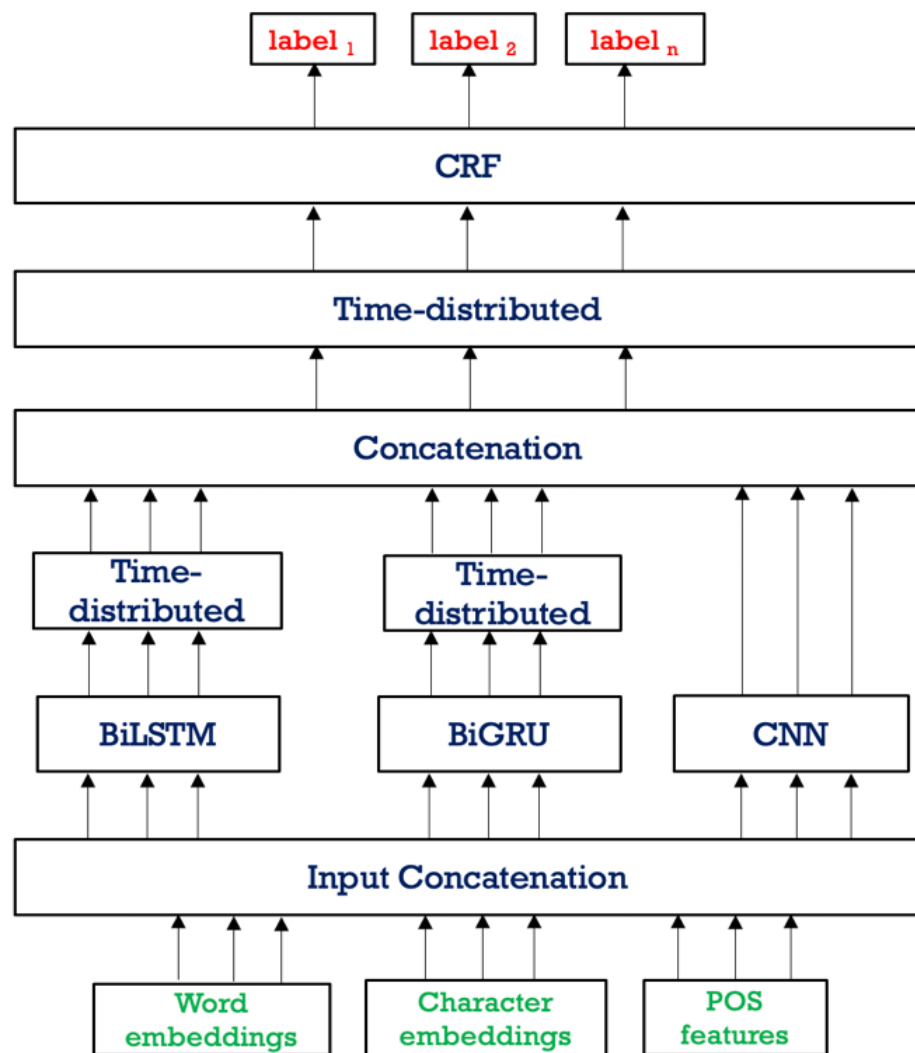
Figure 3-11*1BiGRU-1CNN-CRF Model Architecture*

Figure 3-12

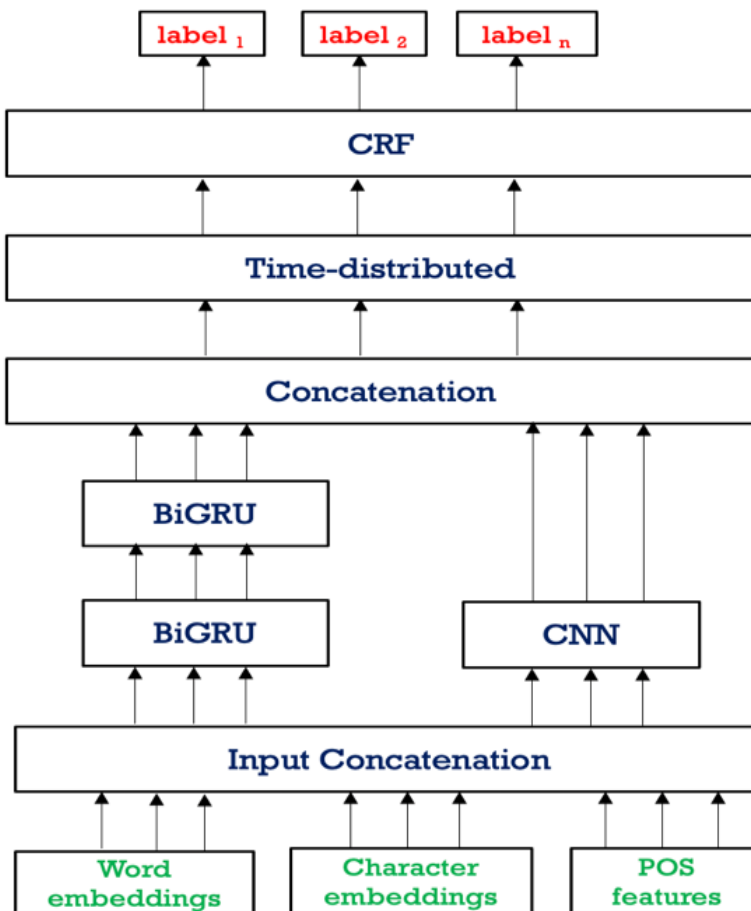
1BiLSTM-1BiGRU-1CNN-CRF Model Architecture

- **2BiGRU-1CNN-CRF models**

As the name suggests, the 2BiGRU-1CNN-CRF model incorporates a stacked architecture featuring two BiGRU layers alongside a CNN layer. As depicted in Figure 3-13, these layers operate in parallel, each processing the model's input independently. Their respective outputs are then fused together through a concatenation operation, forming a unified feature representation.

This concatenated output serves as the input to a subsequent time-distributed layer, which applies the same transformation at every time step of the sequence. Finally, a Conditional Random Field (CRF) layer serves as the terminus of the model, responsible for sequence labeling tasks.

This architecture thus leverages the strengths of both recurrent and convolutional layers to capture complex features and dependencies, thereby providing a robust solution for sequence labeling problems.

Figure 3-13*2BiGRU-1CNN-CRF Model Architecture*

It's crucial to emphasize that the configuration parameters for the convolutional layer, such as the number of filters and dropout rates, were not arbitrarily selected. Instead, they were determined through a series of experimental trials to evaluate their impact on the Named Entity Recognition (NER) model's performance metrics. By experimenting with various settings, the aim is to identify the optimal parameters that would yield the best results.

Table 3-10 presents a representative example of these experiments, specifically focusing on a 2BiGRU(75)-CNN-CRF model which is evaluated on one fold of data. Through this iterative process, we were able to fine-tune the model's architecture, ultimately achieving a configuration that effectively balances accuracy and computational efficiency.

Table 3-10

Experiment with Different Parameter Values of the CNN Model

2BiGRU(75)-CNN-CRF		Evaluation Metrics		
No. of filters	Dropout	P	R	F1
64	no dropout	93.0	93.3	93.0
100	0.2	94.0	95.3	94.7
100	0.3	93.7	93.0	93.3
200	0.2	94.0	92.3	93.0
200	0.3	93.3	91.3	92.3

As shown in Table 3-10, the best F1 score, 94.7, was obtained when the CNN layer was configured with 100 filters and a dropout value of 0.3. Therefore, in all our experiments we used these parameter values with the CNN layer utilized in the deep learning models.

On the other hand, Table 3-11, summarizes all the experimented NER deep learning models in our study for detecting dish names in customer reviews datasets.

Table 3-11

All Experimented Deep Learning Models in Our Study.

Model Architectures	
1BiLSTM-CRF	BiLSTM(50N)-CRF (rmsprop) (non-Trainable)
	BiLSTM(50N)-CRF (rmsprop)
	BiLSTM(25N)-CRF (Adam)
	BiLSTM(50N)-CRF (Adam)
	BiLSTM(75N)-CRF (Adam)
	BiLSTM(100N)-CRF (Adam)
	BiLSTM(125N)-CRF (Adam)
	BiLSTM(150N)-CRF (Adam)
1BiGRU-CRF	BiGRU(50N)-CRF (rmsprop) (non-Trainable)
	BiGRU(50N)-CRF (rmsprop)
	BiGRU(25N)-CRF (Adam)
	BiGRU(50N)-CRF (Adam)
	BiGRU(75N)-CRF (Adam)
	BiGRU(100N)-CRF (Adam)
	BiGRU(125N)-CRF (Adam)
	BiGRU(150N)-CRF (Adam)
2BiLSTM-CRF	2BiLSTM(50N)-CRF (rmsprop)
	2BiLSTM(25N)-CRF (Adam)
	2BiLSTM(50N)-CRF (Adam)

Model Architectures	
2BiGRU-CRF	2BiLSTM(75N)-CRF (Adam)
	2BiLSTM(100N)-CRF (Adam)
	2BiGRU(50N)-CRF (rmsprop)
	2BiGRU(25N)-CRF (Adam)
	2BiGRU(50N)-CRF (Adam)
1BiLSTM-1BiGRU-CRF	2BiGRU(75N)-CRF (Adam)
	2BiGRU(100N)-CRF (Adam)
	1BiLSTM(50N)-1BiGRU(50N)-CRF (rmsprop)
	1BiLSTM(25N)-1BiGRU(25N)-CRF (Adam)
	1BiLSTM(50N)-1BiGRU(50N)-CRF (Adam)
2BiLSTM-2BiGRU-CRF	1BiLSTM(75N)-1BiGRU(75N)-CRF (Adam)
	1BiLSTM(100N)-1BiGRU(100N)-CRF (Adam)
	2BiLSTM(50N)-2BiGRU(50N)-CRF (rmsprop)
	2BiLSTM(25N)-2BiGRU(25N)-CRF (Adam)
	2BiLSTM(50N)-2BiGRU(50N)-CRF (Adam)
1BiLSTM-1CNN-CRF	2BiLSTM(75N)-2BiGRU(75N)-CRF (Adam)
	2BiLSTM(100N)-2BiGRU(100N)-CRF (Adam)
	1BiLSTM(25N)-1CNN-CRF (Adam)
	1BiLSTM(50N)-1CNN-CRF (Adam)
	1BiLSTM(75N)-1CNN-CRF (Adam)
	1BiLSTM(100N)-1CNN-CRF (Adam)

Model Architectures	
1BiGRU-1CNN-CRF	1BiGRU(25N)-1CNN-CRF (Adam)
	1BiGRU(50N)-1CNN-CRF (Adam)
	1BiGRU(75N)-1CNN-CRF (Adam)
	1BiGRU(100N)-1CNN-CRF (Adam)
1BiLSTM-1BiGRU-1CNN-CRF	1BiLSTM(25N)-1BiGRU(25N)-1CNN-CRF (Adam)
	1BiLSTM(50N)-1BiGRU(50N)-1CNN-CRF (Adam)
	1BiLSTM(75N)-1BiGRU(75N)-1CNN-CRF (Adam)
2BiGRU-1CNN-CRF	1BiLSTM(100N)-1BiGRU(100N)-1CNN-CRF (Adam)
	2BiGRU(75N)-1CNN-CRF (rmsprop)
	2BiGRU(25N)-1CNN-CRF (Adam)
	2BiGRU(50N)-1CNN-CRF (Adam)
	2BiGRU(75N)-1CNN-CRF (Adam)
	2BiGRU(100N)-1CNN-CRF (Adam)

3.4.7 Experiment Settings

In the process of evaluating deep learning models for the named entity recognition task, systematic experiments were undertaken. These were subjected to diverse conditions and hyperparameters. Notably, there were inconsistencies in the computational environments including variations in processor capabilities (both CPUs and GPUs) and available RAM resources. These inconsistencies were even evident when the same model was tested on different folds of the same dataset.

For some experiments, a dedicated server was employed, characterized by the following specifications:

- **Operating System:** Microsoft Windows Server 2016
- **Processor:** Intel Xeon CPU, 3.20 GHz
- **Memory:** 32 GB RAM
- **System Architecture:** 64-bit

In contrast, other tests were facilitated using the Google Colab Pro+ platform, which provisioned the experiments with:

- **Memory:** 50 GB RAM
- **Processor:** GPU variants, specifically V100, A100, and T4.

Given the heterogeneous nature of the test environments, there were observable differences in model convergence times: some models converged in few minutes, whereas others necessitated hours. Due to this variability, a direct comparison of training durations among models is not practicable. As a more robust comparative measure, the quantity of trainable parameters for each model was documented. This offers a solid foundation for assessing the anticipated computational resources needed for training these models

Chapter 4 : Experimental Results and Analysis

Chapter 4 presents the outcomes of the experiments as outlined in Section 3.4. Furthermore, each section in Chapter 4 offers a comprehensive analysis and discussion of the obtained results. After analyzing the results, a comparative analysis is presented to compare the performance of our proposed NER deep learning models and the performance of the proposed models by Syed and Chung (2021). At the end, the results of experimenting our models on the *ReviewsDB* dataset and the comparison between the results of these experiments and the experiments conducted on *MenuNER* dataset are detailed.

4.1 Analysis of the BiLSTM-CRF Model

This section documents the experiments we replicated based on the proposed baseline model by Cenikj et al. (2020). Also, we present our results of exploring the potential effectiveness of using domain adaptation between detecting food names in food recipes and detecting dish names in customer reviews on NER model performance. Additionally, the analysis of the obtained results of the BiLSTM-CRF NER model when using fixed pre-trained word embeddings, part of speech (POS) features and the utilization of Bert embeddings are detailed.

4.1.1 Results and Analysis of Replicated Baseline Experiments by Cenikj et al. (2020)

In accordance with the experiments conducted by Cenikj et al. (2020), our study involves the replication of the training process of the BiLSTM-CRF model on the *FoodBase* dataset. This replication is performed in two scenarios: one where character embeddings are not included as input to the model, and the other where character embeddings are

incorporated, referred to as "BiLSTM-CRF with CharEmbd". The outcomes of our replicated experiments are detailed in Table 4-1, which presents the macro average F1 scores for both the BiLSTM-CRF and BiLSTM-CRF with CharEmbd models.

Table 4-1

Macro Average F1 Scores of the Trained Baseline Models on FoodBase Dataset

Word Embedding	BiLSTM-CRF		BiLSTM-CRF with CharEmbd	
	Baseline Results	Our Results	Baseline Results	Our Results
GloVe	0.91	0.88	0.92	0.89
Word2Vec	0.90	0.89	0.91	0.90
FastText	0.90	0.88	0.92	0.89
Lexical	0.94	0.91	0.95	0.92

Note: Baseline results are adapted from “BuTTER: bidirectional LSTM for food named-entity recognition”, by G. Cenikj, G. Popovski, R. Stojanov, B. K. Seljak, & T. Eftimov, 2020, In *2020 IEEE International Conference on Big Data (Big Data)* (pp. 3550-3556). IEEE.

Copyright 2020 by the authors.

These results are derived from training the baseline models on a single fold of the *FoodBase* dataset. It is noteworthy that our findings closely align with those reported by Cenikj et al. (2020), underscoring the reproducibility of the baseline model. Consequently, we opted not to employ 5-fold cross-validation, as our results corroborate the baseline model's performance.

Upon a thorough examination of the replicated experiments, we observed that augmenting character embeddings alongside word embeddings in the BiLSTM-CRF model

resulted in an enhanced macro average F1 score across all employed word embedding models. Notably, the lexical word embeddings outperformed other pre-trained embeddings in both models, yielding macro average F1 scores of 0.91 for the BiLSTM-CRF model and 0.92 for the BiLSTM-CRF model augmented with character embeddings.

Furthermore, all the evaluated models consistently achieved a macro F1 score of no less than 0.89, as reported by Cenikj et al. (2020). In our repeated experiments, we achieved a minimum score of 0.88. In addition to macro average scores, we also computed precision, recall, and F1 scores for individual classes. These metrics hold particular significance for our analysis and comparison of various models. It is noteworthy that both models consistently exhibited higher precision and recall for detecting the B-FOOD class compared to the I-FOOD class, irrespective of the chosen embedding model. This observation suggests that the models excel in recognizing the initial portion (token) of a food name entity compared to identifying the subsequent part.

In summary, the results obtained from running the BuTTER system (baseline model) and evaluating the models introduced by Cenikj et al. (2020) establish a foundational baseline for subsequent experiments within our research. These findings serve as a valuable starting point for further investigations and analyses in our study.

Further analysis and exploration of the experiment results can be found in Appendix C1.1.

4.1.2 Results and Analysis of Domain Adaptation

To implement the concept of domain adaptation, our approach involves initially training the baseline models on the *FoodBase* dataset and subsequently evaluating their performance on a subset of 100 annotated reviews from the *ReviewsDB* dataset. The primary objective is to measure the efficacy of our proposed methodology in identifying dish name entities within customer review data, leveraging models initially trained on the *FoodBase* dataset, which primarily consists of food recipes.

Table 4-2 serves as a reference offering a comprehensive comparison of macro average F1 scores derived from our experiments. We assess the performance of two model architectures, namely BiLSTM-CRF and BiLSTM-CRF with CharEmbd, using different word embedding models.

Table 4-2

Macro Average F1 Scores of BiLSTM-CRF and BiLSTM-CRF With CharEmbd Models

Word Embedding	BiLSTM-CRF		CharEmbd-BiLSTM-CRF	
	FoodBase/FoodBase	FoodBase/ ReviewsDB	FoodBase/FoodBase	FoodBase/ ReviewsDB
GloVe	0.88	0.39	0.89	0.40
Word2Vec	0.89	0.51	0.90	0.40
FastText	0.88	0.50	0.89	0.39
Lexical	0.91	0.54	0.92	0.45

Notably, the performance of the proposed models when tested on the *ReviewsDB* dataset yields significantly inferior results for both model architectures, regardless of the word embedding models employed. The highest average macro F1 score achieved, which is 0.54, corresponds to the BiLSTM-CRF model when utilizing the lexical word embedding model. In contrast, the same BiLSTM-CRF model exhibits notably superior performance, achieving a macro F1 score of 0.91 when assessed on the *FoodBase* dataset.

In summation, the findings obviously demonstrate that the attempt to apply domain adaptation by training models on the *FoodBase* dataset for food name identification and subsequently deploying these models for dish name detection in the *ReviewsDB* dataset does not yield satisfactory outcomes. This discrepancy is evident across all evaluation metrics, signifying that the transfer of knowledge between these two domains, i.e., identifying food names in recipes and identifying dish names in customer reviews, does not lead to favorable results.

Further analysis and exploration of the experiment results can be found in Appendix C1.2

4.1.3 Results and Analysis of Dish Name Detection in MenuNER Using Fixed Pre-Trained Embeddings

The outcomes presented in this section stem from the training and evaluation of baseline models on the *MenuNER* dataset, utilizing fixed pre-trained word embeddings. Table 4-3 offers a comprehensive overview of the models' performance, encompassing macro-

average precision, recall, and F1 scores for both the BiLSTM-CRF and BiLSTM-CRF with CharEmbd architectures across various word embeddings.

Table 4-3

Macro-Average Evaluation Metric Scores of BiLSTM-CRF and BiLSTM-CRF With CharEmbd Models Using Fixed Pre-Trained Word Embeddings

Word Embedding Models	Macro Average Scores			Macro Average Scores		
	(BiLSTM-CRF)			(BiLSTM-CRF with CharEmbd)		
	P	R	F1	P	R	F1
glove-twitter-25	0.783	0.638	0.682	0.847	0.821	0.831
glove-twitter-100	0.753	0.655	0.691	0.861	0.811	0.835
glove-twitter-200	0.720	0.661	0.687	0.869	0.802	0.833
glove-wiki-gigaword-50	0.777	0.673	0.708	0.869	0.807	0.832
glove-wiki-gigaword-300	0.742	0.675	0.703	0.855	0.807	0.829
Word2vec-300	0.749	0.659	0.695	0.848	0.820	0.831
FastText-300	0.757	0.699	0.720	0.831	0.839	0.829

The following noteworthy observations emerge from the performance results of the trained models utilizing pre-trained word embeddings:

Glove Embeddings:

- For the BiLSTM-CRF model, the F1-scores range from 0.682 to 0.708.

The highest F1-score is achieved with the *glove-wiki-gigaword-50* embeddings.

- When incorporating character embeddings BiLSTM-CRF with CharEmbd, the F1-scores improve notably, ranging from 0.831 to 0.835. The highest F1-score is achieved with the *glove-twitter-100* embeddings.

Word2Vec and FastText Embeddings:

- The BiLSTM-CRF model produces F1-scores ranging from 0.695 to 0.720, with the highest F1-score using *FastText* embeddings, 0.720.
- Similar to Glove, when character embeddings are added, F1-scores improve, ranging from 0.829 to 0.831. The highest F1-score is achieved with *Word2vec-300*, 0.831.

In summary, the performance results of our trained models using pre-trained word embeddings reveal several noteworthy observations. First and foremost, the inclusion of character embeddings in BiLSTM-CRF with CharEmbd model consistently enhances F1-scores across all word embedding models. This phenomenon stems from the character embeddings' capacity to capture intricate linguistic details and contextual information within words, contributing to more accurate predictions in the named entity recognition task.

Furthermore, when considering different word embedding models, *glove-twitter-100*, *Word2vec-300*, and *FastText-300* consistently emerge as top performers, irrespective of the presence of character embeddings. Several factors contribute to their superiority. The higher-dimensional nature of *Word2vec-300* and *FastText-300* embeddings grants them greater expressive power, facilitating the capture of complex patterns within customer reviews. Additionally, these specific word embedding models are well-known for their high-quality

representations of words and phrases as the quantity of the training data used to create these embeddings contributes to their effectiveness

Finally, *FastText-300* embeddings stand out with exceptional performance, particularly when character embeddings are incorporated. This distinction can be attributed to *FastText*'s unique ability to capture subword information. As previously elaborated in the Thesis, *FastText* divides words into subword units, enabling it to represent words that may not have appeared in the training data and this is an invaluable asset for recognizing named entities within diverse menus. Additionally, *FastText*'s proficiency in handling out-of-vocabulary words by aggregating subword embeddings contributes to its consistent dominance.

In the end, these observations collectively underscore the crucial role of character embeddings, the effectiveness of specific word embedding models, and the distinct advantages offered by *FastText-300* in elevating the performance of named entity recognition, particularly within the context of varied menu data.

Further analysis and exploration of the experiment results can be found in Appendix C1.3.

4.1.4 Results and Analysis of Models Enhanced with Part-of-Speech Tags

Table 4-4 illustrates the performance outcomes of the BiLSTM-CRF model enriched with character embeddings (CharEmbd) when augmented with Part-of-Speech (POS) features, an augmentation not included in the models proposed by Cenikj et al. (2020). This table provides a comprehensive overview of diverse performance metrics, encompassing macro average precision, recall, and F1 scores.

Table 4-4

Macro-Average Evaluation Metric Scores of BiLSTM-CRF With CharEmbd Model Using POS Features as Additional Input

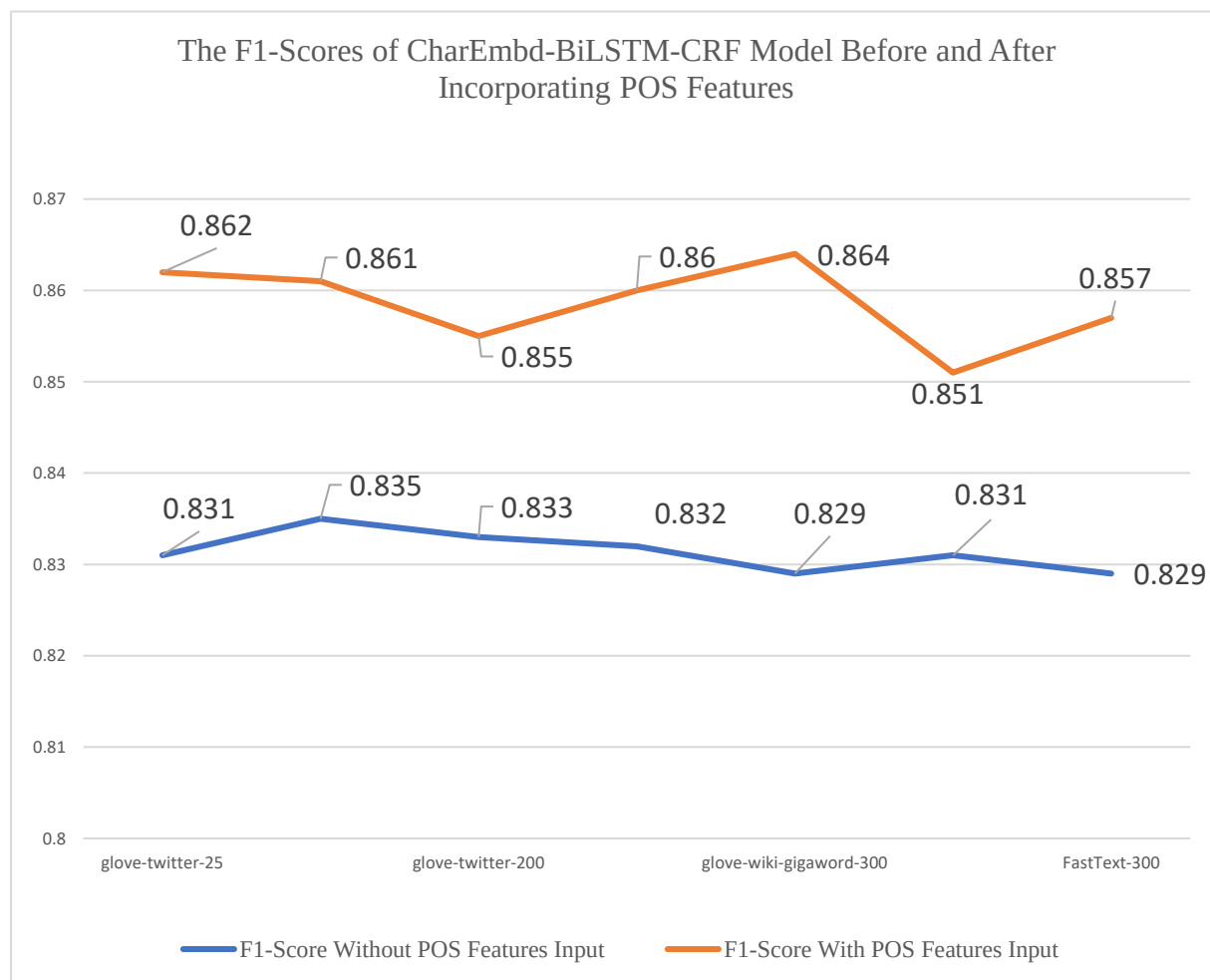
Embedding Models	BiLSTM-CRF With CharEmbd Model With POS		
	Input Features		
	P	R	F1
glove-twitter-25	87.40%	85.47%	86.20%
glove-twitter-100	88.67%	84.00%	86.07%
glove-twitter-200	88.73%	82.27%	85.53%
glove-wiki-gigaword-50	87.60%	85.20%	86.00%
glove-wiki-gigaword-300	89.20%	84.33%	86.40%
Word2vec-300	86.40%	84.60%	85.13%
FastText-300	84.00%	87.67%	85.73%

Table 4-4 shows that precision values range from approximately 84.00% to 89.20%, indicating the model's ability to make accurate positive predictions for named entities. This consistent high precision across different word embedding models underscores the model's reliability in correctly identifying named entities. Similarly, the recall values range from approximately 82.27% to 87.67%, reflecting the model's effectiveness in capturing a substantial proportion of the actual named entities present in the data. The balanced F1-scores, ranging from approximately 85.13% to 86.40%, highlight the model's capacity to balance between accurate predictions and capturing a high proportion of named entities. One notable observation is the consistency in performance metrics across various word embedding models,

indicating the model's robustness. This robustness suggests that the model's performance is not overly reliant on a specific type of word embeddings. This versatility can be attributed to the model's ability to effectively leverage the complementary information provided by Part-of-Speech (POS) tags as additional input features. The inclusion of POS tags augments the model's understanding of the linguistic context within the text, enabling it to capture intricate patterns and relationships between words and their roles in sentences. As a result, the model becomes less sensitive to variations in word embeddings, as it can rely on the supplementary contextual information provided by POS tags to make accurate predictions. Furthermore, the impact of using POS features on the model's performance can be observed in Figure 4-1. The figure compares the macro average F1 scores before and after incorporating the POS features into the input of the BiLSTM-CRF with CharEmbd model.

Figure 4-1

Macro Average F1 Score of BiLSTM-CRF With CharEmb Model Before and After Adding POS Features as Additional Input



By comparing the performance before and after adding the POS features to the model's input it is noted that all performance metrics are relatively consistent across different word embedding models. This suggests that the model's performance is robust and not overly dependent on a specific type of word embeddings. Moreover, the inclusion of Part-of-Speech (POS) input features enhances the model's performance, resulting in higher precision, recall, and F1-scores compared to a model without POS features. This suggests that POS information provides valuable contextual information that aids in named entity recognition.

In summary, the BiLSTM-CRF model with CharEmbd and the addition of Part-of-Speech (POS) input features consistently achieves strong performance in detecting named entities across different word embedding models. The model exhibits high precision, balanced F1-scores, and effective recall, highlighting its effectiveness in the named entity recognition task. The incorporation of POS features further enhances its performance.

Further analysis and exploration of the experiment results can be found in Appendix C1.4

4.1.5 Results and Analysis of BiLSTM-CRF Model Using Contextual Word Embeddings Using BERT

Table 4-5 presents the performance results of the BiLSTM-CRF with CharEmbd model with the incorporation of Part-of-Speech (POS) features, using various Bert embeddings. The metrics evaluated include macro average precision, recall, and F1 score.

Table 4-5

Performance of BiLSTM-CRF With CharEmbd Model With the Use of Different Variations of Bert Embeddings

Embedding Models	Macro Average for BiLSTM-CRF With CharEmbd and POS Input		
	P	R	F1
bert-cased-768	90.60%	82.73%	86.27%
bert-uncased-768	84.40%	85.07%	84.53%
bert-combined-mean-768	87.40%	84.33%	85.73%
bert-combined-conct-1536	84.80%	85.73%	85.07%

One notable observation is the consistently high F1 scores achieved across all Bert embeddings, with values ranging from 84.53% to 86.27%. This suggests that the model when enhanced with POS features and coupled with Bert embeddings, performs well in the task of detecting dish name entities. The highest F1 score of 86.27% is attained when using *bert-cased-768* embeddings. This particular embedding model provides the best balance between precision and recall, resulting in superior overall F1 performance. *bert-cased-768* appears to effectively leverage the contextual information and linguistic details within the customer review text, enabling the model to make accurate predictions as its highest precision of 90.60%, shows its highest capability in detecting dish names in comparison to other Bert variants.

Other Bert embeddings demonstrate strong performance with F1 scores of 85.73%, 85.07 and 84.53%, representing *bert-combined-mean-768*, *bert-combined-conct-1536*, and *bert-uncased-768* respectively.

It can be concluded that the *bert-cased-768* embedding performs slightly better than the other variations in terms of precision, recall, and F1-score and this is attributed to the cased nature of *bert-cased-768* retains the original capitalization of words, which can be crucial for distinguishing between dish entities and non-dish entities, especially in the context of customer reviews where capitalization is often describing proper nouns like dish names. However, it is worth noting that the differences in performance among the variations are relatively small.

On the other hand, Table 4-6 showcases the top three best-performing word embedding models based on their F1-scores.

Table 4-6

Top Performance Results of BiLSTM-CRF with Character Embedding and POS Features as Additional Inputs to the NER Model

Embedding Models	Macro Average		
	BiLSTM-CRF With CharEmbd and POS Tags		
	P	R	F1
glove-wiki-gigaword-300	89.20%	84.33%	86.40%
bert-cased-768	90.60%	82.73%	86.27%
glove-twitter-25	87.40%	85.47%	86.20%

The *glove-wiki-gigaword-300* embedding achieves the highest F1 score of 86.40%, followed by *bert-cased-768*, 86.27%, and then *glove-twitter-25*, 86.20%. These results indicates that with the contextual information augmented with the *bert-cased-768* embeddings, the *glove-wiki-gigaword-300* outperformed the bert embeddings.

In summary, the results show that the *bert-cased-768* embedding and the *glove-wiki-gigaword-300* embedding perform comparably in terms of F1-score, with *bert-cased-768* having a higher precision and *glove-wiki-gigaword-300* having a higher recall.

Further analysis and exploration of the experiment results can be found in Appendix C1.5.

4.1.6 Summary of Results and Analysis

Below is an analysis of the key findings from the various experiments conducted, as detailed in Section 4.1.

Replicated Baseline Experiments (Section 4.1.1)

The replication of BiLSTM-CRF models as per Cenikj et al. (2020) demonstrated high reproducibility, with our F1 scores closely aligning with the original study. It provides a foundation for comparing the model's performance under different conditions. This can be useful for validating the efficacy of the original model as well as its adaptability to new experiments.

Also, the addition of character embeddings generally improved the macro average F1 scores across various word embedding techniques. This improvement can be attributed to different factors. For example, character-level embeddings allow the model to capture more

granular linguistic phenomena like prefixes, suffixes, and internal morphological structures. This level of granularity is often missed by word-level embeddings, providing the model with more context for making accurate predictions. Also, the combination of word-level and character-level features can create a more comprehensive representation of text as word embeddings encapsulate semantic meanings and relationships between words, while character embeddings capture the morphological and syntactical details. The fusion of these features can improve the model's overall understanding of the textual reviews.

In both the BiLSTM-CRF model and its enhanced variant featuring character embeddings, lexical word embeddings outperformed other types of embeddings. This superior performance can be attributed to the model's ability to learn these embeddings during the training phase, thereby capturing contextual information more effectively than fixed, pre-trained word embeddings.

Domain Adaptation (Section 4.1.2)

When domain adaptation was applied to test the models initially trained on the *FoodBase* dataset against the *ReviewsDB* dataset, a significant drop in performance was observed. Across all metrics, the model's capacity to generalize from recognizing food names in recipes to identifying dish names in customer reviews was found lacking. The highest F1 score on the *ReviewsDB* dataset was 0.54, much lower than 0.91 on the *FoodBase* dataset, indicating that the knowledge adaptation between these domains was not effective.

Fixed Pre-trained Word Embeddings (Section 4.1.3)

In experiments involving fixed pre-trained embeddings, the addition of character embeddings universally improved F1 scores, highlighting their importance in capturing

linguistic details. *Glove-twitter-100*, *Word2vec-300*, and *FastText-300* stood out as top-performing embeddings. Notably, *FastText-300* showed particular strength, possibly due to its ability to capture subword information, which is advantageous for named entity recognition in diverse menu data.

Incorporation of Part-of-Speech Tags (Section 4.1.4)

Incorporating Part-of-Speech (POS) tags into the BiLSTM-CRF model with character embeddings produced highly balanced and consistent performance across various word embeddings. The precision, recall, and F1 scores were robust, reflecting the model's ability to leverage the additional linguistic context provided by the POS tags. This enhancement made the model less sensitive to the choice of word embeddings. The high precision and balanced F1-scores suggest that the model is not only accurate but also balanced in its ability to identify true positives and negatives. This points to the model's reliability and efficacy in practical applications.

The Utilization of Contextual Word Embeddings (BERT)

Among the Bert variants, the *bert-cased-768* emerged as the most effective, boasting the highest F1 score of 86.27% as well as the highest precision of 90.60%. The cased nature of this particular embedding seems to enhance its performance by helping the model distinguish more effectively between dish names and other entities, particularly in customer reviews where capitalization often denotes proper nouns like dish names. Also, while the *bert-cased-768* embedding stood out among Bert variants, it's important to note that the differences in performance metrics among various Bert models were relatively small. This suggests that

while *bert-cased-768* may offer slight advantages, other Bert embeddings also provide a robust solution for the task at hand.

In summary, the BiLSTM-CRF model with CharEmbd and POS features yields strong performance in the task of dish name entity recognition when enhanced with advanced word embeddings like Bert and Glove. Both *bert-cased-768* and *glove-wiki-gigaword-300* performed comparably in terms of F1-scores, while *bert-cased-768* had a higher precision of 90.60%, *glove-wiki-gigaword-300* exhibited a higher recall of 84.33%. This indicates that although Bert embeddings are powerful, they are not necessarily the best option for every use case.

In conclusion, the top three most effective fixed word embeddings, when integrated into the BiLSTM-CRF model alongside additional Part-of-Speech (POS) features and character embeddings, are *glove-wiki-gigaword-300*, *bert-cased-768*, and *glove-twitter-25*. These models distinguish themselves based on their F1 scores, which stand at 86.40%, 86.27%, and 86.20%, respectively.

4.2 Analysis of Deep Learning Models with Trainable Word Embeddings

We are discussing and analyzing the obtained results from the experiments we conducted to replicate the baseline experiments that was carried out by Syed and Chung (2021). Subsequently, a focus on the examination of the obtained results by running various NER deep learning models is also presented. These models employ trainable word embeddings and are classified according to the type of the utilized neural layers in each respective architecture.

4.2.1 Results and Analysis of Replicated Baseline Experiments by Syed and Chung (2021)

As outlined in Section 2.3.2, the BERT embeddings, which are used by the named entity recognition model to detect dish names in customer reviews, are generated based on two approaches proposed by Syed and Chung (2021):

- **BERT Domain Embeddings:** This approach involves domain adaptation of the BERT model. The original BERT model is further trained using an unlabeled Yelp customer reviews dataset. This training uses an unsupervised learning approach to achieve a better contextual representation of dish entities.
- **BERT Base Embeddings:** This method involves fine-tuning the BERT model using a labeled customer reviews dataset. Here, domain adaptation is achieved through supervised learning using a specific labeled dataset, namely the *MenuNER* dataset.

Table 4-7 showcases the optimal results obtained by the Named Entity Recognition (NER) models using the two different BERT embeddings approaches. The performance metrics include precision, recall, and F1-score:

Table 4-7

Optimal Experimental Results of Syed and Chung (2021) Models

NER Network (Word Embeddings)	Precision	Recall	F1-Score
BiLSTM-CRF (Bert Domain Embeddings)	92.70	92.30	92.50
BiLSTM-CRF (Bert Base Embeddings)	90.30	91.30	91.80

Note: Adapted from “MenuNER: Domain-adapted BERT based NER approach for a domain with limited dataset and its application to food menu domain”, by M. Syed, & S Chung, 2021, *Applied Sciences*, 11(13). <https://doi.org/10.3390/app11136007>. Copyright 2021 by the authors.

In our study, we either employed pre-trained word embeddings such as *GloVe*, *Word2Vec*, or *FastText*, or we fine-tuned BERT embeddings on the *MenuNER* dataset for the NER task. We chose to compare our results with the latter approach. Unlike methods that necessitate further training of the BERT language model, this approach simply fine-tunes the BERT model on the *MenuNER* dataset for dish name detection. Subsequently, the generated embeddings are incorporated into the BiLSTM-CRF model, as outlined in the work of Syed and Chung (2021).

As outlined by Syed and Chung (2021), the model features two consecutive bidirectional BiLSTM layers. The output from the initial BiLSTM layer acts as the input for the subsequent BiLSTM layer. The output from this second BiLSTM layer then proceeds to a CRF layer for the sequence labeling task. Each BiLSTM layer is equipped with 512 hidden units. The input for the model is constructed by concatenating several elements: the BERT word embeddings derived from *bert-base-cased* (768-dimensional), POS features via one-hot encoding, and character embeddings created through a convolutional neural network (25-dimensional). It’s important to highlight that the BERT embeddings were derived by averaging the hidden outputs from the last four layers of the BERT model.

Regarding the experimental configurations, the parameters were set as follows: batch size is 64, evaluation batch size is 128, the maximum sentence length is 180, the learning rate

is set to 0.001, and a patience threshold of 7 is implemented for early stopping based on the F1 score. The model optimization is carried out using the Adam optimizer. To reproduce the baseline experiment by Syed and Chung (2021), the network was constructed employing all the specified settings from the paper, coupled with the original *MenuNER* datasets (train, valid, and test).

Despite leveraging the resources of Google Colab Pro, multiple attempts to execute the experiment proved unsuccessful. Upon detailed examination, it was identified that the RAM resources provided by Colab were capped at 50 GB. This allocation was inadequate for managing the expansive model and its training parameters during the training phase, which comprised more than 40 million parameters. To mitigate this challenge, adjustments were made to certain model parameters. Specifically, the batch size was decreased to reduce the number of trainable parameters. Table 4-8 displays the outcomes of varying combinations of batch size and evaluation batch size.

Table 4-8

Impact of Batch Size Combinations on Experiment Outcomes

Batch Size	Evaluation Batch Size	Result
64	128	Experiment failed
64	64	Experiment failed
64	32	Experiment failed
32	32	The model converged after 11 epochs and needed taking approximately 10 hours to complete training

As indicated in Table 4-8, successful training of the BiLSTM-CRF model was achieved when both the batch size and the evaluation batch size were reduced to 32, while the remaining combinations led to failed experiment which means that the training phase of the model has crashed before completing the phase.

On the other hand, Table 4-9 contrasts the evaluation metrics from the original baseline experiment, as reported by Syed and Chung (2021), with the outcomes from our replicated experiment.

Table 4-9

Comparison of Original Results By Syed and Chung (2021) and Our Replicated Experiment Outcomes

Performance Metric	Original Results	Replicated Experiment Results
Precision	90.30%	86.3%,
Recall	91.30%	76.7%,
F1-Score	91.80%	77.3%

Note: The original results are adapted from “MenuNER: Domain-adapted BERT based NER approach for a domain with limited dataset and its application to food menu domain”, by M. Syed, & S Chung, 2021, *Applied Sciences*, 11(13). <https://doi.org/10.3390/app11136007>.

Copyright 2021 by the authors.

It's clear that the replicated experiment generated outcomes that vary from the original paper's results. For instance, while the original model achieved an F1-score of 91.80%, our replication yielded 77.3%. Such a variation may stem from our constraints in implementing all

prescribed settings due to limited computational resources. Additionally, there's uncertainty regarding the comprehensiveness of the experimental settings provided in the original paper.

4.2.2 Results and Analysis of Deep Learning Models with Single Recurrent Layer Type

This section is organized into two distinct sections. The first section explores models featuring one or two Bidirectional Long Short-Term Memory (BiLSTM) layers, systematically examining their performance across various optimizer configurations and neuron counts. The subsequent section mirrors this approach, focusing on models incorporating one or two Bidirectional Gated Recurrent Unit (BiGRU) layers. It's crucial to note that in all experiments, the pre-trained word embeddings undergo additional training and adaptation during the models' training phase.

4.2.2.1 Experiment with BiLSTM-CRF and 2BiLSTM-CRF Models. Table 4-10 offers a summary of the experimental findings, presenting an evaluation of the various models' performance. The table highlights the highest macro-average F1 score attained by each model and specifies the trainable word embedding model responsible for achieving this result.

Table 4-10

Best Achieved Results When Experiment with BiLSTM-CRF and 2BiLSTM-CRF Models

	Model Architecture	F1-Score	Word Embeddings
BiLSTM-CRF	BiLSTM(50N)-CRF (rmsprop)	91.13	glove-wiki-gigaword-50
	BiLSTM(25N)-CRF (Adam)	91.53	glove-wiki-gigaword-50
	BiLSTM(50N)-CRF (Adam)	92.53	glove-twitter-25,
		92.53	glove-wiki-gigaword-50

	Model Architecture	F1-Score	Word Embeddings
	BiLSTM(75N)-CRF (Adam)	92.73	glove-twitter-100
	BiLSTM(100N)-CRF (Adam)	92.93	glove-twitter-200
	BiLSTM(125N)-CRF (Adam)	92.60	glove-twitter-100
	BiLSTM(150N)-CRF (Adam)	92.80	glove-wiki-gigaword-300
2BiLSTM-CRF	2BiLSTM(50N)-CRF	91.53	glove-twitter-100,
	(rmsprop)	91.53	glove-wiki-gigaword-300
	2BiLSTM(25N)-CRF (Adam)	91.53	bert-combined-concat-1536
	2BiLSTM(50N)-CRF (Adam)	92.07	glove-twitter-200
	2BiLSTM(75N)-CRF (Adam)	92.33	glove-twitter-100
	2BiLSTM(100N)-CRF (Adam)	92.33	glove-wiki-gigaword-50

Within the BiLSTM-CRF models, the variant with 50 neurons employing the rmsprop optimizer achieved an F1-score of 91.13% when combined with the *glove-wiki-gigaword-50* embeddings. However, a shift to the Adam optimizer led to a performance improvement, raising the score to 92.53% with either *glove-wiki-gigaword-50* or *glove-twitter-25* embeddings. Table 4-11 illustrates how the use of Adam optimizer consistently outperformed the rmsprop optimizer in terms of F1-score, irrespective of the word embeddings used. Also, Adam optimizer highly enhances the precision of the model. For example, for the *bert-uncased-768*, the precision increased from 85.80% to 92.80%.

Table 4-11

Comparative Performance of BiLSTM(50N)-CRF Architectures With Rmsprop and Adam Optimizers Across Various Word Embeddings

Word Embeddings	BiLSTM(50N)-CRF (rmsprop)			BiLSTM(50N)-CRF (Adam)		
	P	R	F1	P	R	F1
glove-twitter-25	91.40%	91.33%	90.93%	93.27%	91.73%	92.53%
glove-twitter-100	90.00%	91.80%	90.80%	93.53%	91.13%	92.27%
glove-twitter-200	89.33%	91.87%	90.60%	93.80%	90.67%	92.27%
glove-wiki-gigaword-50	90.67%	91.73%	91.13%	94.00%	91.13%	92.53%
glove-wiki-gigaword-300	89.27%	91.67%	90.40%	93.87%	90.73%	92.27%
word2vec-300	89.53%	91.60%	90.47%	93.93%	90.33%	92.07%
fasttext-300	86.60%	92.53%	89.20%	93.20%	89.93%	91.47%
bert-cased-768	92.27%	87.33%	89.40%	92.73%	85.67%	88.87%
bert-uncased-768	85.80%	92.20%	88.60%	92.80%	91.20%	92.07%
bert-combined-mean-768	85.13%	92.13%	88.20%	92.80%	91.20%	92.07%
bert-combined-conct-1536	85.93%	92.67%	88.80%	92.53%	91.47%	92.07%
lexical-300	87.33%	92.27%	89.47%	93.93%	90.13%	92.07%

In comparison, the BiLSTM (100)-CRF model, which employed the Adam optimizer and was paired with *glove-twitter-200* embeddings, recorded the highest F1-score of 92.93%. In terms of neuron count, there is a trend: as the number of neurons increased, performance improved, reaching an optimal point at 100 neurons. After this, performance either remained consistent or experienced a slight decrease.

For the 2BiLSTM (75)-CRF model, the highest F1-score of 92.33% was achieved when using either *glove-twitter-100* or *glove-wiki-gigaword-50* embeddings. Another point of note is that the inclusion of *bert-combined-conct-1536* embeddings in a model resulted in an F1-score of 91.53%, a competitive figure, but it did not exceed the scores of the top-performing models.

Overall, models utilizing the Adam optimizer typically exhibited better, or at the very least, comparable performance to those using the rmsprop optimizer. Additionally, the GloVe embeddings sourced from Twitter data showed consistent effectiveness, which may be attributed to the alignment of the nature of the training data with the content of customer reviews.

Further analysis and exploration of the experiment results can be found in Appendix C2.1.

4.2.2.2 Experiment with BiGRU-CRF and 2BiGRU-CRF Models. Table 4-12 offers a comprehensive summary of the results from experiments conducted on the two BiGRU model variations. It systematically lists the highest macro-average F1 scores attained by each model configuration and details the specific trainable word embedding model that played the role in achieving these top-performing results.

Table 4-12

Best Achieved Results When Experiment With BiGRU-CRF and 2BiGRU-CRF Models

	Model Architecture	F1-Score	Word Embeddings
BiGRU-CRF	BiGRU(50N)-CRF (rmsprop)	91.40	glove-twitter-25
	BiGRU(25N)-CRF (Adam)	92.53	glove-twitter-25
	BiGRU(50N)-CRF (Adam)	92.73	bert-uncased-768
	BiGRU(75N)-CRF (Adam)	92.87	glove-wiki-gigaword-300
	BiGRU(100N)-CRF (Adam)	93.07	glove-twitter-100
	BiGRU(125N)-CRF (Adam)	93.00	glove-wiki-gigaword-50
	BiGRU(150N)-CRF (Adam)	92.60	glove-twitter-100,
		92.60	glove-wiki-gigaword-300
2BiGRU-CRF	2BiGRU(50N)-CRF (rmsprop)	91.80	glove-wiki-gigaword-300
	2BiGRU(25N)-CRF (Adam)	92.07	bert-combined-concct-1536
	2BiGRU(50N)-CRF (Adam)	91.73	fasttext-300
	2BiGRU(75N)-CRF (Adam)	92.53	glove-twitter-25
	2BiGRU(100N)-CRF (Adam)	92.40	glove-twitter-200

Table 4-12 reveals that the BiGRU (50)-CRF model when using the rmsprop optimizer and *glove-twitter-25* embeddings is achieving an F1-score of 91.40%. However, when the Adam optimizer is paired with *bert-uncased-768* embeddings, the performance is enhanced to 92.73%, mirroring findings from BiLSTM-CRF models. The best performance for BiGRU-CRF models is represented by the BiGRU(100N)-CRF (Adam) configuration, which achieved an F1-score of 93.07% with *glove-twitter-100* embeddings. It's noteworthy to mention that as

the neuron count increased to 125 and 150, there was a marginal decrement in the F1-score to 93.00% and 92.60% respectively.

Turning to the 2BiGRU-CRF models, specifically those employing the Adam optimizer, the 2BiGRU(75N)-CRF configuration stood out with an F1-score of 92.53% when using the *glove-twitter-25* embeddings. Also, the performance of the other configurations under the Adam optimizer remained in proximity, spanning between 91.73% and 92.40%.

To summarize the findings, the main trend observed is the superior or equivalent performance of models employing the Adam optimizer compared to those using rmsprop. Furthermore, the consistent prominence of GloVe embeddings, particularly those developed from Twitter data.

Further analysis and exploration of the experiment results can be found in Appendix C2.1.

4.2.3 Results and Analysis of Deep Learning Models with Recurrent Layers

This section is divided into two specific sections. The initial section delves into models that comprise one BiLSTM layer coupled with one BiGRU layer. Here, we systematically evaluate their performance metrics across varying optimizer strategies and neuron configurations. The subsequent section is dedicated to the analysis of more complex models that integrate two BiLSTM layers along with two BiGRU layers.

It is important to highlight that for every experiment conducted, the pre-trained word embeddings are further trained and fine-tuned during the models' training process.

4.2.3.1 Experiment with 1BiLSTM+ 1 BiGRU-CRF Models. The results of the experiments are summarized in Table 4-13, which provides an overview of the performance of models incorporating both BiLSTM and BiGRU layers. The table highlights the highest macro average F1 score achieved by each model, along with the corresponding pre-trained word embeddings used to attain that score.

Table 4-13

Best Achieved Results When Experiment With 1BiLSTM+1BiGRU-CRF Models

	Model Architecture	F1-Score	Word Embeddings
1BiLSTM- 1BiGRU- CRF	1BiLSTM(50N)-1BiGRU(50N)-CRF (rmsprop)	91.93	glove-twitter-200
	1BiLSTM(25N)-1BiGRU(25N)-CRF (Adam)	92.93	glove-twitter-100
		92.93	glove-twitter-200
		92.93	glove-wiki-gigaword-50
	1BiLSTM(50N)-1BiGRU(50N)-CRF (Adam)	92.80	glove-twitter-100
		92.80	word2vec-300
	1BiLSTM(75N)-1BiGRU(75N)-CRF (Adam)	92.87	glove-twitter-200
	1BiLSTM(100N)-1BiGRU(100N)- CRF (Adam)	92.80	glove-twitter-100
		92.80	glove-wiki-gigaword-50
		92.80	bert-combined-conct- 1536

Table 4-13 showcases the results of the 1BiLSTM+1BiGRU-CRF models. Consistent with previous experiments, the utilization of the Adam optimizer led to performance improvements for all models, as reflected in their macro average F1 scores. Notably, a superior performance of 92.93% was achieved when using the Adam optimizer with either the *glove-twitter-100*, *glove-twitter-200*, or *glove-wiki-gigaword-50* embeddings. Increasing the neuron count to 50 or 75 showed only marginal changes in performance. However, the F1-score remains relatively unchanged at 92.80% for the configurations with 100 neurons.

It's noteworthy that when the model 1BiLSTM(100N)-1BiGRU(100N)-CRF (Adam) utilized the *bert-combined-conct-1536* embeddings, the F1-score was 92.80%, which is aligned with the scores achieved using other embeddings for the same model configuration. This suggests that the *bert-combined-conct-1536* embeddings, despite their complexity, offer comparable performance to simpler embeddings like *glove-twitter-100* and *glove-wiki-gigaword-50* for this model architecture.

4.2.3.2 Experiment with 2BiLSTM+ 2 BiGRU-CRF Models. The experimental results for the 2BiLSTM+2BiGRU-CRF models are presented in Table 4-14. The table summarizes the performance of these models using different optimizers (rmsprop and Adam) and highlights the highest macro average F1 score achieved, along with the corresponding pre-trained word embeddings utilized.

Table 4-14

Best Achieved Results When Experiment With 2BiLSTM-2BiGRU-CRF Models

	Model Architecture	F1-Score	Word Embeddings
2BiLSTM- 2BiGRU- CRF	2BiLSTM(50N)-2BiGRU(50N)-CRF	91.47	glove-twitter-200
	(rmsprop)	91.47	glove-wiki-gigaword-300
	2BiLSTM(25N)-2BiGRU(25N)-CRF	92.40	glove-wiki-gigaword-300
	(Adam)		
	2BiLSTM(50N)-2BiGRU(50N)-CRF	92.60	bert-uncased-768,
	(Adam)	92.60	bert-combined-mean-768
	2BiLSTM(75N)-2BiGRU(75N)-CRF	92.67	bert-uncased-768
	(Adam)	92.67	bert-combined-mean-768
	2BiLSTM(100N)-2BiGRU(100N)- CRF (Adam)	92.47	glove-twitter-200

According to the results, contrary to the improvements observed in the models with a single BiLSTM and BiGRU layer, the addition of two layers does not lead to enhanced performance. The best F1 score achieved is 92.67%, obtained with either *bert-uncased-768* or *bert-combined-mean-768* embeddings for the 2BiLSTM(75N)-2BiGRU(75N)-CRF (Adam) model. In comparison, the simpler model 1BiLSTM(75N)-1BiGRU(75N)-CRF which achieved a higher F1 score of 92.87% with *glove-twitter-200* embeddings.

Different from Table 4-13, the Table 4-14 demonstrates a favorable performance for the BERT-based embeddings, like *bert-uncased-768* and *bert-combined-mean-768* in multiple

configurations. However, *GloVe* embeddings, particularly those trained on Twitter data and the *glove-wiki-gigaword-300*, still remained effective.

In summary, while the 2BiLSTM-2BiGRU-CRF model architecture offers comparable performance to the 1BiLSTM-1BiGRU-CRF models, the added complexity does not necessarily.

Further analysis and exploration of the experiment results can be found in Appendix C2.2.

4.2.4 Results and Analysis of Deep Learning Models with Recurrent and CNN Layers

In this section, the assessment of the models that combine the features of convolutional neural network (CNN) layers with recurrent neural network (RNN) layers is presented. The aim is to see how these combinations affect performance. This section has two sections. In the first section, we look at models that use one RNN layer and one CNN layer. We also study a model that uses BiLSTM, BiGRU, and CNN layers together. This mixed model aims to use the best features of each layer for better results in detecting dish names.

In the second section, we focus on models that use two BiGRU layers with one CNN layer. Different settings, including changing optimizers and the number of neurons, are tested to find the best setup.

For all these tests, the pre-trained word embeddings are further trained during the model training.

4.2.4.1 Experiment with 1BiLSTM-1CNN-CRF, 1BiGRU-1CNN-CRF and 1BiLSTM-1BiGRU-1CNN-CRF Models. Table 4-15 presents a summary of the experimental results for the various models. The evaluation compares the performance of different models optimized by Adam optimizer under different configurations in terms of the number of neurons in hidden layers of the models.

The table also highlights the highest macro average F1 score achieved by each model, along with the corresponding pre-trained word embeddings used to attain that score.

Table 4-15

Best Achieved Results When Experiment With 1BiLSTM-CNN-CRF, 1BiGRU-1CNN-CRF , and 1BiLSTM-1BiGRU-1CNN-CRF Models

	Model Architecture	F1-Score	Word Embeddings
1BiLSTM-	1BiLSTM(25N)-1CNN-CRF (Adam)	92.47	glove-twitter-25
1CNN-	1BiLSTM(50N)-1CNN-CRF (Adam)	92.67	glove-twitter-25
CRF	1BiLSTM(75N)-CNN-CRF (Adam)	92.53	glove-twitter-100
	1BiLSTM(100N)-1CNN-CRF (Adam)	92.33	glove-wiki-gigaword-50
1BiGRU-	1BiGRU(25N)-1CNN-CRF (Adam)	92.67	glove-wiki-gigaword-300
1CNN-	1BiGRU(50N)-1CNN-CRF (Adam)	92.87	glove-twitter-200
CRF	1BiGRU(75N)-CNN-CRF (Adam)	92.67	glove-twitter-200
		92.67	glove-wiki-gigaword-50
	1BiGRU(100N)-1CNN-CRF (Adam)	92.87	glove-twitter-25
1BiLSTM-	1BiLSTM(25N)-1BiGRU(25N)-	92.67	glove-twitter-25, word2vec

	Model Architecture	F1-Score	Word Embeddings
1BiGRU-	1CNN-CRF (Adam)		
1CNN-	1BiLSTM(50N)-1BiGRU(50N)-	92.93	glove-twitter-25
CRF	1CNN-CRF (Adam)		
	1BiLSTM(75N)+1BiGRU(75N)-	92.87	glove-twitter-100
	1CNN-CRF (Adam)		
	1BiLSTM(100N)-1BiGRU(100N)-	92.67	glove-wiki-gigaword-50
	1CNN-CRF (Adam)		

The following are the main observations on the results reported in Table 4-15:

For **1BiLSTM-1CNN-CRF Models**: The models primarily leverage the *glove-twitter-25* embeddings, with the F1-scores ranging between 92.33% to 92.67% as the number of neurons in the BiLSTM layer increases from 25 to 50. However, a slight decline in performance is noted when moving from 75 neurons to 100 neurons. This might suggest that there's an optimal complexity level for this model architecture, and beyond a certain point, adding more neurons might not significantly benefit the performance.

For **1BiGRU-1CNN-CRF Models**: These models appear to be slightly more consistent in their performance across varying neuron counts. The F1-score hovers around 92.67% to 92.87%. The model with 50 neurons using the *glove-twitter-200* embeddings achieves the highest F1-score of 92.87%. It's noteworthy that the *glove-twitter-25* embeddings also allow the model with 100 neurons to reach the same highest performance, suggesting that these embeddings are particularly effective in this context of customer review data.

For **1BiLSTM-1BiGRU-1CNN-CRF Models**: These combined models, which utilize both BiLSTM and BiGRU layers alongside a CNN layer, show strong performance. The model with 50 neurons for both BiLSTM and BiGRU layers, using *glove-twitter-25* embeddings, achieves the highest F1-score of 92.93% among all the models listed. This suggests that the complementary strengths of BiLSTM and BiGRU might be advantageous when combined in this configuration.

In summary, the experimental results indicate that the 1BiGRU(75N)-CNN-CRF model generally outperforms the 1BiLSTM(75N)-CNN-CRF model, except when *glove-twitter-100* embeddings are employed. Moreover, the addition of the CNN layer to the 1BiLSTM(75N)+1BiGRU(75N)-CRF model preserves its superior performance, with the highest F1 score of 92.87 achieved using *glove-twitter-100* embeddings.

4.2.4.2 Experiment with 2BiGRU-1CNN-CRF Models. Table 4-16 showcases the optimal results obtained from the experiments where the outputs of two BiGRU layers are concatenated with the output of a CNN layer to contribute to the final prediction of the model.

Table 4-16*Best Achieved Results When Experiment With 2BiGRU-1CNN-CRF Models*

	Model Architecture	F1-Score	Word Embeddings
2BiGRU-	2BiGRU(75N)-1CNN-CRF (rmsprop)	92.00	glove-twitter-25
1CNN-CRF	2BiGRU(25N)-1CNN-CRF (Adam)	92.67	glove-twitter-200
	2BiGRU(50N)-1CNN-CRF (Adam)	92.53	glove-twitter-100
		92.53	glove-wiki-gigaword-300
		92.53	bert-combined-concat
	2BiGRU(75N)-1CNN-CRF (Adam)	92.53	word2vec-300
	2BiGRU(100N)-1CNN-CRF (Adam)	92.60	glove-twitter-25

The results in Table 4-16 demonstrate that by replacing the rmsprop optimizer with Adam, the F1-score of the 2BiGRU(75N)-1CNN-CRF model improves from 92.00% with *glove-twitter-25* embeddings to 92.53% with *word2vec-300* embeddings. Also, it can be observed that the model with the lowest neuron count (25 neurons) combined with the Adam optimizer and *glove-twitter-200* embeddings achieves the highest F1-score of 92.67%. This indicates that a simpler model architecture can sometimes be more effective, possibly because it's less prone to overfitting. As the neuron count increases, the F1-score remains relatively consistent, floating in the range of 92.53% to 92.60%. This suggests that, after a certain point, increasing the complexity of the model by adding more neurons doesn't necessarily yield significant improvements in performance for the named entity task and the *MenuNER* dataset.

Moreover, the *glove-twitter* series of embeddings appears to be consistently effective across various model configurations. It's also worth noting that the model with 50 neurons achieved the same F1-score of 92.53% with three different embeddings: *glove-twitter-100*, *glove-wiki-gigaword-300*, and *bert-combined-concat*. This might suggest that the architecture is robust enough to deliver consistent results across different embedding types.

In summary, the 2BiGRU-1CNN-CRF architecture with the Adam optimizer consistently provides competitive results, with slight variances depending on the neuron count and word embeddings. The *glove-twitter* embeddings, in particular, seem to be highly effective for this specific task. However, it's evident that simply increasing model complexity by increasing neuron counts does not always lead to improved performance

Further analysis and exploration of the experiment results can be found in Appendix C2.3.

4.2.5 Summary of Results and Analysis

Below is an analysis of the key findings from the various experiments conducted, as detailed in Section 4.2.

Replicated Baseline Experiments (Section 4.2.1)

The replicated experiment produced different results from the original study by Syed and Chung (2021). For example, the original F1-score of the BiLSTM-CRF model that utilizes the generated Bert embeddings via fine-tuning the BERT model on the *MenuNER* dataset is 91.80%, while the achieved score after replicating the experiment is 77.3%. The variations in

achieved results could be attributed to the computational constraints during the replication and potential ambiguities in the original experimental settings.

Deep Learning Models with Single Recurrent Layer Type (Section 4.2.2)

The choice of word embeddings significantly influenced the models' performance. Different word embeddings capture distinct semantic and contextual information from the text. In this context, the BiLSTM-CRF models and 2BiLSTM-CRF achieved optimal results when employing *glove-twitter* embeddings. For example, the BiLSTM(100)-CRF(Adam) model achieved the highest F1 score of 92.93 when *glove-twitter-200* embedding was employed. While the 2 BiLSTM(75)-CRF(Adam) and 2 BiLSTM(100)-CRF(Adam) achieved the highest score of 92.33 when employing *glove-twitter-100* and *glove-wiki-gigaword-50* respectively.

The utilization of *GloVe* embeddings contributed to the models' superior performance, as these embeddings contain more relevant and informative features for the respective models. Additionally, variations in the performance based on different word embeddings indicate that each embedding type captures distinct aspects of semantic and syntactic information, affecting the model's performance. Notably, the F1 scores of models using *GloVe* embeddings surpassed those using Bert embeddings

On the other hand, the addition of an extra BiLSTM layer in the 2BiLSTM-CRF models or an extra BiGRU layer in the 2BiGRU-CRF models did not result in performance improvement. Contrarily, the performance of these models with a single layer outperformed those with two layers. This outcome may be attributed to overfitting. The 1BiLSTM-CRF and 1BiGRU-CRF models potentially exhibit lower susceptibility to overfitting compared to their respective two-layer counterparts. The introduction of an extra layer increases model

complexity, leading to a higher number of trainable parameters. In cases where training data is limited, as with the *MenuNER* dataset, this increased complexity may cause overfitting, resulting in reduced generalization performance on unseen data. Consequently, the 1BiLSTM-CRF and 1BiGRU-CRF models, with fewer parameters, strike a better balance between model complexity and the available training data, resulting in superior performance.

Deep Learning Models with Recurrent Layers (Section 4.2.3)

Combining the effect of using two recurrent layers has improved the results in terms of the F1-score. For example, the BiLSTM(25)-CRF(Adam) and the BiGRU(25)-CRF(Adam) models have achieved F1-score of 91.53% and 92.53%, while the model BiLSTM(25) + BiGRU(25)-CRF(Adam) enhances the performance and achieved an F1-score of 92.93%.

In the case of the 2BiLSTM+2BiGRU-CRF models, the addition of two layers does not lead to improved performance compared to the single-layer models. Again, this could be due to the model's complexity and the limited size of the dataset, *MenuNER*. While increased model complexity can potentially capture more complex patterns and dependencies in the data, it does not always guarantee improved performance. In this case, the additional layers may introduce more complexity without significantly enhancing the model's ability to detect dish names in customer reviews.

Deep Learning Models with Recurrent and CNN Layers (Section 4.2.4)

The findings suggest that the addition of a single CNN layer does not significantly enhance the performance of the existing models. This may suggest that the CNN layer failed to provide additional information to the recurrent layers BiLSTM or BiGRU. The CNN layer primarily focuses on local patterns and features in the input data, while the recurrent layers

capture sequential dependencies and long-term contextual information. Since dish name detection requires capturing both local and global patterns, the CNN layer alone may not contribute significantly to the overall performance improvement. Another reason might be that the nature of the data used for dish name detection, i.e. customer reviews, might not benefit significantly from the added complexity of the CNN layer. If the input data lacks complex local patterns or if the relevant information can be adequately captured by the recurrent layers alone, the CNN layer may not contribute significantly to the overall performance improvement. Finally, introducing a CNN layer to the existing models might increase the risk of overfitting. The additional layer introduces more parameters and complexity to the model, which can lead to over-optimization of the training data and reduced generalization of unseen data. This could explain why the performance of the models did not improve despite the addition of the CNN layer.

General Observations (Section 4.2.1), (Section 4.2.2), (Section 4.2.3), and (Section 4.2.4)

For all trained models, the use of trainable word embeddings led to notable improvements in performance metrics across all experimented models. The recurrent layers succeeded in extracting more contextual information from the pre-trained word embeddings leading to the enhancement of the model's performance and the increase of the F1 score of the models.

Furthermore, the utilization of the Adam optimizer consistently yielded improved performance compared to rmsprop. This improvement can be attributed to the adaptive learning rate and momentum of the Adam optimizer. By dynamically adjusting the learning rate for each parameter during training, Adam facilitates faster convergence and enhanced

optimization. In contrast, rmsprop employs a fixed learning rate, which may be less effective in certain scenarios. Consequently, models trained with Adam achieved higher F1 scores than those trained with rmsprop.

Moreover, the increase in the number of neurons in the hidden layers does not always result in enhancing the models' performance. Instead, after a certain point, increasing the complexity of the model by adding more neurons doesn't yield significant improvements in performance for the named entity task. This could be due to risks of overfitting, where the model memorizes training data rather than generalizing to new inputs. Additionally, larger models can face challenges like vanishing or exploding gradients, making training difficult. Furthermore, with computational and data limitations, there's a point where added complexity yields diminishing or even negative returns.

The outcomes for the 53 models tested on the *MenuNER* dataset can be found in Appendix C2.4. For a quick overview, Table 4-17 lists the best-performing models for detecting dish names in the *MenuNER* dataset. Specifically, this table highlights the models that achieved the top four macro average F1-scores, which are between 92.87 and 93.07.

Table 4-17

The Top Four F1-Scores Achieved By Our Experimented Models on MenuNER Dataset

Model	F1-Score	Trainable Parameters	Word Embeddings
1BiGRU(100N)-CRF (Adam)	93.07	1,260,780	glove-twitter-100
1BiGRU(125N)-CRF (Adam)	93.00	742,330	glove-wiki-gigaword-50
1BiLSTM(50N)-1BiGRU(50N)-1CNN-CRF (Adam)	92.93	420,855	glove-twitter-25
1BiLSTM(25N)-1BiGRU(25N)-CRF (Adam)		602,830	glove-wiki-gigaword-50
		1,162,580	glove-twitter-100
		2,282,080	glove-twitter-200
BiLSTM(100N)-CRF (Adam)		2,477,880	glove-twitter-200
1BiGRU(100N)-1CNN-CRF (Adam)	92.87	437,055	glove-twitter-25
1BiLSTM(75N)-1BiGRU(75N)-1CNN-CRF (Adam)		1,418,780	glove-twitter-100
1BiGRU(50N)-1CNN-CRF (Adam)		2,362,330	glove-twitter-200
1BiLSTM(75N)-1BiGRU(75N)-CRF (Adam)		2,548,580	glove-twitter-200
1BiGRU(75N)-CRF (Adam)		3,465,980	glove-wiki-gigaword-300

The following points highlight the main findings in Table 4-17:

- The 1BiGRU(100N)-CRF (Adam) model stands out as the top performer, achieving the highest F1-score of 93.07%. Interestingly, this model

employs the *glove-twitter-100* word embeddings and consists of 1,260,780 trainable parameters. It's noteworthy how word embeddings from the *glove-twitter* variants seem to be favored among the models, suggesting their utility for our task. Especially, the *glove-twitter-100* and *glove-twitter-200* embeddings are recurrent, indicating their potential effectiveness in achieving superior F1-scores. However, not all models with high trainable parameters demonstrate a high F1-score. The model 1BiGRU(125N)-CRF (Adam) proves to be quite efficient, achieving an F1-score of 93.00% with 742,330 trainable parameters only. This observation implies that, in this case, higher complexity in terms of trainable parameters does not necessarily equate to higher performance.

- The table showcases a range of models with diverse complexities, from simpler architectures like 1BiGRU(100N)-CRF (Adam) to more complicated ones such as 1BiLSTM(75N)-1BiGRU(75N)-1CNN-CRF (Adam). Yet, the results emphasize that a rise in model complexity doesn't necessarily guarantee a high increase in F1-score.
- Models that incorporate CNN layers, like the 1BiLSTM-1BiGRU-1CNN-CRF, achieve competitive F1-scores close to the top-performing models, which suggests the potential of using CNN layers in conjunction with recurrent architectures for the named entity recognition task. Single-layer BiGRU and single-layer BiLSTM models are also prominent in achieving high F1-scores. It's evident that BiGRU architectures, in particular, seem to perform very well, as they appear multiple times at the top of the performance spectrum.

- Another consistent pattern across the models is the usage of the Adam optimizer. This indicates its effectiveness across different neural architectures for this specific task.

In summary, while the 1BiGRU(100N)-CRF (Adam) model with glove-twitter-100 embeddings emerges as the leader in F1-score, it's pivotal to consider both performance and complexity when determining a model for practical applications. The choice of word embeddings, particularly from the *glove-twitter* variants, has a notable impact on model efficacy. Furthermore, BiGRU architectures, whether standalone or combined with other layers like CNN, consistently yield high performance.

4.3 Comparative Analysis: Our Proposed Models Vs. Syed and Chung (2021) Models

Based on the analysis of our experimental data, the three most effective deep learning models for detecting dish names from *MenuNER* dataset, in terms of macro average F1 scores, were BiGRU(100N)-CRF, 1BiGRU(125N)-CRF, and 1BiLSTM(50N)-1BiGRU(50N)-1CNN-CRF. All these models utilized the Adam optimizer during training. Table 4-18 provides a detailed breakdown of the average macro precision, recall, and F1-score for each model. Additionally, the table presents the specific word embeddings employed and the total number of trainable parameters for each model.

Table 4-18

Comparison of State-of-the-Art Results: Our Proposed Models Vs. Syed and Chung (2021)

Model Architecture	Trainable Parameters	Precision	Recall	F1- Score	Word Embeddings
Our Proposed Models					
1BiGRU(100N)- CRF (Adam)	1,260,780	94.33%	91.80%	93.07%	glove-twitter-100
1BiGRU(125N)- CRF (Adam)	742,330	94.00%	91.67%	93.00%	glove-wiki-gigaword-50
1BiLSTM(50N)- 1BiGRU(50N)- 1CNN-CRF (Adam)	420,855	93.67%	91.93%	92.93%	glove-twitter-25
Proposed Models by Syed and Chung (2021)					
BiLSTM -CRF	Over 40M	90.30%	91.30%	91.80%	bert-base-cased-768
	110M +	92.70%	92.30%	92.50%	Bert domain
	Over 40M				embeddings

Note: The best results in literature for the NER task in detecting dish names are adapted from “MenuNER: Domain-adapted BERT based NER approach for a domain with limited dataset and its application to food menu domain”, 2021, *Applied Sciences*, 11(13).

<https://doi.org/10.3390/app11136007>. Copyright 2021 by the authors.

When comparing our proposed models with the proposed models by Syed and Chung (2021), a comprehensive evaluation across multiple dimensions emerges, as follows:

Model complexity and trainable parameters: The 1BiGRU(100N)-CRF model has 1,260,780 trainable parameters, which is substantially fewer than the Syed and Chung model which utilized *bert-base-cased-768* embeddings and has more than 40 million trainable parameters. The latter model is considerably more complex in terms of its parameter count, suggesting a potentially longer training time and a greater computational demand.

Remarkably, the performance of our best models has also surpassed the state of art model, proposed by Syed and Chung (2021), which is based on providing the BiLSTM-CRF model with Bert embeddings after training the bert-base model to learn from unlabeled customer reviews. The additional training of BERT model required the use of 110 million parameters!

Performance metrics: The 1BiGRU(100N)-CRF model has a precision of 94.33%, which is noticeably higher than the 90.3% and 92.70% achieved by Syed and Chung's models. Also, the BiGRU model has achieved slightly higher recall of 91.8% in compared to 91.30 achieved by Syed and Chung's first model but less than the obtained recall by the second model which is 92.30%. When it comes to the harmonic mean of precision and recall, the 1BiGRU(100N)-CRF model has a higher F1-score at 93.07%, compared to the 91.3% and 92.50% of Syed and Chung's models. This indicates a better overall performance for the 1BiGRU(100N)-CRF model in terms of balancing precision and recall.

Word embeddings: The 1BiGRU(100N)-CRF model utilizes the *glove-twitter-100* word embeddings. These embeddings are based on Twitter data and have a dimensionality of 100. In contrast, Syed and Chung's models use the *bert-base-cased-768* embedding, derived

from the BERT model, which is known for its robustness in various NLP tasks. The choice of BERT embedding suggests a possible emphasis on capturing more complex contextual information in the *MenuNER* dataset with a higher dimensionality of 768 embeddings. However, Glove embeddings proved their superiority in providing the best contextual understanding of the dataset in compared to Bert embeddings.

In conclusion, the 1BiGRU(100N)-CRF model exhibits superior performance in terms of F1-score, precision and recall with significantly fewer parameters. The choice of embeddings also showcases different strategies, with the 1BiGRU(100N)-CRF model leveraging more domain-specific embeddings from Twitter, while Syed and Chung's model relies on the power of BERT embeddings generated by fine tuning the BERT model on the *MenuNER* dataset. The efficiency and effectiveness of the 1BiGRU(100N)-CRF model are commendable, especially given its reduced complexity compared to the model proposed by Syed and Chung.

4.4 Analysis of Experimental Results on the *ReviewsDB* Dataset

About 49 models were trained on the *ReviewsDB* dataset for the named entity recognition task of dish names on customer reviews. Our aim is to compare the obtained results after evaluating the performance of different deep-learning models on *MenuNER* dataset. It is important to note that we've experimented with the same pre-trained word embeddings (Glove, Word2Vec, and FastText) as with *MenuNER* dataset. However, Bert embeddings are not experimented on *ReviewsDB* dataset.

The outcomes for the 49 models tested on the *ReviewsDB* dataset can be found in Appendix C.3 For a quick overview, Table 4-19 lists the best-performing models for detecting

dish names in the *ReviewsDB* dataset. Specifically, this table highlights the models that achieved the top four macro average F1-scores, which are between 82.13 and 82.40.

Table 4-19

The Top Four F1-Scores Achieved by Our Experimented Models on ReviewsDB Dataset

Model	F1-Score	Trainable Parameters	Word Embeddings
2BiGRU(75N)-1CNN-CRF (Adam)	82.40	467,125	glove-twitter-25
	82.40	1,270,150	glove-twitter-100
1BiLSTM(100N)-1BiGRU(100N)-1CNN-CRF (Adam)	82.27	1,232,780	glove-twitter-100
1BiLSTM(75N)-1BiGRU(75N)-CRF (Adam)	82.20	1,259,880	glove-twitter-100
1BiLSTM(100N)-1BiGRU(100N)-CRF (Adam)	82.20	813,100	glove-wiki-gigaword-50
2BiLSTM(50N)-2BiGRU(50N)-CRF (Adam)	82.20	463,105	glove-twitter-25
1BiGRU(75N)-1CNN-CRF (Adam)	82.13	632,650	glove-wiki-gigaword-50
1BiLSTM(50N)-1BiGRU(50N)-1CNN-CRF (Adam)	82.13	665,230	glove-wiki-gigaword-50
2BiGRU(100N)-1CNN-CRF (Adam)	82.13	858,580	glove-wiki-gigaword-50

As shown in Table 4-19, the F1-scores for the models on the *ReviewsDB* dataset range from 82.13% to 82.40%. This indicates a close performance among the top models, suggesting that this particular task's complexities might limit how much models can differ in their effectiveness. However, even slight improvements in F1-score can be significant for our named entity recognition task.

The model achieving the highest F1-score of 82.40% is the 2BiGRU(75N)-1CNN-CRF (Adam) configuration. Two variants of this model achieved the same F1-score but differ in the number of trainable parameters and word embeddings used. The first variant uses *glove-twitter-25* with 467,125 trainable parameters, while the second variant uses *glove-twitter-100* and has 1,270,150 parameters. This suggests that increasing the dimensionality of word embeddings from 25 to 100 in this case doesn't necessarily result in performance enhancement, even though it considerably raises the model's complexity.

Additionally, it's noteworthy that models with diverse architectures, like 1BiLSTM(100N)-1BiGRU(100N)-1CNN-CRF (Adam) and 2BiLSTM(50N)-2BiGRU(50N)-CRF (Adam), both achieved an F1-score of 82.20%. This highlights that a combination of BiLSTM and BiGRU layers with different neuron counts can lead to similar performance on this dataset, given appropriate configurations.

Regarding word embeddings, both the *glove-twitter* and *glove-wiki-gigaword* variants are popular choices among the top-performing models. However, increasing the dimensionality doesn't always correlate with performance improvement. For instance, models with *glove-twitter-100* and *glove-twitter-25* have both achieved top scores. Similarly, models

with *glove-wiki-gigaword-50* consistently appear across different F1-score ranges, indicating its versatility.

Lastly, the presence of the CNN layer in several top-performing models suggests its effectiveness in capturing local features from word embeddings, which may be beneficial for the dish name detection task on similar dataset to *ReviewsDB*.

In summary, while several models demonstrate competitive performance on the *ReviewsDB* dataset, the selection of word embeddings and model complexity should be carefully balanced, as increased complexity doesn't always guarantee better results.

4.5 Comparing and Analyzing Model Performance: *MenuNER* Vs. *ReviewsDB*

The analysis of the obtained results by different deep learning models to detect dish names in the customer reviews dataset, *MenuNER* and *ReviewsDB* datasets, demonstrate the high difference in performance. For example, the best F1-score achieved on the *MenuNER* dataset is 93.07% with the use of 1BiGRU(100N)-CRF model utilizing glove-twitter-100 embeddings. On the other hand, the best achieved F1-score on the *ReviewsDB* dataset is 82.40 with the use of 2BiGRU(75N)-1CNN-CRF model utilizing either glove-twitter-25 or glove-twitter-100. The high difference between the top F1-scores 93.07% and 82.40% encouraged us to investigate the two datasets as both datasets are designed for named entity recognition (NER) task focused on identifying dish names in Yelp customer reviews. They share common label categories such as B-FOOD for the beginning of a dish name, I-FOOD for the inside of a dish name, and O for other tokens. Also, all conducted experiments, deep learning models, the experimental settings, and the models' hyperparameters were identical between the two datasets.

After a deep investigation of both datasets, the following conclusions are drawn:

- Both datasets have a similarity in the types of tokens categorized as O which predominantly include punctuation and common English words.
- *MenuNER* is consistently larger in terms of the total number of tokens across all five folds, offering more data for model training. For example, *MenuNER* comprises a total of 168,804 tokens when aggregating its original training, validation, and testing sets. In contrast, the 1000 reviews in *ReviewsDB* amount to 123,658 tokens. Consequently, the token count in each of the five folds of *MenuNER* exceeds that in *ReviewsDB*.
- In terms of label distribution, both datasets are heavily skewed towards the O label, but *MenuNER* generally has a higher percentage of O labels and fewer instances of B-FOOD and I-FOOD. For instance, in each fold of *MenuNER*, only 5% of the tokens are representative of either complete dish names or parts thereof, while the remaining 95% of tokens do not relate to dish names. In the *ReviewsDB* dataset, an average of 6.4% of labeled tokens signify either full dish names or their components, leaving 93.6% of the tokens to represent other types of information.
- Another crucial difference lies in the specificity of food items. *MenuNER* often includes more traditional or specialized dishes like "naan", "curry" and "tandoori" whereas *ReviewsDB* tends to feature more general and Western food items like "ice cream" "pizza" and "fries".

- The case sensitivity of tokens also varies between the two datasets, with *MenuNER* generally using proper capitalization and *ReviewsDB* often using lowercase, even for pronouns like "I"
- The transition between labels in sequences also varies. *ReviewsDB* exhibits a higher frequency of transitions from I-FOOD to another I-FOOD potentially making the NER task more challenging. Furthermore, the complexity of annotated segments differs, with *MenuNER* generally having shorter and less complex food name annotations, while *ReviewsDB* features longer and more varied segments.
- Finally, the tokens appearing in the I-FOOD category also show differences in context. In *MenuNER*, these are often specific continuations of food names, like "masala" following "chicken" making it easier for models to understand the context. In contrast, *ReviewsDB* often include more general terms like "and" and "with" in the I-FOOD category, which could make the context less clear for NER models.

Overall, these differences likely contribute to *MenuNER*'s better performance in NER tasks, as its annotations are more uniform, specific, and simpler, thereby making it easier for models to learn effectively. On the other hand, *ReviewsDB* presents a more challenging environment due to its complexity and variety.

Chapter 5 : Conclusion and Future Work

This chapter is organized into five distinct sections. It begins with an overview of the research, progresses to a detailed examination of the primary findings from our research questions, and highlights the novel contributions of our study. We also discuss the limitations faced during our research before wraps up by recommending directions for future research.

5.1 Research Summary

Customer reviews on platforms like Yelp and social media increasingly shape the reputation of restaurants and drive their business strategies. Such feedback provides critical insights, highlighting popular dishes and pinpointing areas for enhancement. The literature indicates that while extensive research exists on food entity recognition from structured or semi-structured sources, such as recipe databases (Komariah et al., 2022), there remains a notable absence in the specific domain of extracting dish names from customer reviews. Many extant NER systems are generalist in nature, and not tailored to address the distinct challenges inherent in this task. This gap is further widened by the scarcity of dedicated, annotated datasets essential for training and validating models specific to dish name extraction (Syed & Chung, 2021).

Moreover, in the food name NER domain, there's a notable lack of diverse deep learning model exploration. Furthermore, no efforts have been made to investigate the efficacy of domain adaptation between detecting food names in recipes and identifying dish names in customer reviews. Additionally, the proposed state of art NER system for detecting dish names in customer reviews in literature suffers from the intensive required computational resources to achieve the task.

Therefore, our research focuses on building an effective deep learning named entity recognition model for accurately detecting dish names in restaurant customer reviews with the utilization of moderate computational resources. The study spanned various dimensions, including the effect of dataset choice, the performance of domain adaptation, the influence of pre-trained word embeddings, and the comparison of different deep learning architectures. For a comprehensive evaluation, our experiments were conducted across two unique datasets, providing insights into how models perform under different data conditions.

5.2 Key Findings

The core insights from our study stem from our rigorous exploration aimed at addressing our research questions. For clarity, we revisit our research questions and report the main findings:

- **RQ1:** *How does the performance of NER models trained on our manually annotated ReviewsDB dataset compared to those trained on the benchmark MenuNER dataset in the specific task of dish name detection?*

Research outcomes of experimenting with a benchmark dataset, MenuNER, and our manually annotated dataset, ReviewsDB: At the commencement of our study, the unavailability of a dataset specifically annotated for dish names in customer reviews encouraged us to create our own semi-manually annotated dataset, *ReviewsDB*, extracted from restaurant customer reviews on the Yelp platform. During this period, Syed and Chung (2021) introduced the *MenuNER* dataset, which provided an additional benchmark for evaluation. Consequently, we assessed various deep-learning models for the NER task using both datasets.

Our analyses revealed an obvious divergence in outcomes between our *ReviewsDB* dataset and the benchmark *MenuNER* dataset, underscoring the crucial role dataset selection holds in influencing model results. Leading performance metrics were predominantly drawn from the *MenuNER* dataset, with the 1BiGRU(100)-CRF model achieving a peak F1-score of 93.07 with *glove-twitter-100* embeddings. This underscores *MenuNER*'s enhanced suitability for evaluating sophisticated modeling techniques. On the other hand, the F1-score achieved on the *ReviewsDB* dataset reached a peak of 82.40 using the 2BiGRU(75)-1CNN-CRF model, especially when paired with *glove-twitter-25* or *glove-twitter-100* embeddings. This performance contrast between models trained on *MenuNER* and *ReviewsDB* can be traced back to several factors. Firstly, the large size of the *MenuNER* dataset makes it more expansive than *ReviewsDB*. Additionally, the specific focus of *MenuNER* on Indian cuisine, with its unique patterns such as naan, lamb, tandoori, masala, and curry, potentially simplifies the task for the NER model. On the other hand, *ReviewsDB* offers a broader spectrum of food and beverage terms, ranging from ice and coffee to beer, donuts, drinks, fries, and pizza. Further complicating the NER task is the variability in annotation segment lengths in the *ReviewsDB* dataset. Longer dish names, like "ice cream with waffle", are notably more frequent in *ReviewsDB* than in *MenuNER*. In essence, the inherent attributes and intricacies of each dataset play a crucial role in the resultant performance differentials.

- **RQ2:** *How does domain adaptation influence the performance of a model trained on a recipe dataset for food name detection when applied to the domain of dish name detection in customer reviews?*

Limitations in domain adaptation: To examine the efficiency of domain adaptation in identifying dish names within customer reviews, we replicated a baseline experiment by Cenikj et al. (2020). In this replication, a BiLSTM-CRF model was first trained on the gold-standard food recipes dataset, *FoodBase*, and subsequently evaluated on a customer review dataset to detect dish names.

The model exhibited a marked reduction in performance upon transferring knowledge from general food-related NER tasks to the specific task of detecting dish names in customer reviews. A sharp drop of nearly 50% in the average F1 score clearly highlights the challenges of domain adaptation, even when transitioning between seemingly similar domains.

- **RQ3:** *How does the choice of pre-trained word embeddings like Glove, Word2Vec, and FastText influence the effectiveness of NER models designed for dish name identification?*

Impact of pre-trained word embeddings: Multiple experiments to assess the performance of various named entity recognition (NER) models using different word embeddings were conducted. The experiments utilized a range of word embeddings, including *Glove*, *Word2Vec*, *FastText*, and *BERT* variants.

Our research shows that the choice of word embeddings played a key role in the models' efficiency. Amongst the various embeddings tested, GloVe surfaced as the most effective, highlighting the importance of the selection and implementation of these embeddings in model development. This observation is verified by parallel experiments conducted on both the *ReviewsDB* and *MenuNER* datasets, wherein models leveraging different GloVe variants consistently outperformed their counterparts. It's imperative to

acknowledge that the commendable performance of these pre-trained word embeddings can be attributed to their additional fine-tuning during the NER model's training phase.

- **RQ4:** *How can various neural network architectures, including BiLSTM, BiGRU, and CNN, be effectively utilized in developing a computationally efficient deep learning NER model for the accurate detection of dish names in customer reviews compared to the computationally expensive state of art model?*

Optimal performance with simplified architecture: Our experimental journey began with the baseline architecture BiLSTM(50N)-CRF as a foundation for our dish detection tasks. As our research progressed, we explored several modifications, from tweaking the number of neurons in the hidden layers to experimenting with different optimization algorithms. We also attempted alternative architectures, introducing BiGRU and CNN layers. Cumulatively, these iterations gave rise to 53 distinct deep-learning models. Each model underwent thorough evaluation using 5-fold cross-validation techniques on both the *MenuNER* and *ReviewsDB* datasets.

Concurrently, understanding the balancing act between computational resources and performance led us to compare the outcomes of simplified deep learning architectures against their more intricate, resource-expensive counterparts. Although we were unable to replicate the intensive model proposed by Syed and Chung (2021) due to computational constraints, we experimented with a range of simplified models.

Notably, a simple deep-learning model featuring a singular BiGRU layer with just 100 hidden neurons, coupled with a CRF layer (denoted as 1BiGRU(125N)-CRF), showcased exemplary performance. This model attained a commendable F1-score of 93.07 for dish name

detection on the *MenuNER* dataset when utilizing *glove-twitter-100*-word embeddings. This achieved F1- score outperformed the recorded F1-score for the same NER task for dish name detection which is 91.30%. achieved by Syed and Chung (2021).

Additionally, our research demonstrates a noteworthy balance between model complexity and performance in NER endeavors. For example, our top-performing simple architecture like 1BiGRU, containing a modest count of trainable parameters (around 1 million), still delivered robust F1 scores. Such findings challenge the premise of resorting to complex, resource-intensive architectures, illustrated by models like the one by Syed and Chung (2021) which boasts a staggering 40 million parameters.

- **RQ5:** *What are the implications of employing contextualized embeddings like BERT for Named Entity Recognition in the specific area of dish name detection? Does it offer a significant advantage over non-contextualized embeddings?*

Evaluation of BERT embeddings: With the intention of delving into the possible benefits of BERT-based contextual embeddings, we conducted a series of experiments. These centered on generating both cased and uncased BERT embeddings for the *MenuNER* customer review dataset. Broadening our investigative scope, we combined these embeddings in two distinct strategies: by averaging and concatenating. The primary goal of these trials was to study the potential enhancement BERT embeddings might bring to the models.

Contrary to expectations, our analyses surrounding the bert-based contextual embeddings revealed that their incorporation failed to render significant performance advantages for our task in comparison to other pre-trained word embeddings. The results suggest that the high performance of BERT embeddings might necessitate computationally

intensive models, thereby raising questions regarding their utility in specific NER tasks, especially when comparable performance can be achieved with simpler architectures.

5.3 Research Contributions

In this section, we delve into the significant contributions of our research. These contributions are categorized into two distinct yet interconnected realms: Theoretical Contributions and Practical Implications. Each category addresses different facets of our research's impact.

5.3.1 Theoretical Contributions

The following are the advancements and innovations brought forth by our study in the academic domain of Named Entity Recognition:

- *Development of ReviewsDB dataset:* I developed a dataset called *ReviewsDB* using 1,000 restaurant reviews from Yelp and marked mentions of dish names myself. This dataset is important for the study. I used it to test how well different deep-learning models can identify dish names. I compared its results with another dataset, *MenuNER*, to understand the named entity recognition (NER) process. Both datasets helped train and check various deep-learning models. I believe *ReviewsDB* will be useful for upcoming studies in this area.
- *Advancement in NER model performance:* I developed a deep learning model, named 1BiGRU(100)-CRF, which notably outperformed other models in the task of identifying dish names from customer reviews. Specifically, on the *MenuNER* dataset, our model achieved an F1 score of 93.07. This surpasses the previous best

result of 92.50 as documented by Syed and Chung (2021) using the same dataset. What makes our model stand out is not just its superior accuracy, but also its simplicity. Unlike the state of art model which demands a lot of computational power, ours is streamlined and efficient. This means it can run effectively even on systems with limited resources, making it practical for real-world applications.

- *Comprehensive comparative analysis of deep-learning models:* My research didn't stop at just one model. I carried out a thorough study comparing various deep-learning models, each with different layers and embeddings. Notably, we didn't limit our evaluation to just one dataset; we extended it to two, allowing a more comprehensive understanding of how these models function across different sets of data. This approach gave us insights into the factors that play a key role in the success of systems aimed at detecting dish names in customer reviews.

5.3.2 Practical Implications

The following are the real-world applications of our research findings, particularly their significance in the restaurant and food industry:

- *Enhanced Efficiency and Accessibility:* The simplified and efficient nature of our 1BiGRU(100)-CRF model makes it practically applicable, even in businesses with limited computational resources, widening its usability.
- *Impact on the restaurant and food industry:* The application of our NER system in automating the extraction of dish names from customer reviews has

significant practical implications, such as improving data analytics for menu planning, inventory management, and marketing strategies in the restaurant industry.

- *Potential for enhanced customer satisfaction:* By enabling businesses to respond more effectively to customer feedback, our research paves the way for enhanced customer satisfaction, offering a direct benefit to the restaurant and food industry.

5.4 Limitations

While the present research has paved the way for several insightful outcomes, it is essential to acknowledge the limitations inherent in our study:

1. Although both *ReviewsDB* and *MenuNER* proved valuable for our experiments, future work could benefit from even broader and more diverse datasets. It's worth noting that *ReviewsDB* was created through my initiative, a personal effort, which may lead to questions about annotation consistency when compared to other datasets. Also, the diversity on datasets highlighted importance of the extraction of customer reviews from different cuisines order to have better generalization when we apply the trained models on restaurant customer reviews.
2. Our models have been tailored for the task of recognizing dish names within restaurant reviews. Their performance might not be as consistent when applied to different domains or more extensive contexts.
3. There were notable differences in how models performed on the *MenuNER* and *ReviewsDB* datasets. A more granular investigation, perhaps involving

error analysis, could provide insights into why certain models struggled more with the *ReviewsDB* dataset.

4. The deep learning architectures of the models, especially those enhanced with Convolutional Neural Network (CNN) layers. While these models have shown promising results in recognizing dish names, there remains intact potential in optimizing their performance. The limitation primarily stems from the restricted scope of hyperparameter tuning. Due to constraints in computational resources and time, the exploration of hyperparameters such as layer depths, filter sizes, and learning rates was not as exhaustive as it could have been. This limited tuning might have prevented the models from achieving their maximum effectiveness and efficiency in dish name recognition tasks.

5. The computational constraints prevented us from replicating the experiments on the models that heavily relied on BERT embeddings.

5.5 Future Work

Building upon the findings of the present research, there are multiple avenues for future exploration:

1. To create universally relevant models, future studies could curate more extensive datasets that encompass reviews from varied sources, regions, and cuisines. This would allow for a richer and more comprehensive understanding of customer perspectives. A key area for future research is the application of semi-supervised learning techniques to expand and enhance the annotation of customer reviews for dish

names. This approach could alleviate the limitations of dataset size, leveraging a mix of labeled and unlabeled data to improve model generalization and accuracy.

2. Future studies should build on the methodologies and findings of this thesis to extend the applicability of the models to additional food-related domains. Potential areas of expansion include analyzing grocery product reviews, food delivery service feedback, and the broader spectrum of hospitality industry reviews. These avenues promise to further validate and enhance the generalizability of the current models, leveraging the foundational work presented herein.

3. The combination of various successful architectures could result in hybrid models boasting superior performance. Delving deeper into different configurations and layer combinations in deep learning models might reveal optimal structures for dish name recognition tasks including deeper BiLSTM stacks, and transformer-based architectures like BERT-BiLSTM-CRF.

4. An in-depth and comprehensive hyperparameter search holds the potential to significantly refine and enhance the models, particularly those incorporating CNN layers. Future studies could focus on systematically experimenting with various parameter combinations, exploring a wider range of layer configurations, filter dimensions, and learning rates. Such exhaustive experimentation could unveil optimal settings that lead to marked improvements in the accuracy and efficiency of dish name recognition models.

5. The exploration of ensemble methods has a significant opportunity for future work given that a variety of model architectures were implemented and tested

during this research. The combination of predictions from the top-performing models represents a logical next step to potentially boost the NER system's capabilities.

6. Future research should investigate the application and fine-tuning of pre-trained Large Language Models (LLMs) for dish name Named Entity Recognition tasks. This exploration would assess the efficiency and effectiveness of LLMs in reducing data requirements and enhancing model performance, offering a promising direction for advancing the field. Subsequent studies could involve a comparative analysis of different contextualized embeddings, like ELMo, GPT, and others, to potentially uncover more nuanced advantages in specific NER scenarios

7. As real-time feedback becomes more crucial, especially on platforms like mobile apps or review websites, research should prioritize creating efficient, yet powerful lightweight models tailored for immediate applications.

8. Pairing NER with sentiment analysis can offer a complete picture of a dish's perception. Such an integrated approach would be invaluable, providing valuable insights into customer preferences and critiques, which in turn could serve as pivotal feedback for restaurants and culinary professionals.

In conclusion, our exploration into the domain of named entity recognition, specifically in the context of dish name detection, has shed light on several critical aspects that have been previously overlooked. As we move forward, it is essential to continue probing, experimenting, and refining our approaches, ensuring that the models we develop are not only accurate but also practical and efficient in real-world applications.

References

- Abdulkadirov, R., Lyakhov, P., & Nagornov, N. (2023). Survey of optimization algorithms in modern neural networks. *Mathematics*, 11(11), 2466-2503.
- Alfred, R., Leong, L. C., On, C. K., & Anthony, P. (2014). Malay named entity recognition based on rule-based approach. *International Journal of Machine Learning and Computing*, 4(3), 300-306.
- Ali, N., Alshahrani, A., Alghamdi, A., & Novikov, B. (2022). Smart tips: online products recommendations system based on analyzing customers reviews. *Applied Sciences*, 12(17), 8823-8838.
- Ali, M. N., Tan, G., & Hussain, A. (2018). Bidirectional recurrent neural network approach for Arabic named entity recognition. *Future Internet*, 10(12), 123-134.
- Al-Smadi, M., Al-Zboon, S., Jararweh, Y., & Juola, P. (2020). Transfer learning for Arabic named entity recognition with deep neural networks. *IEEE Access*, 8, 37736-37745.
- Amplayo, R., Bražinskas, A., Suhara, Y., Wang, X., & Liu, B. (2022). Beyond opinion mining: summarizing opinions of customer reviews. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM. <https://doi.org/10.1145/3477495.3532676>
- Ariyasriwatana, W., & Quiroga, L. M. (2016). A thousand ways to say 'delicious!' - categorizing expressions of deliciousness from restaurant reviews on the social network site Yelp. *Appetite*, 104, 18-32.
- Asghari, M., Sierra-Sosa, D., & Elmaghraby, A. S. (2022). BINDER: a low-cost biomedical named entity recognition. *Information Sciences*, 602, 184-200.

- Askalidis, G., Kim, S., & Malthouse, E. (2017). Understanding and overcoming biases in online review systems. *Decision Support Systems*, 97, 23-30.
- Asudani, D. S., Nagwani, N. K., & Singh, P. (2023). Impact of word embedding models on text analytics in deep learning environment: a review. *Artificial Intelligence Review*, 56, 10345-10425.
- Augustyniak, Ł., Kajdanowicz, T., & Kazienko, P. (2021). Comprehensive analysis of aspect term extraction methods using various text embeddings. *Computer Speech & Language*, 69, 101217-101236.
- Bai, S., Zheng, X., Han, C., & Bi, X. (2023). Exploring user-generated content related to vegetarian customers in restaurants: an analysis of online reviews. *Frontiers in Psychology*, 13, 1043844-1043859.
- Bam, S. B., & Shahi, T. B. (2014). Named entity recognition for Nepali text using support vector machine. *Intelligent Information Management*, 6(2), 21-29.
- Berard, A., Calapodescu, I., Dymetman, M., Roux, C., Meunier, J., & Nikoulina, V. (2019). Machine translation of restaurant reviews: new corpus for domain adaptation and robustness. In *Proceedings of the 3rd Workshop on Neural Generation and Translation* (pp. 168–176).
- Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5, 135-146.
- Boulos, M., Yassine, A., Shirmohammadi, S., Namahoot, C., & Brückner, M. (2015). Towards an “internet of food”: food ontologies for the internet of things. *Future Internet*, 7(4), 372-392.

- Cai, X., Dong, S., & Hu, J. (2019). A deep learning model incorporating part of speech and self-matching attention for named entity recognition of Chinese electronic medical records. *BMC Medical Informatics and Decision Making*, 19(2), 101-109.
- Castellano-Escuder, P., González-Domínguez, R., Wishart, D. S., Andrés-Lacueva, C., & Sánchez-Pla, A. (2020). FOBI: an ontology to represent food intake data and associate it with metabolomic data. *Database*, 2020, 1-8.
- Cenikj, G., Petelin, G., Seljak, B. K., & Eftimov, T. (2022). SciFoodNER: food named entity recognition for scientific text. In *2022 IEEE International Conference on Big Data (Big Data)* (pp. 4065-4073). IEEE.
- Cenikj, G., Popovski, G., Stojanov, R., Seljak, B. K., & Eftimov, T. (2020). BuTTER: bidirectional LSTM for food named-entity recognition. In *2020 IEEE International Conference on Big Data (Big Data)* (pp. 3550-3556). IEEE.
- Chen, D., Zhang, D., & Liu, A. (2019). Intelligent Kano classification of product features based on customer reviews. *CIRP Annals - Manufacturing Technology*, 68(1), 149–152.
- Chenze, B., Lee, E., & Panangadan, A. (2022). Iterative approach for novel entity recognition of foods in social media messages. In *2022 IEEE 23rd International Conference on Information Reuse and Integration for Data Science (IRI)* (pp. 126-131). IEEE.
- Chieu, H. L., & Ng, H. T. (2002). Named entity recognition: a maximum entropy approach using global information. In *Proceedings of the 19th International Conference on Computational Linguistics* (pp. 1-7). <https://doi.org/10.3115/1072228.1072253>

- Chieu, H. L., & Ng, H. T. (2003). Named entity recognition with a maximum entropy approach. In *Proceedings of the 7th Conference on Natural Language Learning and Natural Language Processing* (pp. 160-163).
- Cho, K., Merriënboer, B., Bahdanau, D., & Bengio, Y. (2014a). On the properties of neural machine translation: encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation* (pp. 103–111). Association for Computational Linguistics.
- Cho, K., Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014b). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1724-1734). Association for Computational Linguistics.
- Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning* (pp. 1-9).
- Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12, 2493-2537.
- Deniz, E., Erbay, H., & Coşar, M. (2022). Multi-label classification of e-commerce customer reviews via machine learning. *Axioms*, 11, 1-17.
<https://doi.org/10.3390/axioms111208823>
- Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019*

Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Vol. 1, pp. 4171-4186).

Diwan, N., Batra, D., & Bagler, G. (2020). A named entity-based approach to model recipes.

In *2020 IEEE 36th International Conference on Data Engineering Workshops (ICDEW)* (pp. 88-93). IEEE.

Donnelly, K. (2006). SNOMED-CT: the advanced terminology and coding system for eHealth. *Studies in Health Technology and Informatics*, 121, 279-290.

Eftimov, T., Seljak, B. K., & Korošec, P. (2017). A rule-based named-entity recognition method for knowledge extraction of evidence-based dietary recommendations. *PLOS One*, 12(6). <https://doi.org/10.1371/journal.pone.0179488>

Ekbal, A., & Bandyopadhyay, S. (2010). Named entity recognition using support vector machine: A language-independent approach. *International Journal of Electronics, Communications, and System Engineering*, 4(2), 155-170.

Etzioni, O., Cafarella, M., Downey, D., Popescu, A. M., Shaked, T., Soderland, S., Weld, D. S., & Yates, A. (2005). Unsupervised named-entity extraction from the web: An experimental study. *Artificial Intelligence*, 165, 91-134.

Fan, Y., & Hunter, A. (2023). Understanding the cooking process with English recipe text. In *Findings of the Association for Computational Linguistics: ACL 2023* (pp. 4244-4264).

Fellbaum, C. (2005). WordNet and wordnets. In *Encyclopedia of Language and Linguistics* (2nd ed., pp. 665-670). Elsevier.

Finkel, J. R., Grenager, T., & Manning, C. (2005). Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL '05)* (pp. 363-370).

- FoodDB. (n.d.). FoodDB: The food database. FoodDB version 1.0. <https://foodb.ca/>
- Friedrich, S., Groll, A., Ickstadt, K., Kneib, T., Pauly, M., Rahnenführer, J., & Friede, T. (2023). Regularization approaches in clinical biostatistics: A review of methods and their applications. *Statistical Methods in Medical Research*, 32(2), 425-440.
- Goh, T. T., Jamaludin, N. A. A., Mohamed, H., Ismail, M. N., & Chua, H. S. (2022). A comparative study on part-of-speech taggers' performance on examination questions classification according to Bloom's taxonomy. In *Journal of Physics: Conference Series*, 2224(1), 1-8. IOP Publishing.
- Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.
<http://www.deeplearningbook.org>
- Goyal, A., Gupta, V., & Kumar, M. (2021). Recurrent neural network-based model for named entity recognition with improved word embeddings. *IETE Journal of Research*.
<https://doi.org/10.1080/03772063.2021.2006805>
- Grandini, M., Bagli, E., & Visani, G. (2020). Metrics for multi-class classification: An overview. *arXiv*. <https://doi.org/10.48550/arXiv.2008.05756>
- Gridach, M., & Haddad, H. (2018). Arabic named entity recognition: A bidirectional GRU-CRF approach. In *International Conference on Computational Linguistics and Intelligent Text Processing* (Vol. 10761, pp. 264–275). Springer.
- Griffiths, E., Brinkman, F., Dooley, D., Hsiao, W., Buttigieg, P., & Hoehndorf, R. (2016). FoodON: A global farm-to-fork food ontology. In *International Conference of Biomedical Ontology* (pp. 1-2).
- Güngör, O., Güngör, T., & Üsküdarlı, S. (2018). The effect of morphology in named entity recognition with sequence tagging. *Natural Language Engineering*, 25(1), 147-169.

- Haytowitz, D., & Pehrsson, P. (2018). USDA's national food and nutrient analysis program (NFNAP) produces high-quality data for USDA food composition databases: Two decades of collaboration. *Food Chemistry*, 238, 134-138.
- Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv*.
<https://doi.org/10.48550/arXiv.1207.0580>
- Hinton, G., Srivastava, N., & Swersky, K. (2012). Lecture 6.5: RMSprop: Divide the gradient by a running average of its recent magnitude.
http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf
- Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735-1780.
- Howard, J., & Ruder, S. (2018). Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics* (Vol. 1, pp. 328–339).
- Huang, Z., Xu, W., & Yu, K. (2015). Bidirectional LSTM-CRF models for sequence tagging. *arXiv*. <https://doi.org/10.48550/arXiv.1508.01991>
- Jia, S. (2019). Measuring tourists' meal experience by mining online user-generated content about restaurants. *Scandinavian Journal of Hospitality and Tourism*, 19(4-5), 371-389.
- Jiao, X., Yin, Y., Shang, L., Jiang, X., Chen, X., Li, L., Wang, F., & Liu, Q. (2020). TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020* (pp. 4163-4174). Association for Computational Linguistics.

- Jonquet, C., Shah, N., Youn, C., Musen, M., Callendar, C., & Storey, M. A. (2009). NCBO annotator: Semantic annotation of biomedical data. In *International Semantic Web Conference, Poster, and Demo Session* (Vol. 110, pp. 1-3).
- Justus, D., Brennan, J., Bonner, S., & McGough, A. S. (2018). Predicting the computational cost of deep learning models. In *2018 IEEE International Conference on Big Data (Big Data)* (pp. 3873-3882). IEEE.
- Khan, W., Daud, A., Alotaibi, F., Aljohani, N., & Arafat, S. (2020). Deep recurrent neural networks with word embeddings for Urdu named entity recognition. *ETRI Journal*, 42(1), 90-100.
- Khan, W., Daud, A., Shahzad, K., Amjad, T., Banjar, A., & Fasihuddin, H. (2022). Named entity recognition using conditional random fields. *Applied Sciences*, 12(13), 6391-6409. <https://doi.org/10.3390/app12136391>
- Khurana, D., Koli, A., Khatter, K., & Singh, S. (2023). Natural language processing: State of the art, current trends and challenges. *Multimedia Applications*, 82(3), 3713-3744.
- Kim, Y., Jernite, Y., Sontag, D., & Rush, A. (2016). Character-aware neural language models. In *Proceedings of the 13th AAAI Conference on Artificial Intelligence* (Vol. 30(1), pp. 2741-2749).
- Kingma, D., & Ba, J. (2015). Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, San Diego, CA, USA (pp. 1-15).
- Komariah, K. S., Purnomo, A. T., Satriawan, A., Hasanuddin, M. O., Setianingsih, C., & Sin, B. K. (2023). SMPT: A semi-supervised multi-model prediction technique for food ingredient named entity recognition (FINER) dataset construction. *Informatics*, 10(1), 1-19.

- Komariah, K. S., & Sin, B. K. (2022). Enhancing food ingredient named-entity recognition with recurrent network-based ensemble (RNE) model. *Applied Sciences*, 12(20), 1-19. <https://doi.org/10.3390/app122010310>
- Kwon, W., Lee, M., & Back, K. (2020). Exploring the underlying factors of customer value in restaurants: A machine learning approach. *International Journal of Hospitality Management*, 91. <https://doi.org/10.1016/j.ijhm.2020.102643>
- Lafferty, J., McCallum, A., & Pereira, F. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning* (pp. 282-289).
- Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 260-270).
- Lee, E., Chenze, B., & Panangadan, A. (2021). Encouraging sustainability practices through entity recognition of food items on social media. In *2021 IEEE 22nd International Conference on Information Reuse and Integration for Data Science (IRI)* (pp. 263-266). IEEE.
- Lee, K. J., Hwang, Y. S., Kim, S., & Rim, H. C. (2004). Biomedical named entity recognition using two-phase model based on SVMs. *Journal of Biomedical Informatics*, 37(6), 436-447.
- Li, L., Jin, L., & Huang, D. (2015). Exploring recurrent neural networks to detect named entities from biomedical text. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data: 14th China National*

- Conference, CCL 2015 and Third International Symposium, NLP-NABD 2015, Proceedings* (Vol. 14, pp. 279-290). Springer International Publishing.
- Li, J., Sun, A., Han, J., & Li, C. (2020). A survey on deep learning for named entity recognition. *IEEE Transactions on Knowledge and Data Engineering*, 34(1), 50-70.
- Ma, X., & Hovy, E. (2016). End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* (Vol. 1, pp. 1064–1074). Association for Computational Linguistics.
- Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., & Galstyan, A. (2021). A survey on bias and fairness in machine learning. *ACM Computing Surveys (CSUR)*, 54(6), 1-35.
- Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013a). Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, Workshop Track Proceedings* (pp. 1-14).
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems* (Vol. 26, pp. 1-9).
- Morwal, S., Jahan, N., & Chopra, D. (2012). Named entity recognition using hidden Markov model (HMM). *International Journal of Natural Language Computing*, 1(4), 15-23.
- Nadeau, D., & Sekine, S. (2007). A survey of named entity recognition and classification. *Linguisticae Investigationes*, 30(1), 3-26.
- Nadeau, D., Turney, P. D., & Matwin, S. (2006). Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity. In *Advances in Artificial Intelligence:*

19th Conference of the Canadian Society for Computational Studies of Intelligence
(pp. 266-277). Springer Berlin Heidelberg.

NLTK. (2023). WordNet Interface. <https://www.nltk.org/howto/wordnet.html>

Özen, İ., & Katlav, E. (2023). Aspect-based sentiment analysis on online customer reviews: A case study of technology-supported hotels. *Journal of Hospitality and Tourism Technology*, 14(2), 102-120.

Paliouras, G., Karkaletsis, V., Petasis, G., & Spyropoulos, C. (2000). Learning decision trees for named-entity recognition and classification. In *Proceedings of the ECAI Workshop on Machine Learning for Information Extraction* (pp. 1-7).

Panchendrarajan, R., & Amaresan, A. (2018). Bidirectional LSTM-CRF for named entity recognition. In *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation* (pp. 531-540).

Patil, N., Patil, A., & Pawar, B. (2020). Named entity recognition using conditional random fields. *Procedia Computer Science*, 167(2020), 1181-1188.

Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1532-1543).

Perera, N., Nguyen, T. T. L., Dehmer, M., & Emmert-Streib, F. (2022). Comparison of text mining models for food and dietary constituent named-entity recognition. *Machine Learning and Knowledge Extraction*, 4(1), 254-275.

Petasis, G., Vichot, F., Wolinski, F., Paliouras, G., Karkaletsis, V., & Spyropoulos, C. D. (2001). Using machine learning to maintain rule-based named-entity recognition and

- classification systems. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics* (pp. 426-433).
- Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (Vol. 1, pp. 2227-2237). Association for Computational Linguistics.
- Peters, K., & Remaud, H. (2020). Factors influencing consumer menu-item selection in a restaurant context. *Food Quality and Preference*, 82, 103887.
<https://www.sciencedirect.com/science/article/pii/S0950329319303726>
- Popovski, G., Kochev, S., Seljak, B. K., & Eftimov, T. (2019). FoodIE: A rule-based named-entity recognition method for food information extraction. In *Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2019)* (pp. 915–922).
- Popovski, G., Seljak, B. K., & Eftimov, T. (2019). FoodBase corpus: A new resource of annotated food entities. *Database*, 2019, 1-13.
- Popovski, G., Seljak, B., & Eftimov, T. (2020). A survey of named-entity recognition methods for food information extraction. *IEEE Access*, 8, 31586-31594.
- Prechelt, L. (2002). Early stopping-but when?. In *Neural Networks: Tricks of the Trade* (pp. 55-69). Springer Berlin Heidelberg.
- Princeton University. (2010). About WordNet. WordNet. <https://wordnet.princeton.edu/>
- Qamar, Z. (2018). Foodkeeper. *Journal of Nutrition Education and Behavior*, 50(1), 101–101.

- Qiu, Q., Xie, Z., Wu, L., Tao, L., & Li, W. (2019). BiLSTM-CRF for geological named entity recognition from the geoscience literature. *Earth Science Informatics*, 12(4), 565-579.
- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8), 1-24.
- RaRe-Technologies. (2018). gensim-data. <https://github.com/RaRe-Technologies/gensim-data>
- Raschka, S. (2018). Model evaluation, model selection, and algorithm selection in machine learning. *arXiv*. <https://arxiv.org/abs/1811.12808>
- Reimers, N., & Gurevych, I. (2017). Optimal hyperparameters for deep LSTM networks for sequence labeling tasks. *arXiv*. <https://arxiv.org/abs/1707.06799>
- Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT: A distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv*. <https://arxiv.org/abs/1910.01108>
- Settles, B. (2004). Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications (NLPBA/BioNLP)* (pp. 104-107).
- Shaalán, K., & Raza, H. (2009). NERA: Named entity recognition for Arabic. *Journal of The American Society for Information Science and Technology*, 60(8), 1652-1663.
- Shah, D. N., & Bhadka, H. B. (2017). Named entity recognition from Gujarati text using rule-based approach. In *Proceedings of the International Conference on Intelligent System Design and Applications* (pp. 797-805). Springer.
- Sharma, K., Mandapati, K. V., Pattekar, M. C., Kumar, K. K. T., & Srinivasa, G. (2023). Food recommendation system based on collaborative filtering and taste profiling. In 2023

3rd International Conference on Intelligent Technologies (CONIT) (pp. 1-6).

<https://doi.org/10.1109/CONIT59222.2023.10205379>

Simran, K., Sriram, S., Vinayakumar, R., & Soman, K. P. (2020). Deep learning approach for intelligent named entity recognition of cyber security. In *Advances in Signal Processing and Intelligent Recognition Systems* (pp. 163-172). Springer.

Sornlertlamvanich, V., & Yuenyong, S. (2022). Thai named entity recognition using biLSTM-CNN-CRF enhanced by TCC. *IEEE Access*, 10, 53043-53052.

Smădu, R. A., Dinică, I. R., Avram, A. M., Cercel, D. C., Pop, F., & Cercel, M. C. (2022). Legal named entity recognition with multi-task domain adaptation. In *Proceedings of the Natural Legal Language Processing Workshop 2022* (pp. 305-321).

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1), 1929-1958.

Stenetorp, P., Pyysalo, S., Topić, G., Ohta, T., Ananiadou, S., & Tsujii, J. I. (2012). BRAT: A web-based tool for NLP-assisted text annotation. In *Proceedings of the Demonstrations at the 13th Conference of the European Chapter of the Association for Computational Linguistics* (pp. 102-107).

Stojanov, R., Popovski, G., Cenikj, G., Seljak, B. K., & Eftimov, T. (2021). A fine-tuned bidirectional encoder representation from transformers model for food named-entity recognition: Algorithm development and validation. *Journal of Medical Internet Research*, 23(8), e28229. <https://www.jmir.org/2021/8/e28229>

Stojanov, R., Popovski, G., Jofce, N., Trajanov, D., Seljak, B. K., & Eftimov, T. (2020). Foodviz: Visualization of food entities linked across different standards. In *Machine*

Learning, Optimization, and Data Science: 6th International Conference, LOD 2020, Siena, Italy, July 19–23, 2020 (pp. 28-38). Springer International Publishing.

Strubell, E., Verga, P., Belanger, D., & McCallum, A. (2017). Fast and accurate entity recognition with iterated dilated convolutions. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing* (pp. 2670-2680).

Sun, S., Cao, Z., Zhu, H., & Zhao, J. (2020). A survey of optimization methods from a machine learning perspective. *IEEE Transactions on Cybernetics*, 50(8), 3668-3681.

Sun, S., Jiang, F., Feng, G., Wang, S., & Zhang, C. (2022). The impact of COVID-19 on hotel customer satisfaction: Evidence from Beijing and Shanghai in China. *International Journal of Contemporary Hospitality Management*, 34, 382-406.

Sun, J., Liu, Y., Cui, J., & He, H. (2022). Deep learning-based methods for natural hazard named entity recognition. *Scientific Reports*, 12(1), 1-15.

Sutskever, I., Martens, J., Dahl, G., & Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning* (Vol. 28, pp. 1139-1147).

Syed, M., & Chung, S. (2021). MenuNER: Domain-adapted BERT based NER approach for a domain with limited dataset and its application to food menu domain. *Applied Sciences*, 11(13). <https://doi.org/10.3390/app11136007>

Szarvas, G., Farkas, R., & Kocsor, A. (2006). A multilingual named entity recognition system using boosting and C4.5 decision tree learning algorithms. In *Proceedings of the International Conference on Discovery Science* (pp. 267-278). Springer.

- Szeghalmy, S., & Fazekas, A. (2023). A comparative study of the use of stratified cross-validation and distribution-balanced stratified cross-validation in imbalanced learning. *Sensors*, 23(4), Article 2333. <https://www.mdpi.com/1424-8220/23/4/2333>
- Tareq, A. (2019). Positive unlabelled learning to recognize dishes as named entity. Master's Thesis, The British University in Dubai. <https://bspace.buid.ac.ae/handle/1234/1459>
- Tian, Y., & Zhang, Y. (2022). A comprehensive survey on regularization strategies in machine learning. *Information Fusion*, 80, 146-166.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 1-15.
- Wang, Q., & Su, X. (2022). Research on named entity recognition methods in Chinese forest disease texts. *Applied Sciences*, 12(8). <https://www.mdpi.com/2076-3417/12/8/3885>
- Wang, J., Xu, B., & Zu, Y. (2021). Deep learning for aspect-based sentiment analysis. In *International Conference on Machine Learning and Intelligent Systems Engineering (MLISE)* (pp. 267-271). IEEE.
- Wróblewska, A., Kaliska, A., Pawłowski, M., Wiśniewski, D., Sosnowski, W., & Ławrynowicz, A. (2022). TASTeSet-Recipe dataset and food entities recognition benchmark. *arXiv*. <https://arxiv.org/abs/2204.07775>
- Xu, K., Yang, Z., Kang, P., Wang, Q., & Liu, W. (2019). Document-level attention-based biLSTM-CRF incorporating disease dictionary for disease named entity recognition. *Computers in Biology and Medicine*, 108, 122-132.
- Yang, Z., Salakhutdinov, R., & Cohen, W. (2016). Multi-task cross-lingual sequence tagging from scratch. *arXiv*. <https://doi.org/10.48550/arXiv.1603.06270>

- Yates, L. A., Aandahl, Z., Richards, S. A., & Brook, B. W. (2023). Cross validation for model selection: A review with examples from ecology. *Ecological Monographs*, 93(1).
<https://esajournals.onlinelibrary.wiley.com/doi/10.1002/ecm.1557>
- Yelp. (2018). Yelp dataset. <https://www.yelp.com/dataset/documentation/main>
- Yelp. (2019). Recommended reviews. https://www.yelp-support.com/Recommended_Reviews
- Yi, F., Liu, H., Wang, Y., Wu, S., Sun, C., Feng, P., & Zhang, J. (2023). Medical named entity recognition fusing part-of-speech and stroke features. *Applied Sciences*, 13(15), 8913-8923. <https://doi.org/10.3390/app13158913>
- Yu, B., Zhou, J., Zhang, Y., & Cao, Y. (2017). Identifying restaurant features via sentiment analysis on Yelp reviews. *arXiv*. <https://arxiv.org/abs/1709.08698>
- Yusup, F., Bijaksana, M., & Huda, A. (2019). Narrator's name recognition with support vector machine for indexing Indonesian hadith translations. *Procedia Computer Science*, 157, 191-198.
- Zhou, G., & Su, J. (2002). Named entity recognition using an HMM-based chunk tagger. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)* (pp. 473-480).
- Zhou, G., & Su, J. (2005). Machine learning-based named entity recognition via effective integration of various evidences. *Natural Language Engineering*, 11(2), 189-206.
- Zhang, S., & Elhadad, N. (2013). Unsupervised biomedical named entity recognition: Experiments with clinical and biological texts. *Journal of Biomedical Informatics*, 46(6), 1088-1098.
- Zhang, Y., & Wallace, B. (2017). A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification. In *Proceedings of the Eighth*

International Joint Conference on Natural Language Processing (Vol. 1, pp. 253–263).

Zhang, X., Zhao, J., & LeCun, Y. (2015). Character-level convolutional networks for text classification. *Advances in Neural Information Processing Systems*, 28, 1-9.

Appendices

Appendix A.

Part-of-Speech Tags (POS)

Each POS tag in the list represents a specific syntactic category or part of speech. The following are the meaning of each tag occurred in the dataset (Goh et al., 2022):

- **RB:** Adverb
- **VBP:** Verb, non-3rd person singular present
- **PDT:** Predeterminer
- **PRP:** Personal pronoun
- **VB:** Verb, base form
- **\$:** Dollar sign
- **"":** Closing quotation mark
- **NNPS:** Noun, proper, plural
- **CD:** Cardinal number
- **VBG:** Verb, gerund, or present participle
- **LS:** List item marker
- **UH:** Interjection
- **:** Colon
- **RBR:** Adverb, comparative
- **RP:** Particle
- **`:** Opening quotation mark
- **RBS:** Adverb, superlative

- **#:** Number sign
- **IN:** Preposition or subordinating conjunction
- **VBZ:** Verb, 3rd person singular present
- **POS:** Possessive ending
- **FW:** Foreign word
- **∴:** Sentence-final punctuation
- **VBD:** Verb, past tense
- **(:** Opening parenthesis
- **JJS:** Adjective, superlative
- **DT:** Determiner
- **NN:** Noun, singular or mass
- **JJR:** Adjective, comparative
- **NNS:** Noun, plural
- **JJ:** Adjective
- **,:** Comma
- **WRB:** Wh-adverb
- **MD:** Modal
- **WP:** Wh-pronoun
- **SYM:** Symbol
- **TO:** to
- **NNP:** Proper noun, singular
- **VBN:** Verb, past participle

-): Closing parenthesis
- **PRP\$**: Possessive pronoun
- **EX**: Existential there
- **CC**: Coordinating conjunction
- **WDT**: Wh-determiner

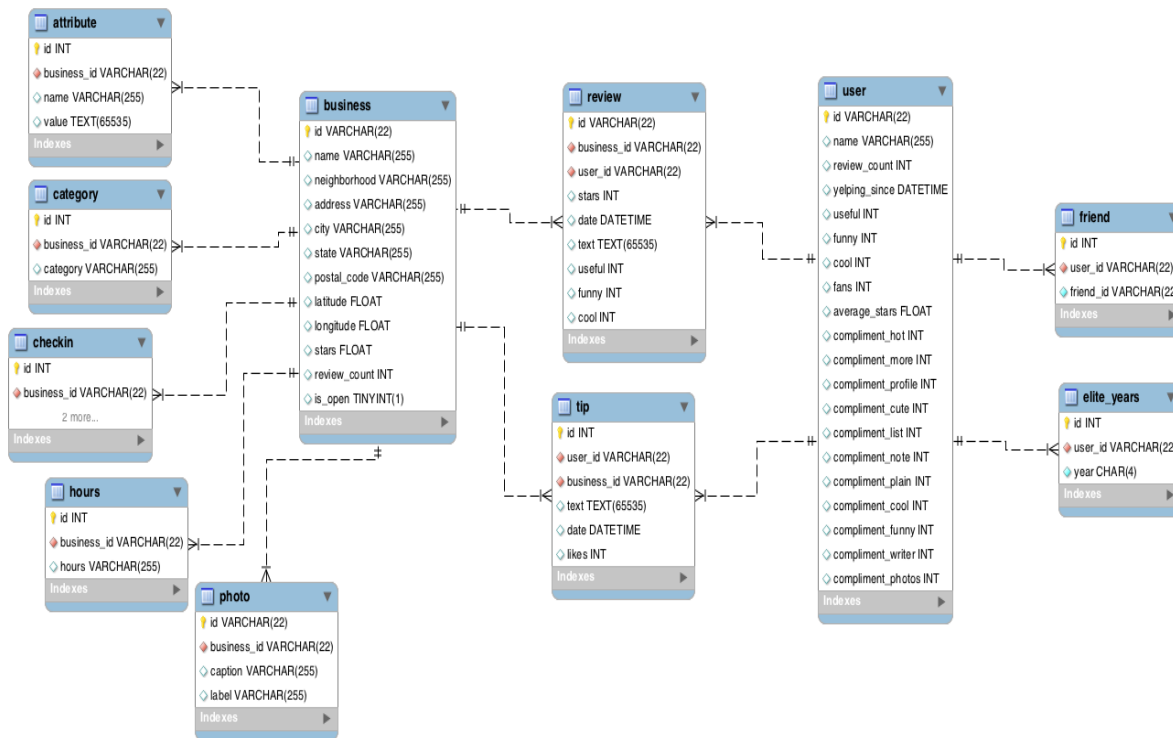
Appendix B.

Customer Reviews Extraction from Yelp

The obtained Yelp dataset consists of eleven SQL tables. The database schema is represented in Figure A-1 (Yelp, 2018).

Figure A-1

Yelp Database Schema



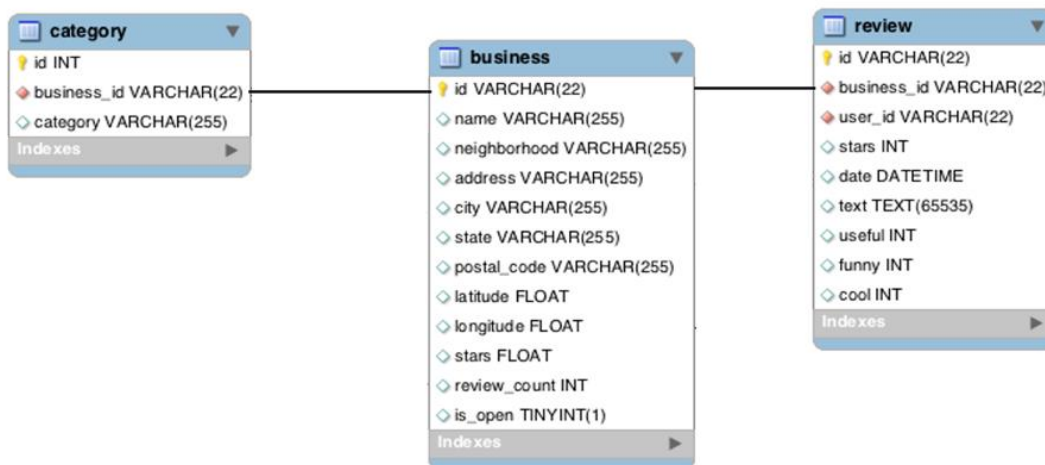
Note: The figure displays the framework that defines the organization, relationships of tables within the Yelp database. Adapted from “Yelp dataset”, 2018,

<https://www.yelp.com/dataset/documentation/main>. Copyright 2018 by the authors.

To extract the reviews pertaining to the food industry from the extensive Yelp dataset, we leverage a relational SQL database and use three tables to accomplish this task. These tables include the business, review, and category tables as shown in the below figure. With a vast collection of approximately five million reviews at our disposal, this systematic approach allows us to narrow down and focus specifically on the food-related reviews within the dataset.

Figure A-2

The Three Yelp Related Tables: Business, Category, and Review



Note: The figure adapted from “Yelp dataset”, 2018,

<https://www.yelp.com/dataset/documentation/main>. Copyright 2018 by the authors.

The description of the three tables and their attributes are shown in Table A-1, Table A-2, and Table A-3.

Table A-1.

Business Table in Yelp Dataset

Business Table	
Description	Contains business data including location, attributes, and categories.
Attributes	There are 12 attributes: • business_id: string, 22-character unique string business id. • name: string, the business's name • neighborhood: string, a nearby landmark • address: string, the full address of the business • city: string, the city • state: string, 2-character state code, if applicable • postal code: string, the postal code • latitude: float, latitude • longitude: float, longitude • stars: float, star rating, rounded to half-stars • review_count: integer, number of reviews. • is_open: integer, 0 or 1 for closed or open, respectively

Note: The table describes the structure of the Business table. Adapted from “Yelp dataset”, 2018, <https://www.yelp.com/dataset/documentation/main>. Copyright 2018 by the authors.

Table A-2*Review Table in Yelp Dataset*

Review Table	
Description	Contains full review text data including the user_id that wrote the review and the business_id the review is written for.
Attributes	There are nine attributes: • review_id: string, 22-character unique review id • user_id: string, 22-character unique user id • business_id: string, 22-character business id • stars: integer, star rating • date: string, date formatted YYYY-MM-DD • text: string, the review itself • useful: integer, number of useful votes received • funny: integer, number of funny votes received • cool: integer, number of cool votes received

Note: The table describes the structure of the Review table. Adapted from “Yelp dataset”, 2018, <https://www.yelp.com/dataset/documentation/main>. Copyright 2018 by the authors.

Table A-3*Category Table in Yelp Dataset*

Category Table	
Description	Contains the categories of each business.
Attributes	There are three attributes: • id: integer • business_id: string, 22 characters • category: string, 255 characters. Examples: restaurants, gym, health and medical, food, hair salon, etc.

Note: The table describes the structure of the Category table. Adapted from “Yelp dataset”, 2018, <https://www.yelp.com/dataset/documentation/main>. Copyright 2018 by the authors.

The 'category' table in the Yelp dataset serves as a repository for various business types, providing information about the nature of each business and its associated activities. Within this table, we discovered a total of 1,293 unique business categories. These categories encompass a wide range of businesses, such as nail salons, soup establishments, car rentals, cocktail bars, cafes, and pet-related services, among others.

To identify the categories relevant to the food industry, we conducted a thorough manual examination. This involved reading multiple reviews associated with each category from the 'review' table. It is worth noting that some businesses were assigned to multiple categories. For instance, a single business could be linked to six categories, including American cuisine, nightlife, bars, sandwiches, burgers, and restaurants.

Following the filtering process, we identified 61 food-related categories based on the baseline research. Although our Yelp dataset contained 226 categories related to food.

The 61 food related categories are: 'Food', 'Acai Bowls', 'Bagels', 'Bakeries', 'Beer', 'Wine & Spirits', 'Beverage Store', 'Breweries', 'Brewpubs', 'Bubble Tea', 'Butcher', 'CSA', 'Chimney Cakes', 'Cideries', 'Coffee & Tea', 'Coffee Roasteries', 'Convenience Stores', 'Cupcakes', 'Custom Cakes', 'Desserts', 'Distilleries', 'Do-It-Yourself Food', 'Donuts', 'Empanadas', 'Farmers Market', 'Food Delivery Services', 'Food Trucks', 'Gelato', 'Grocery', 'Honey', 'Ice Cream & Frozen Yogurt', 'Imported Food', 'International Grocery', 'Internet Cafes', 'Juice Bars & Smoothies', 'Kombucha', 'Organic Stores', 'Patisserie/Cake Shop', 'Piadina', 'Poke', 'Pretzels', 'Shaved Ice', 'Shaved Snow', 'Smokehouse', 'Specialty Food', 'Candy Stores', 'Cheese Shops', 'Chocolatiers & Shops', 'Fruits & Veggies', 'Health Markets', 'Herbs & Spices', 'Macarons', 'Meat Shops', 'Olive Oil', 'Pasta Shops', 'Popcorn Shops', 'Seafood Markets', 'Street Vendors', 'Tea Rooms', 'Water Stores', 'Wineries', 'Wine Tasting Room'.

To explore the Yelp dataset and extract the required data from it, **MySQL Database engine** and **MySQL Workbench** software are used to accomplish the task.

The following is a sample for a record in the ‘business’ table:

- **id** 2iTsRqUsPGRH1li1WVRvKQ
- **name** Carson Kitchen
- **neighborhood** Downtown
- **address** 124 S 6th St, Ste 100
- **city** Las Vegas
- **state** NV

- **postal_code** 89101
- **latitude** 36.1679
- **longitude** -115.141
- **stars** 4.5
- **review_count** 1586
- **is_open** 1

Also, a sample record obtained from the ‘review’ table is as follows:

- **id** ----X0BIDP9tA49U3RvdSQ
- **business_id** Ue6-WhXvI-_1xUIuapl0zQ
- **user_id** gVmUR8rqUFdbSeZbsg6z_w
- **stars** 4
- **date** 2014-02-17 00:00:00
- **text** Red, white and bleu salad was super yum and a great addition to the menu! This location was clean with great service and food served at just the right temps! Kids pizza is always a hit too with lots of great side dish options for the kiddos! When I'm on this side of town, this will definitely be a spot I'll hit up again!
- **useful** 1
- **funny** 0
- **cool** 0

And finally, sample records obtained from the ‘category’ table showing that one business is assigned to different categories as shown in Table A-4:

Table A-4*Sample Records from 'Category' Table*

# Id	Business_id	Category
8791	TdQw5Q6KZeJJSvc18u_SWg	Restaurants
8792	TdQw5Q6KZeJJSvc18u_SWg	Coffee & Tea
8793	TdQw5Q6KZeJJSvc18u_SWg	Food
8794	TdQw5Q6KZeJJSvc18u_SWg	Sandwiches
8795	TdQw5Q6KZeJJSvc18u_SWg	Breakfast & Brunch
8796	TdQw5Q6KZeJJSvc18u_SWg	Cafes

As shown above, some categories are generic like 'food' and some are more specific such as 'sandwiches'.

Utilizing the Yelp SQL relational database facilitated the whole retrieval and integration of data from multiple tables. An instance of this was accomplished through the process of performing an inner join between the 'Category' and 'Business' tables, enabling us to specifically obtain businesses associated with the food industry. Within the extensive list of 1,293 categories available in the Yelp database, we identified 61 categories that were directly relevant to food-related enterprises. Subsequently, we exclusively extracted food-related reviews from the 'Review' table, while also capturing supplementary details such as review ID, business star rating, business ID, and business name.

Appendix C.

Further Results and Analysis of Research Experiments

C.1 Further Analysis of the BiLSTM-CRF Model Results

This section provides a detailed analysis of the experimental results presented in Section 4.1.

C1.1 Further Exploration of Baseline Experiment Replication Outcomes

Table C-1 provides a comprehensive overview of performance metrics for both baseline models, with 'P' representing precision, 'R' indicating recall, and 'F1' representing the F1-score. The metrics are reported for the classes B-FOOD, I-FOOD, and O, across various word embedding models and model architectures.

Table C-1.

Precision, Recall, and F1 Scores of the Three Classes of the Two Baseline Models

Word Embedding	Model Architecture	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
GloVe	BiLSTM-CRF	0.87	0.85	0.86	0.81	0.79	0.80	0.98	0.99	0.98
	BiLSTM-CRF with	0.88	0.89	0.88	0.79	0.85	0.81	0.99	0.98	0.98
	CharEmbd									
Word2vec	BiLSTM-CRF	0.87	0.87	0.87	0.81	0.81	0.81	0.98	0.98	0.98
	BiLSTM-CRF with	0.87	0.92	0.89	0.78	0.88	0.83	0.99	0.98	0.99
	CharEmbd									
FastText	BiLSTM-CRF	0.85	0.89	0.87	0.73	0.86	0.79	0.99	0.98	0.98
	BiLSTM-CRF with	0.90	0.86	0.88	0.81	0.79	0.80	0.98	0.99	0.98
	CharEmbd									
Lexical- 300	BiLSTM-CRF	0.89	0.94	0.92	0.80	0.87	0.83	0.99	0.98	0.99
	BiLSTM-CRF with	0.91	0.93	0.92	0.83	0.87	0.85	0.99	0.99	0.99
	CharEmbd									

As evident from Table C-1, the *lexical-300* embedding model consistently outperforms other pre-trained models in most metrics when used with the BiLSTM-CRF model, except for the recall score for the I-FOOD class. This superiority may be attributed to the *lexical-300* model's capacity to learn embeddings during training, which enables it to gain a deeper understanding of the contextual nuances in the data.

Moreover, it is noteworthy that the precision of FastText embeddings notably increases from 0.85 to 0.90, while the recall of Word2Vec embeddings demonstrates a substantial improvement, rising from 0.87 to 0.92. Conversely, for the pre-trained models GloVe and Word2Vec, the precision for the I-FOOD class slightly decreases from 0.81 to 0.79 and 0.78, respectively. However, their recall values benefit from the incorporation of character embeddings, showing an increase from 0.79 to 0.85 for GloVe and from 0.81 to 0.88 for Word2Vec embeddings.

In the case of the *Lexical-300* model, the addition of character embeddings as input positively influences precision values for both the B-FOOD and I-FOOD classes, resulting in increases from 0.89 to 0.91 and from 0.80 to 0.83, respectively. However, this favorable impact does not extend to recall values, as the recall for the B-FOOD class experiences a slight decline from 0.94 to 0.93, while the recall for the I-FOOD class remains constant at 0.87. This observation suggests that the incorporation of character embeddings equips the model with an enhanced capacity to extract additional insights from the dataset, contributing to an overall improvement in performance.

Additionally, we examined the number of epochs required for both baseline models to converge. Notably, the *lexical-300*-word embeddings consistently demonstrated superior convergence performance. In contrast, the FastText embedding model required the highest number of epochs for convergence in both the BiLSTM-CRF and BiLSTM-CRF with CharEmbd models, as detailed in Table C-2:

Table C-2.*Average Number of Epochs for the Baseline Models*

Word Embedding	BiLSTM-CRF		BiLSTM-CRF with Char Embd	
	Baseline	Our Results	Baseline	Our Results
	Results		Results	
GloVe	155.2	187.4	125.2	181.2
Word2Vec	146.0	157	131.2	156
FastText	233.0	245	185.4	224
Lexical	130.4	136	118.8	136

It is essential to note that while our reported average epoch values may be higher than those presented in the original paper, they consistently align with the reported trends, indicating a consistent pattern of convergence in our results.

C1.2 Further Exploration of Domain Adaptation Outcomes

The precision and recall values for the B-FOOD and I-FOOD classes exhibit a notable decline when applying the trained models from the *FoodBase* dataset to evaluate their performance on the *ReviewsDB* dataset. Table C-3 provides a detailed breakdown of precision, recall, and F1-score values, highlighting the marked decrease in these metrics.

Table C-3.

Precision, Recall, and F1 Scores Referring to the Classes of the Two Baseline Models After Applying Domain Adaptation

Word	Model	B-FOOD			I-FOOD			O		
Embedding	Architecture	P	R	F1	P	R	F1	P	R	F1
GloVe	BiLSTM-CRF	0.10	0.16	0.12	0.27	0.06	0.09	0.96	0.97	0.96
	Char-BiLSTM-CRF	0.13	0.17	0.15	0.31	0.05	0.10	0.96	0.97	0.96
Word2vec	BiLSTM-CRF	0.11	0.19	0.14	0.42	0.08	0.14	0.96	0.97	0.97
	Char-BiLSTM-CRF	0.09	0.15	0.11	0.31	0.08	0.13	0.96	0.97	0.96
FastText	BiLSTM-CRF	0.10	0.19	0.13	0.16	0.11	0.13	0.96	0.95	0.96
	Char-BiLSTM-CRF	0.16	0.10	0.12	0.71	0.04	0.07	0.96	0.99	0.97
Lexical-300	BiLSTM-CRF	0.22	0.31	0.26	0.22	0.17	0.19	0.95	0.94	0.95
	Char-BiLSTM-CRF	0.23	0.31	0.26	0.38	0.09	0.14	0.95	0.96	0.95

The precision values for the B-FOOD class are particularly concerning, with the highest reported precision being 0.23 when employing the lexical word embeddings in conjunction with the BiLSTM-CRF with CharEmbd model. In contrast, when the same model was evaluated on the *FoodBase* dataset, it achieved a significantly higher precision value of 0.91. A similar trend is observed for the I-FOOD class, where the maximum precision value of

0.71 is attained using the FastText word embedding model with the BiLSTM-CRF with CharEmbd architecture on the *ReviewsDB* dataset. However, when assessing the same model on the *FoodBase* dataset, the precision value notably improves to 0.81.

Shifting focus to recall values, the results continue to be suboptimal. The maximum recall value achieved for the B-FOOD class is 0.31, while for the I-FOOD class, it is only 0.17. In contrast, when evaluating the models on the *FoodBase* dataset, significantly higher maximum recall values of 0.94 for the B-FOOD class and 0.88 for the I-FOOD class are observed.

These findings underscore a substantial degradation in the models' ability to identify B-FOOD and I-FOOD classes when applied to the *ReviewsDB* dataset in comparison to their performance on the *FoodBase* dataset. The observed decline in precision and recall metrics highlights the significant challenges encountered in the domain adaptation of knowledge between the two domains, specifically in the context of detecting food names in recipes versus dish names in customer reviews.

C1.3 Further Exploration on Results of Detecting Dish Names in *MenuNER* Dataset Using Fixed Pre-Trained Word Embeddings

By utilizing the fixed pre-trained word embeddings, Table C-4 demonstrates the performance results for the BiLSTM-CRF model in detecting dish names in *MenuNER* dataset. The following are the analyzed results as per Table C-4:

Table C-4.

Precision, Recall, and F1 Scores for Each With BiLSTM(50N)-CRF Model Using Fixed Pre-Trained Word Embeddings

BiLSTM(50N)-CRF (Rmsprop) – Fixed Word Embeddings									
Embeddings	B-FOOD			I-FOOD			O		
	P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	0.70	0.41	0.50	0.68	0.52	0.57	0.97	0.99	0.98
glove-twitter-100	0.64	0.44	0.51	0.65	0.53	0.58	0.97	0.99	0.98
glove-twitter-200	0.59	0.46	0.52	0.59	0.53	0.56	0.98	0.98	0.98
glove-wiki-gigaword-50	0.68	0.47	0.54	0.67	0.56	0.60	0.98	0.99	0.98
glove-wiki-gigaword-300	0.62	0.48	0.54	0.62	0.56	0.59	0.98	0.99	0.98
word2vec-google-news-300	0.64	0.45	0.52	0.63	0.54	0.58	0.98	0.99	0.98
fasttext-wiki-news-subwords-300	0.64	0.51	0.56	0.65	0.60	0.62	0.98	0.99	0.98

Precision, Recall, and F1-scores: Notably, precision scores range from approximately 0.59 to 0.70 for the B-FOOD class and from 0.44 to 0.51 for the I-FOOD class. These figures indicate the model's ability to correctly identify dish names while minimizing false positives.

Overall Performance: The model demonstrates a good balance between precision and recall, as reflected in the F1-scores, which range from around 0.52 to 0.56 for the B-FOOD class and from 0.54 to 0.62 for the I-FOOD class. This suggests that the model effectively captures dish names in menu descriptions, achieving a harmonious trade-off between precision and recall.

Impact of Word Embeddings: It's worth noting that different pre-trained word embeddings have a varying impact on performance. For example, *glove-twitter-25* and *glove-twitter-100* embeddings yield slightly lower F1-scores compared to other embeddings for both B-FOOD and I-FOOD classes.

In addition to the utilization of different pre-trained word embedding to detect dish names, the character embeddings are also utilized by the BiLSTM-CRF with CharEmbd model as per the demonstrated performance results in Table C-5. The following are the analyzed results as per the table:

Table C-5.

Precision, Recall, and F1 Scores for Each Class With BiLSTM(50N)-CRF With CharEmbd Model Using Fixed Pre-Trained Word Embeddings

BiLSTM(50N)-CRF(rmsprop) With CharEmbd– Fixed Word Embeddings									
Embeddings	B-FOOD			I-FOOD			O		
	P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	0.79	0.72	0.75	0.77	0.75	0.75	0.99	0.99	0.99
glove-twitter-100	0.79	0.71	0.75	0.80	0.74	0.76	0.99	0.99	0.99
glove-twitter-200	0.80	0.71	0.75	0.83	0.71	0.76	0.99	0.99	0.99
glove-wiki-gigaword-50	0.81	0.70	0.74	0.81	0.73	0.76	0.99	0.99	0.99
glove-wiki-gigaword-300	0.77	0.70	0.74	0.81	0.72	0.76	0.99	0.99	0.99
word2vec-google-news-300	0.76	0.72	0.74	0.79	0.75	0.76	0.99	0.99	0.99
fasttext-wiki-news-subwords-300	0.74	0.75	0.74	0.76	0.78	0.76	0.99	0.99	0.99

Precision, Recall, and F1-scores: Notably, precision scores range from approximately 0.74 to 0.81 for the B-FOOD class and from 0.70 to 0.75 for the I-FOOD class. These figures indicate a high degree of accuracy in identifying dish names, with minimal false positives, same as in the BiLSTM-CRF model.

Overall Performance: The model exhibits excellent overall performance, as reflected in the F1-scores, which range from around 0.74 to 0.75 for the B-FOOD class and consistently hover around 0.76 for the I-FOOD class. These high F1-scores underscore the model's effectiveness in capturing dish names within menu descriptions while maintaining a strong balance between precision and recall.

Impact of Word Embeddings: Similar to Table C-4, different pre-trained word embeddings have varying effects on performance. For example, *glove-wiki-gigaword-50* and *glove-wiki-gigaword-300* embeddings exhibit slightly lower F1-scores compared to other embeddings. However, the overall performance remains consistently high for all embeddings.

C1.4 Further Exploration on the Performance of Trained Models After Incorporating the Part-Of-Speech Tags as Additional Input

Table C-6 presents the performance results of the BiLSTM-CRF model enriched with character embeddings after the inclusion of Part-of-Speech (POS) tags as additional input features. These results offer valuable insights into the model's ability to detect named entities, particularly focused on food-related entities, as indicated by the B-FOOD and I-FOOD classes.

Table C-6.

Precision, Recall, and F1 Scores for Each Class With BiLSTM(50N)-CRF(Rmsprop) With CharEmbd Model Using Fixed Pre-Trained Word Embeddings and POS Features.

BiLSTM(50N)-CRF(rmsprop) with CharEmbd– Fixed Word Embeddings									
Embeddings	B-FOOD			I-FOOD			O		
	P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	0.81	0.77	0.79	0.82	0.81	0.81	0.99	0.99	0.99
glove-twitter-100	0.83	0.75	0.78	0.84	0.78	0.81	0.99	0.99	0.99
glove-twitter-200	0.82	0.74	0.77	0.85	0.74	0.80	0.99	0.99	0.99
glove-wiki-gigaword-50	0.83	0.76	0.79	0.81	0.80	0.80	0.99	0.99	0.99
glove-wiki-gigaword-300	0.83	0.75	0.79	0.86	0.78	0.81	0.99	1.00	0.99
word2vec-google-news-300	0.79	0.77	0.77	0.81	0.78	0.79	0.99	0.99	0.99
fasttext-wiki-news-subwords-300	0.76	0.80	0.78	0.77	0.84	0.80	0.99	0.99	0.99

Across various word embedding models, the model consistently exhibits strong performance. It achieves precision values ranging from approximately 0.76 to 0.83 for the B-FOOD class and from 0.77 to 0.86 for the I-FOOD class. These precision values indicate the model's capacity to make accurate positive predictions for food-related named entities. Regarding recall, the model maintains a consistently high standard, with values ranging from approximately 0.74 to 0.80 for the B-FOOD class and from 0.74 to 0.84 for the I-FOOD class. These recall values suggest that the model effectively captures a significant proportion of actual food-related named entities in the dataset. Balancing precision and recall, the F1 scores are also noteworthy. They range from approximately 0.77 to 0.79 for the B-FOOD class and

from 0.79 to 0.81 for the I-FOOD class. Furthermore, the consistently high macro average precision, recall, and F1 scores across different word embedding models underscore the model's robustness.

C1.5 Further Exploration Into the Impact of Utilizing Bert Embeddings

Experiment with BiLSTM-CRF with CharEmbd model using Bert Word Embeddings and POS features as input into the model is illustrated in Table C-7.

Table C-7.

Precision, Recall, and F1 Scores for Each Class With BiLSTM-CRF(Rmsprop) With CharEmbd Model Using Bert Word Embeddings and POS Features

BiLSTM(50N)-CRF(rmsprop) with CharEmbd – Bert Word Embeddings									
Embeddings	B-FOOD			I-FOOD			O		
	P	R	F1	P	R	F1	P	R	F1
bert-cased-768	0.84	0.76	0.80	0.89	0.72	0.80	0.99	1.00	0.99
bert-uncased-768	0.77	0.75	0.76	0.77	0.81	0.78	0.99	0.99	0.99
bert-combined-mean-768	0.82	0.74	0.78	0.81	0.79	0.80	0.99	0.99	0.99
bert-combined-conct-1536	0.79	0.77	0.78	0.76	0.81	0.79	0.99	0.99	0.99

One key observation is the consistently high F1-scores across all Bert Word Embeddings, ranging from 0.76 to 0.80 for the B-FOOD class, and 0.78 to 0.80 for the I-FOOD class. This indicates that when the model is augmented with d with POS tags and Bert Word Embeddings, performs consistently well in recognizing food-related named entities across different embedding types and dimensions.

Among the Bert Word Embeddings, bert-cased-768 achieves the highest F1-scores for both the B-FOOD and I-FOOD classes (0.80 and 0.80, respectively), It also maintains a perfect recall of 1.00 for the O class, indicating its ability to correctly identify non-food entities.

Overall, these results highlight the robustness and effectiveness of the BiLSTM(50N)-CRF(rmsprop) model with CharEmbd when combined with POS tags and Bert Word Embeddings. It showcases the model's capacity to consistently recognize food-related named entities, regardless of the specific Bert Word Embeddings used, making it a valuable tool for the dish entity recognition.

C.2 Further Analysis of Deep Learning Models With Trainable Word Embeddings on *MenuNER* Dataset

This section provides a detailed analysis of the experimental results presented in Section 4.2. It delves into additional findings concerning the efficacy of various evaluation metrics applied to Named Entity Recognition (NER) deep learning models. These models undergo training and assessment via five-fold cross-validation, specifically utilizing the *MenuNER* dataset. Unless explicitly indicated otherwise, all models share a uniform input configuration that comprises trainable word embeddings, character embeddings, and part-of-speech (POS) features.

C2.1 Further Exploration on Results of Models With Single Type of Recurrent Layers

Table C-8

Evaluation Metrics of BiLSTM(50N)-CRF (rmsprop) Model Per Class

BiLSTM(50N)-CRF (rmsprop)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	79.20	0.87	0.88	0.87	0.88	0.86	0.87	1.00	1.00	0.99
glove-twitter-100	61.00	0.85	0.88	0.87	0.85	0.88	0.86	1.00	0.99	0.99
glove-twitter-200	61.60	0.84	0.89	0.86	0.84	0.88	0.86	1.00	0.99	0.99
glove-wiki-gigaword-50	74.20	0.86	0.89	0.88	0.86	0.87	0.86	1.00	0.99	1.00
glove-wiki-gigaword-300	54.20	0.84	0.88	0.86	0.84	0.87	0.86	1.00	0.99	0.99
word2vec-google-news-300	63.20	0.85	0.88	0.86	0.84	0.88	0.86	1.00	0.99	0.99
fasttext-wiki-news- 300	59.40	0.82	0.89	0.85	0.78	0.90	0.83	1.00	0.99	0.99
bert-cased-768	56.40	0.87	0.84	0.85	0.90	0.78	0.83	0.99	1.00	0.99
bert-uncased-768	64.00	0.81	0.88	0.84	0.76	0.90	0.82	1.00	0.99	0.99
bert-combined-mean-768	61.20	0.81	0.88	0.84	0.75	0.90	0.81	1.00	0.99	0.99
bert-combined-conct-1536	60.20	0.81	0.89	0.85	0.77	0.90	0.83	1.00	0.99	0.99
lexical-300	60.40	0.83	0.88	0.85	0.80	0.90	0.84	1.00	0.99	0.99

Table C-9

Comparison Between BiLSTM(50)-CRF (rmsprop) With Fixed Embeddings and BiLSTM(50)-CRF(rmsprop) With Trainable Embeddings

Embedding Models	BiLSTM-CRF (Fixed)			BiLSTM-CRF (Trainable)		
	P	R	F1	P	R	F1
glove-twitter-25	87.40%	85.47%	86.20%	91.40%	91.33%	90.93%
glove-twitter-100	88.67%	84.00%	86.07%	90.00%	91.80%	90.80%
glove-twitter-200	88.73%	82.27%	85.53%	89.33%	91.87%	90.60%
glove-wiki-gigaword-50	87.60%	85.20%	86.00%	90.67%	91.73%	91.13%
glove-wiki-gigaword-300	89.20%	84.33%	86.40%	89.27%	91.67%	90.40%
word2vec-google-news-300	86.40%	84.60%	85.13%	89.53%	91.60%	90.47%
fasttext-wiki-news- 300	84.00%	87.67%	85.73%	86.60%	92.53%	89.20%
bert-cased-768	90.60%	82.73%	86.27%	92.27%	87.33%	89.40%
bert-uncased-768	84.40%	85.07%	84.53%	85.80%	92.20%	88.60%
bert-combined-mean-768	87.40%	84.33%	85.73%	85.13%	92.13%	88.20%
bert-combined-conct-1536	84.80%	85.73%	85.07%	85.93%	92.67%	88.80%
lexical-300	-	-	-	87.33%	92.27%	89.47%

Table C-10*Evaluation Metrics of BiLSTM(25N)-CRF (Adam) Model Per Class*

BiLSTM(25N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	25.80	0.90	0.87	0.89	0.90	0.88	0.89	1.00	1.00	1.00
glove-twitter-100	17.20	0.91	0.87	0.89	0.90	0.86	0.88	0.99	1.00	1.00
glove-twitter-200	13.60	0.91	0.86	0.88	0.92	0.85	0.88	0.99	1.00	1.00
glove-wiki-gigaword-50	19.80	0.91	0.87	0.89	0.91	0.86	0.88	0.99	1.00	1.00
glove-wiki-gigaword-300	15.60	0.91	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
word2vec-google-news-300	14.00	0.91	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	16.80	0.90	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
bert-cased-768	64.80	0.81	0.80	0.81	0.87	0.72	0.79	0.99	1.00	0.99
bert-uncased-768	13.80	0.90	0.86	0.88	0.89	0.85	0.87	0.99	1.00	1.00
bert-combined-mean-768	13.80	0.90	0.86	0.88	0.89	0.85	0.87	0.99	1.00	1.00
bert-combined-conct-1536	14.60	0.90	0.87	0.88	0.90	0.84	0.87	0.99	1.00	1.00
lexical-300	17.40	0.90	0.86	0.87	0.91	0.84	0.87	0.99	1.00	1.00

Table C-11*Evaluation Metrics of BiLSTM(50N)-CRF (Adam) Model Per Class*

BiLSTM(50N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	45.60	0.90	0.88	0.89	0.90	0.87	0.88	0.99	1.00	1.00
glove-twitter-100	33.60	0.90	0.88	0.89	0.91	0.86	0.88	0.99	1.00	1.00
glove-twitter-200	29.40	0.90	0.87	0.89	0.92	0.85	0.88	0.99	1.00	1.00
glove-wiki-gigaword-50	34.40	0.91	0.87	0.89	0.91	0.86	0.88	0.99	1.00	1.00
glove-wiki-gigaword-300	30.40	0.91	0.87	0.89	0.91	0.85	0.88	0.99	1.00	1.00
word2vec-google-news-300	36.40	0.91	0.87	0.89	0.92	0.84	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	50.60	0.90	0.86	0.88	0.90	0.84	0.87	0.99	1.00	1.00
bert-cased-768	59.60	0.87	0.81	0.84	0.92	0.76	0.83	0.99	1.00	1.00
bert-uncased-768	36.20	0.90	0.87	0.89	0.89	0.87	0.88	0.99	1.00	1.00
bert-combined-mean-768	36.20	0.90	0.87	0.89	0.89	0.87	0.88	0.99	1.00	1.00
bert-combined-conct-1536	31.80	0.90	0.88	0.89	0.88	0.87	0.87	0.99	1.00	1.00
lexical-300	37.60	0.90	0.87	0.88	0.92	0.84	0.88	0.99	1.00	1.00

Table C-12*Evaluation Metrics of BiLSTM(75N)-CRF (Adam) Model Per Class*

BiLSTM(75N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	32.00	0.90	0.88	0.89	0.90	0.88	0.89	1.00	1.00	1.00
glove-twitter-100	24.20	0.91	0.88	0.90	0.91	0.87	0.89	0.99	1.00	1.00
glove-twitter-200	23.80	0.91	0.87	0.89	0.92	0.86	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	28.20	0.91	0.88	0.89	0.91	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	19.40	0.91	0.87	0.89	0.92	0.86	0.89	0.99	1.00	1.00
word2vec-google-news-300	22.80	0.91	0.87	0.89	0.93	0.85	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	23.00	0.92	0.87	0.89	0.93	0.83	0.88	0.99	1.00	1.00
bert-cased-768	42.60	0.87	0.83	0.85	0.91	0.76	0.83	0.99	1.00	0.99
bert-uncased-768	22.80	0.91	0.87	0.89	0.90	0.85	0.87	0.99	1.00	1.00
bert-combined-mean-768	22.80	0.91	0.87	0.89	0.90	0.85	0.87	0.99	1.00	1.00
bert-combined-conct-1536	21.40	0.91	0.87	0.89	0.90	0.86	0.87	0.99	1.00	1.00
lexical-300	24.00	0.91	0.87	0.89	0.92	0.84	0.88	0.99	1.00	1.00

Table C-13*Evaluation Metrics of BiLSTM(100N)-CRF (Adam) Model Per Class*

BiLSTM(100N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	37.00	0.90	0.88	0.89	0.88	0.88	0.88	1.00	1.00	1.00
glove-twitter-100	21.20	0.91	0.88	0.89	0.90	0.88	0.89	1.00	1.00	1.00
glove-twitter-200	18.40	0.91	0.88	0.89	0.92	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	23.80	0.91	0.88	0.89	0.90	0.88	0.89	1.00	1.00	1.00
glove-wiki-gigaword-300	18.80	0.91	0.88	0.89	0.91	0.87	0.89	0.99	1.00	1.00
word2vec-google-news-300	18.40	0.91	0.88	0.89	0.91	0.87	0.89	1.00	1.00	1.00
fasttext-wiki-news- 300	37.60	0.90	0.87	0.89	0.92	0.85	0.88	0.99	1.00	1.00
bert-cased-768	65.80	0.89	0.82	0.85	0.91	0.77	0.83	0.99	1.00	1.00
bert-uncased-768	21.20	0.91	0.87	0.89	0.90	0.86	0.88	0.99	1.00	1.00
bert-combined-mean-768	21.20	0.91	0.87	0.89	0.90	0.86	0.88	0.99	1.00	1.00
bert-combined-conct-1536	18.60	0.90	0.87	0.88	0.87	0.89	0.88	1.00	1.00	1.00
lexical-300	18.00	0.91	0.87	0.89	0.92	0.85	0.88	0.99	1.00	1.00

Table C-14*Evaluation Metrics of BiLSTM(125N)-CRF (Adam) Model Per Class*

BiLSTM(125N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	84.00	0.88	0.87	0.88	0.87	0.87	0.87	1.00	1.00	1.00
glove-twitter-100	18.40	0.90	0.88	0.89	0.88	0.89	0.89	1.00	1.00	1.00
glove-twitter-200	18.00	0.91	0.88	0.89	0.90	0.88	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	61.00	0.89	0.87	0.88	0.89	0.87	0.88	0.99	1.00	1.00
glove-wiki-gigaword-300	17.20	0.90	0.88	0.89	0.90	0.87	0.89	0.99	1.00	1.00
word2vec-google-news-300	34.20	0.90	0.87	0.88	0.90	0.86	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	18.20	0.92	0.87	0.89	0.91	0.86	0.88	0.99	1.00	1.00
bert-cased-768	27.20	0.88	0.82	0.85	0.91	0.78	0.84	0.99	1.00	0.99
bert-uncased-768	16.80	0.90	0.87	0.89	0.90	0.87	0.88	0.99	1.00	1.00
bert-combined-mean-768	16.80	0.90	0.87	0.89	0.90	0.87	0.88	0.99	1.00	1.00
bert-combined-conct-1536	17.20	0.91	0.87	0.89	0.88	0.87	0.88	0.99	1.00	1.00
lexical-300	15.40	0.91	0.87	0.89	0.91	0.86	0.88	0.99	1.00	1.00

Table C-15*Evaluation Metrics of BiLSTM(150N)-CRF (Adam) Model Per Class*

BiLSTM(150N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	33.20	0.91	0.88	0.89	0.88	0.89	0.88	1.00	1.00	1.00
glove-twitter-100	25.20	0.91	0.87	0.89	0.90	0.88	0.89	1.00	1.00	1.00
glove-twitter-200	17.20	0.91	0.88	0.89	0.89	0.88	0.88	1.00	1.00	1.00
glove-wiki-gigaword-50	27.60	0.90	0.87	0.89	0.89	0.88	0.88	0.99	1.00	1.00
glove-wiki-gigaword-300	15.60	0.92	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
word2vec-google-news-300	16.40	0.91	0.88	0.89	0.90	0.87	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	17.00	0.91	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
bert-cased-768	30.60	0.88	0.83	0.85	0.91	0.77	0.83	0.99	1.00	0.99
bert-uncased-768	15.80	0.90	0.87	0.89	0.90	0.87	0.88	0.99	1.00	1.00
bert-combined-mean-768	15.80	0.90	0.87	0.89	0.90	0.87	0.88	0.99	1.00	1.00
bert-combined-conct-1536	16.40	0.90	0.86	0.89	0.87	0.88	0.88	1.00	1.00	1.00
lexical-300	15.40	0.92	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00

Table C-16*Evaluation Metrics of 2BiLSTM(50N)-CRF (rmsprop) Model Per Class*

2BiLSTM(50N)-CRF (rmsprop)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	71.40	0.87	0.88	0.88	0.87	0.88	0.87	1.00	0.99	1.00
glove-twitter-100	55.40	0.89	0.86	0.88	0.87	0.87	0.87	0.99	1.00	1.00
glove-twitter-200	43.20	0.87	0.87	0.87	0.86	0.86	0.86	1.00	1.00	1.00
glove-wiki-gigaword-50	60.40	0.87	0.88	0.88	0.86	0.88	0.87	1.00	1.00	1.00
glove-wiki-gigaword-300	45.20	0.88	0.88	0.88	0.88	0.87	0.87	1.00	1.00	1.00
word2vec-google-news-300	52.40	0.88	0.87	0.87	0.87	0.88	0.87	1.00	1.00	1.00
fasttext-wiki-news- 300	36.60	0.86	0.88	0.87	0.84	0.88	0.86	1.00	0.99	0.99
bert-cased-768	65.40	0.84	0.82	0.83	0.85	0.82	0.83	0.99	1.00	0.99
bert-uncased-768	36.40	0.88	0.87	0.87	0.89	0.84	0.87	0.99	1.00	1.00
bert-combined-mean-768	43.80	0.87	0.87	0.87	0.87	0.85	0.86	1.00	1.00	1.00
bert-combined-conct-1536	36.40	0.85	0.88	0.87	0.83	0.89	0.86	1.00	0.99	0.99
lexical-300	40.20	0.87	0.87	0.87	0.86	0.87	0.86	1.00	0.99	0.99

Table C-17*Evaluation Metrics of 2BiLSTM(25N)-CRF (Adam) Model Per Class*

2BiLSTM(25N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	106.80	0.88	0.86	0.87	0.91	0.84	0.87	0.99	1.00	1.00
glove-twitter-100	62.00	0.89	0.86	0.87	0.92	0.82	0.87	0.99	1.00	1.00
glove-twitter-200	51.20	0.88	0.85	0.86	0.91	0.81	0.86	0.99	1.00	1.00
glove-wiki-gigaword-50	61.80	0.89	0.86	0.87	0.90	0.84	0.87	0.99	1.00	1.00
glove-wiki-gigaword-300	60.00	0.89	0.85	0.87	0.92	0.82	0.87	0.99	1.00	1.00
word2vec-google-news-300	59.60	0.88	0.84	0.86	0.91	0.79	0.85	0.99	1.00	1.00
fasttext-wiki-news- 300	50.00	0.88	0.84	0.86	0.91	0.80	0.85	0.99	1.00	1.00
bert-cased-768	57.20	0.85	0.81	0.83	0.90	0.76	0.82	0.99	1.00	0.99
bert-uncased-768	48.40	0.89	0.84	0.87	0.93	0.82	0.87	0.99	1.00	1.00
bert-combined-mean-768	48.40	0.89	0.84	0.87	0.93	0.82	0.87	0.99	1.00	1.00
bert-combined-conct-1536	45.40	0.89	0.86	0.87	0.90	0.85	0.87	0.99	1.00	1.00
lexical-300	56.00	0.88	0.84	0.86	0.90	0.83	0.86	0.99	1.00	1.00

Table C-18*Evaluation Metrics of 2BiLSTM(50N)-CRF (Adam) Model Per Class*

2BiLSTM(50N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	46.40	0.89	0.86	0.87	0.89	0.87	0.88	0.99	1.00	1.00
glove-twitter-100	43.60	0.90	0.86	0.88	0.89	0.86	0.88	0.99	1.00	1.00
glove-twitter-200	50.80	0.90	0.87	0.88	0.92	0.85	0.88	0.99	1.00	1.00
glove-wiki-gigaword-50	27.60	0.89	0.87	0.88	0.89	0.86	0.88	1.00	1.00	1.00
glove-wiki-gigaword-300	36.80	0.90	0.86	0.88	0.91	0.84	0.87	0.99	1.00	1.00
word2vec-google-news-300	34.20	0.90	0.86	0.88	0.90	0.85	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	41.00	0.89	0.86	0.87	0.90	0.85	0.87	0.99	1.00	1.00
bert-cased-768	44.80	0.86	0.83	0.85	0.90	0.79	0.84	0.99	1.00	1.00
bert-uncased-768	26.60	0.90	0.85	0.88	0.90	0.85	0.88	0.99	1.00	1.00
bert-combined-mean-768	26.60	0.90	0.85	0.88	0.90	0.85	0.88	0.99	1.00	1.00
bert-combined-conct-1536	26.80	0.90	0.86	0.88	0.90	0.85	0.87	0.99	1.00	1.00
lexical-300	40.60	0.90	0.85	0.87	0.89	0.85	0.87	0.99	1.00	1.00

Table C-19*Evaluation Metrics of 2BiLSTM(75N)-CRF (Adam) Model Per Class*

2BiLSTM(75N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	24.20	0.89	0.86	0.88	0.91	0.86	0.88	0.99	1.00	1.00
glove-twitter-100	18.00	0.91	0.87	0.89	0.91	0.86	0.89	1.00	1.00	1.00
glove-twitter-200	31.60	0.89	0.87	0.88	0.90	0.87	0.88	0.99	1.00	1.00
glove-wiki-gigaword-50	20.20	0.89	0.88	0.89	0.89	0.89	0.89	1.00	1.00	1.00
glove-wiki-gigaword-300	16.20	0.90	0.87	0.89	0.89	0.88	0.89	0.99	1.00	1.00
word2vec-google-news-300	16.40	0.91	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	19.00	0.89	0.86	0.87	0.88	0.86	0.87	0.99	1.00	1.00
bert-cased-768	18.20	0.87	0.82	0.84	0.91	0.80	0.85	0.99	1.00	0.99
bert-uncased-768	16.60	0.90	0.86	0.88	0.88	0.87	0.87	0.99	1.00	1.00
bert-combined-mean-768	16.60	0.90	0.86	0.88	0.88	0.87	0.87	0.99	1.00	1.00
bert-combined-conct-1536	17.60	0.90	0.86	0.88	0.88	0.87	0.87	0.99	1.00	1.00
lexical-300	19.80	0.89	0.86	0.87	0.89	0.85	0.87	0.99	1.00	1.00

Table C-20*Evaluation Metrics of 2BiLSTM(100N)-CRF (Adam) Model Per Class*

2BiLSTM(100N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	21.00	0.90	0.87	0.88	0.88	0.89	0.89	1.00	1.00	1.00
glove-twitter-100	32.60	0.88	0.88	0.87	0.88	0.88	0.88	1.00	1.00	1.00
glove-twitter-200	14.20	0.90	0.87	0.89	0.89	0.87	0.88	0.99	1.00	1.00
glove-wiki-gigaword-50	18.40	0.90	0.86	0.88	0.91	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	12.80	0.90	0.87	0.88	0.90	0.87	0.89	0.99	1.00	1.00
word2vec-google-news-300	13.80	0.90	0.87	0.88	0.88	0.88	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	14.20	0.90	0.86	0.88	0.88	0.88	0.88	0.99	1.00	1.00
bert-cased-768	14.60	0.88	0.82	0.85	0.91	0.80	0.85	0.99	1.00	0.99
bert-uncased-768	14.60	0.91	0.86	0.88	0.88	0.88	0.88	0.99	1.00	1.00
bert-combined-mean-768	14.60	0.91	0.86	0.88	0.88	0.88	0.88	0.99	1.00	1.00
bert-combined-conct-1536	24.40	0.90	0.86	0.88	0.88	0.88	0.87	0.99	1.00	1.00
lexical-300	29.20	0.88	0.84	0.86	0.88	0.85	0.87	0.99	1.00	1.00

Table C-21*Evaluation metrics of BiGRU(50N)-CRF (rmsprop) with fixed embeddings per class*

BiGRU(50N)-CRF(rmsprop) – Fixed Word Embeddings										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	158.80	0.84	0.79	0.81	0.85	0.80	0.82	0.96	0.95	0.96
glove-twitter-100	155.60	0.82	0.78	0.80	0.83	0.81	0.82	0.99	0.99	0.99
glove-twitter-200	146.20	0.81	0.77	0.79	0.84	0.79	0.81	0.99	0.99	0.99
glove-wiki-gigaword-50	182.20	0.84	0.81	0.83	0.85	0.82	0.84	0.99	0.99	0.99
glove-wiki-gigaword-300	134.60	0.82	0.77	0.79	0.85	0.79	0.82	0.99	0.99	0.99
word2vec-google-news-300	127.40	0.79	0.81	0.80	0.81	0.83	0.82	0.99	0.99	0.99
fasttext-wiki-news- 300	177.20	0.83	0.81	0.82	0.84	0.83	0.84	0.99	0.99	0.99
bert-cased-768	76.00	0.82	0.74	0.78	0.88	0.71	0.78	0.99	1.00	0.99
bert-uncased-768	159.40	0.81	0.77	0.79	0.82	0.79	0.80	0.99	0.99	0.99
bert-combined-mean-768	159.40	0.81	0.77	0.79	0.82	0.79	0.80	0.99	0.99	0.99
bert-combined-conct-1536	113.00	0.82	0.79	0.80	0.80	0.82	0.80	0.99	0.99	0.99

Table C-22*Evaluation Metrics of BiGRU(50N)-CRF (rmsprop) with Trainable Embeddings Per Class*

BiGRU(50N)-CRF(rmsprop) – Trainable Word Embeddings										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	92.20	0.86	0.90	0.88	0.86	0.89	0.87	1.00	0.99	0.99
glove-twitter-100	70.40	0.86	0.89	0.88	0.85	0.89	0.87	1.00	0.99	1.00
glove-twitter-200	62.00	0.87	0.87	0.87	0.86	0.87	0.86	0.99	1.00	1.00
glove-wiki-gigaword-50	80.20	0.86	0.88	0.87	0.84	0.89	0.86	0.99	0.99	0.99
glove-wiki-gigaword-300	61.60	0.88	0.88	0.88	0.85	0.88	0.86	1.00	0.99	0.99
word2vec-google-news-300	61.60	0.86	0.88	0.87	0.84	0.89	0.86	1.00	0.99	0.99
fasttext-wiki-news- 300	99.20	0.85	0.87	0.86	0.82	0.88	0.84	1.00	0.99	0.99
bert-cased-768	57.00	0.88	0.81	0.84	0.87	0.78	0.82	0.99	1.00	0.99
bert-uncased-768	60.80	0.81	0.89	0.85	0.76	0.90	0.82	1.00	0.99	0.99
bert-combined-mean-768	60.60	0.83	0.88	0.85	0.77	0.89	0.82	1.00	0.99	0.99
bert-combined-conct-1536	61.60	0.83	0.89	0.86	0.79	0.89	0.84	1.00	0.99	0.99
Lexical-300	74.20	0.83	0.88	0.86	0.75	0.90	0.81	1.00	0.99	0.99

Table C-23

Comparison Between BiGRU(50)-CRF (rmsprop) With Fixed Embeddings and BiGRU(50)-CRF(rmsprop) With Trainable Embeddings

Embedding Models	BiGRU-CRF (Fixed)			BiGRU-CRF (Trainable)		
	P	R	F1	P	R	F1
glove-twitter-25	88.27%	84.67%	86.27%	90.47%	92.67%	91.40%
glove-twitter-100	87.80%	86.13%	86.80%	90.27%	92.47%	91.33%
glove-twitter-200	87.93%	85.07%	86.47%	91.00%	91.40%	91.13%
glove-wiki-gigaword-50	89.60%	87.47%	88.53%	90.00%	92.07%	90.87%
glove-wiki-gigaword-300	88.67%	85.20%	86.80%	90.60%	91.93%	91.13%
word2vec-google-news-300	86.47%	87.47%	87.00%	89.80%	92.13%	90.87%
fasttext-wiki-news- 300	88.73%	87.80%	88.27%	88.67%	91.53%	90.00%
bert-cased-768	89.40%	81.80%	85.07%	91.13%	86.13%	88.33%
bert-uncased-768	87.20%	85.33%	86.13%	85.73%	92.53%	88.67%
bert-combined-mean-768	87.20%	85.33%	86.13%	86.33%	92.07%	88.87%
bert-combined-conct-1536	86.87%	86.60%	86.27%	87.20%	92.40%	89.60%
lexical-300	-	-	-	85.73%	92.47%	88.67%

Table C-24*Evaluation Metrics of BiGRU(25N)-CRF (Adam) Per Class*

BiGRU(25N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	201.20	0.91	0.88	0.89	0.91	0.86	0.88	0.99	1.00	1.00
glove-twitter-100	128.40	0.89	0.85	0.87	0.91	0.82	0.86	0.99	1.00	1.00
glove-twitter-200	128.20	0.90	0.85	0.88	0.91	0.83	0.87	0.99	1.00	1.00
glove-wiki-gigaword-50	187.60	0.90	0.86	0.88	0.90	0.85	0.87	0.99	1.00	1.00
glove-wiki-gigaword-300	134.20	0.89	0.84	0.87	0.91	0.81	0.86	0.99	1.00	1.00
word2vec-google-news-300	108.00	0.90	0.85	0.88	0.91	0.82	0.87	0.99	1.00	1.00
fasttext-wiki-news- 300	121.00	0.90	0.84	0.87	0.92	0.81	0.86	0.99	1.00	1.00
bert-cased-768	200.20	0.86	0.77	0.81	0.89	0.74	0.81	0.99	1.00	0.99
bert-uncased-768	107.20	0.90	0.86	0.88	0.91	0.84	0.87	0.99	1.00	1.00
bert-combined-mean-768	107.20	0.90	0.86	0.88	0.91	0.84	0.87	0.99	1.00	1.00
bert-combined-conct-1536	93.80	0.91	0.86	0.88	0.91	0.84	0.88	0.99	1.00	1.00
lexical-300	100.20	0.90	0.85	0.87	0.91	0.82	0.86	0.99	1.00	1.00

Table C-25*Evaluation Metrics of BiGRU(50N)-CRF (Adam) Per Class*

BiGRU(50N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	31.80	0.91	0.87	0.89	0.90	0.87	0.88	1.00	1.00	1.00
glove-twitter-100	31.60	0.91	0.87	0.89	0.91	0.86	0.89	0.99	1.00	1.00
glove-twitter-200	32.20	0.91	0.87	0.89	0.91	0.86	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	34.40	0.91	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	27.40	0.91	0.86	0.89	0.92	0.85	0.88	0.99	1.00	1.00
word2vec-google-news-300	36.00	0.91	0.86	0.89	0.91	0.86	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	33.00	0.90	0.86	0.88	0.90	0.84	0.87	0.99	1.00	1.00
bert-cased-768	75.20	0.87	0.80	0.83	0.90	0.75	0.82	0.99	1.00	0.99
bert-uncased-768	37.20	0.91	0.87	0.89	0.90	0.88	0.89	1.00	1.00	1.00
bert-combined-mean-768	37.20	0.91	0.87	0.89	0.90	0.88	0.89	1.00	1.00	1.00
bert-combined-conct-1536	38.00	0.91	0.87	0.89	0.89	0.88	0.88	0.99	1.00	1.00
lexical-300	32.00	0.91	0.86	0.88	0.90	0.85	0.87	0.99	1.00	1.00

Table C-26*Evaluation metrics Of BiGRU(75N)-CRF (Adam) Per Class*

BiGRU(75N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	27.00	0.91	0.87	0.89	0.90	0.88	0.88	0.99	1.00	1.00
glove-twitter-100	20.40	0.92	0.87	0.89	0.92	0.87	0.89	0.99	1.00	1.00
glove-twitter-200	21.80	0.92	0.87	0.89	0.92	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	23.60	0.91	0.87	0.90	0.91	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	17.80	0.92	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
word2vec-google-news-300	18.60	0.92	0.86	0.89	0.92	0.87	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	20.40	0.92	0.86	0.89	0.92	0.86	0.88	0.99	1.00	1.00
bert-cased-768	46.60	0.87	0.80	0.83	0.91	0.74	0.81	0.99	1.00	0.99
bert-uncased-768	22.80	0.91	0.87	0.89	0.89	0.88	0.88	0.99	1.00	1.00
bert-combined-mean-768	22.80	0.91	0.87	0.89	0.89	0.88	0.88	0.99	1.00	1.00
bert-combined-conct-1536	22.20	0.91	0.87	0.89	0.89	0.87	0.88	0.99	1.00	1.00
lexical-300	21.80	0.91	0.86	0.89	0.91	0.85	0.88	0.99	1.00	1.00

Table C-27*Evaluation Metrics of BiGRU(100N)-CRF (Adam) Per Class*

BiGRU(100N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	25.00	0.91	0.88	0.89	0.90	0.89	0.89	1.00	1.00	1.00
glove-twitter-100	19.20	0.92	0.88	0.90	0.91	0.88	0.89	1.00	1.00	1.00
glove-twitter-200	16.80	0.92	0.88	0.90	0.90	0.88	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	31.40	0.91	0.87	0.89	0.90	0.88	0.89	1.00	1.00	1.00
glove-wiki-gigaword-300	17.60	0.91	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
word2vec-google-news-300	15.60	0.93	0.87	0.89	0.92	0.86	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	18.00	0.92	0.86	0.89	0.91	0.86	0.88	0.99	1.00	1.00
bert-cased-768	29.80	0.88	0.81	0.84	0.90	0.76	0.83	0.99	1.00	0.99
bert-uncased-768	20.40	0.91	0.87	0.89	0.90	0.87	0.88	0.99	1.00	1.00
bert-combined-mean-768	20.40	0.91	0.87	0.89	0.90	0.87	0.88	0.99	1.00	1.00
bert-combined-conct-1536	16.00	0.91	0.87	0.89	0.89	0.88	0.88	0.99	1.00	1.00
lexical-300	18.20	0.92	0.86	0.89	0.91	0.85	0.88	0.99	1.00	1.00

Table C-28*Evaluation Metrics of BiGRU(125N)-CRF (Adam) Per Class*

BiGRU(125N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	22.20	0.91	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
glove-twitter-100	16.20	0.92	0.87	0.90	0.91	0.87	0.89	0.99	1.00	1.00
glove-twitter-200	16.20	0.91	0.87	0.90	0.91	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	21.40	0.92	0.87	0.90	0.90	0.88	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	16.40	0.92	0.87	0.89	0.91	0.88	0.89	0.99	1.00	1.00
word2vec-google-news-300	15.60	0.92	0.86	0.89	0.91	0.87	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	18.80	0.91	0.87	0.89	0.91	0.85	0.88	0.99	1.00	1.00
bert-cased-768	27.20	0.87	0.82	0.84	0.91	0.76	0.83	0.99	1.00	0.99
bert-uncased-768	17.00	0.92	0.87	0.89	0.92	0.88	0.90	0.99	1.00	1.00
bert-combined-mean-768	17.00	0.92	0.87	0.89	0.92	0.88	0.90	0.99	1.00	1.00
bert-combined-conct-1536	16.40	0.91	0.87	0.89	0.89	0.87	0.88	0.99	1.00	1.00
lexical-300	16.20	0.92	0.87	0.89	0.92	0.86	0.89	0.99	1.00	1.00

Table C-29*Evaluation Metrics of BiGRU(150N)-CRF (Adam) Per Class*

BiGRU(150N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	68.20	0.89	0.87	0.88	0.87	0.87	0.87	1.00	1.00	1.00
glove-twitter-100	18.00	0.91	0.88	0.89	0.90	0.88	0.89	0.99	1.00	1.00
glove-twitter-200	17.40	0.90	0.88	0.89	0.89	0.88	0.89	1.00	1.00	1.00
glove-wiki-gigaword-50	24.80	0.91	0.88	0.89	0.89	0.88	0.88	1.00	1.00	1.00
glove-wiki-gigaword-300	15.80	0.91	0.88	0.89	0.90	0.88	0.89	0.99	1.00	1.00
word2vec-google-news-300	36.00	0.90	0.87	0.89	0.91	0.86	0.88	1.00	1.00	1.00
fasttext-wiki-news- 300	19.80	0.91	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
bert-cased-768	29.00	0.88	0.83	0.85	0.91	0.77	0.83	0.99	1.00	0.99
bert-uncased-768	18.60	0.91	0.86	0.89	0.90	0.87	0.88	0.99	1.00	1.00
bert-combined-mean-768	18.60	0.91	0.86	0.89	0.90	0.87	0.88	0.99	1.00	1.00
bert-combined-conct-1536	20.60	0.91	0.87	0.89	0.88	0.88	0.88	0.99	1.00	1.00
lexical-300	18.00	0.92	0.86	0.89	0.91	0.86	0.88	0.99	1.00	1.00

Table C-30*Evaluation Metrics of 2BiGRU(50N)-CRF (rmsprop) Per Class*

2BiGRU(50N)-CRF (rmsprop)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	54.00	0.88	0.87	0.88	0.88	0.87	0.87	1.00	1.00	1.00
glove-twitter-100	55.40	0.86	0.87	0.87	0.86	0.88	0.87	1.00	1.00	1.00
glove-twitter-200	47.00	0.87	0.88	0.88	0.87	0.87	0.87	1.00	1.00	1.00
glove-wiki-gigaword-50	54.60	0.88	0.87	0.87	0.87	0.86	0.86	0.99	1.00	1.00
glove-wiki-gigaword-300	41.00	0.89	0.88	0.88	0.87	0.88	0.87	1.00	1.00	1.00
word2vec-google-news-300	46.40	0.88	0.87	0.87	0.87	0.87	0.87	0.99	1.00	1.00
fasttext-wiki-news- 300	52.20	0.87	0.87	0.87	0.83	0.87	0.85	1.00	0.99	1.00
bert-cased-768	55.40	0.88	0.83	0.86	0.91	0.80	0.85	0.99	1.00	0.99
bert-uncased-768	50.40	0.86	0.87	0.86	0.79	0.90	0.84	1.00	0.99	0.99
bert-combined-mean-768	50.40	0.86	0.87	0.86	0.79	0.90	0.84	1.00	0.99	0.99
bert-combined-conct-1536	82.00	0.85	0.89	0.87	0.81	0.89	0.85	1.00	0.99	0.99
lexical-300	52.40	0.85	0.87	0.86	0.80	0.90	0.84	1.00	0.99	0.99

Table C-31*Evaluation Metrics of 2BiGRU(25N)-CRF (Adam) Per Class*

2BiGRU(25N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	70.8	0.89	0.86	0.88	0.90	0.86	0.88	0.99	1.00	1.00
glove-twitter-100	50.2	0.90	0.86	0.87	0.90	0.84	0.87	0.99	1.00	1.00
glove-twitter-200	55.8	0.90	0.85	0.87	0.90	0.85	0.88	0.99	1.00	1.00
glove-wiki-gigaword-50	52.6	0.89	0.86	0.88	0.89	0.86	0.88	0.99	1.00	1.00
glove-wiki-gigaword-300	49.8	0.90	0.86	0.88	0.91	0.84	0.87	0.99	1.00	1.00
word2vec-google-news-300	55.6	0.90	0.85	0.87	0.90	0.84	0.87	0.99	1.00	1.00
fasttext-wiki-news- 300	57.4	0.90	0.86	0.87	0.90	0.84	0.87	0.99	1.00	1.00
bert-cased-768	52.0	0.90	0.85	0.87	0.91	0.84	0.87	0.99	1.00	1.00
bert-uncased-768	74.6	0.81	0.78	0.79	0.87	0.75	0.81	0.99	1.00	0.99
bert-combined-mean-768	60.2	0.91	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
bert-combined-conct-1536	60.2	0.91	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
lexical-300	60.0	0.90	0.86	0.89	0.91	0.85	0.88	0.99	1.00	1.00

Table C-32*Evaluation Metrics of 2BiGRU(50N)-CRF (Adam) Per Class*

2BiGRU(50N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	72.60	0.88	0.86	0.87	0.88	0.86	0.87	0.99	1.00	1.00
glove-twitter-100	53.40	0.89	0.86	0.87	0.90	0.85	0.88	0.99	1.00	1.00
glove-twitter-200	37.20	0.90	0.86	0.87	0.91	0.85	0.88	0.99	1.00	1.00
glove-wiki-gigaword-50	71.40	0.88	0.87	0.87	0.87	0.88	0.87	1.00	1.00	1.00
glove-wiki-gigaword-300	68.20	0.89	0.86	0.88	0.92	0.84	0.87	0.99	1.00	1.00
word2vec-google-news-300	52.00	0.88	0.86	0.87	0.90	0.84	0.87	0.99	1.00	1.00
fasttext-wiki-news- 300	67.60	0.89	0.86	0.88	0.92	0.84	0.88	0.99	1.00	1.00
bert-cased-768	71.20	0.86	0.82	0.85	0.92	0.77	0.84	0.99	1.00	1.00
bert-uncased-768	48.20	0.89	0.86	0.87	0.90	0.85	0.87	0.99	1.00	1.00
bert-combined-mean-768	48.20	0.89	0.86	0.87	0.90	0.85	0.87	0.99	1.00	1.00
bert-combined-conct-1536	27.00	0.90	0.86	0.88	0.89	0.87	0.87	0.99	1.00	1.00
lexical-300	66.60	0.89	0.86	0.87	0.89	0.85	0.87	0.99	1.00	1.00

Table C-33*Evaluation Metrics of 2BiGRU(75N)-CRF (Adam) Per Class*

2BiGRU(75N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	22.60	0.89	0.88	0.89	0.88	0.89	0.89	1.00	1.00	1.00
glove-twitter-100	17.00	0.90	0.88	0.89	0.89	0.88	0.88	1.00	1.00	1.00
glove-twitter-200	16.20	0.90	0.87	0.88	0.89	0.87	0.88	0.99	1.00	1.00
glove-wiki-gigaword-50	20.80	0.90	0.87	0.88	0.90	0.88	0.89	1.00	1.00	1.00
glove-wiki-gigaword-300	16.20	0.90	0.86	0.88	0.91	0.86	0.89	0.99	1.00	1.00
word2vec-google-news-300	14.60	0.91	0.86	0.89	0.90	0.87	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	16.80	0.90	0.85	0.88	0.89	0.87	0.88	0.99	1.00	1.00
bert-cased-768	16.60	0.86	0.80	0.83	0.91	0.78	0.84	0.99	1.00	1.00
bert-uncased-768	15.00	0.90	0.86	0.88	0.90	0.87	0.88	0.99	1.00	1.00
bert-combined-mean-768	15.00	0.90	0.86	0.88	0.90	0.87	0.88	0.99	1.00	1.00
bert-combined-conct-1536	15.40	0.90	0.87	0.88	0.89	0.87	0.88	0.99	1.00	1.00
lexical-300	13.80	0.78	0.87	0.88	0.91	0.87	0.89	0.99	1.00	1.00

Table C-34*Evaluation Metrics of 2BiGRU(100N)-CRF (Adam) Per Class*

2BiGRU(100N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	22.00	0.90	0.86	0.88	0.89	0.88	0.88	1.00	1.00	1.00
glove-twitter-100	15.00	0.90	0.88	0.89	0.89	0.88	0.88	1.00	1.00	1.00
glove-twitter-200	13.80	0.91	0.87	0.89	0.89	0.87	0.88	1.00	1.00	1.00
glove-wiki-gigaword-50	15.80	0.90	0.88	0.89	0.89	0.89	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	16.80	0.91	0.87	0.89	0.90	0.87	0.88	0.99	1.00	1.00
word2vec-google-news-300	14.80	0.90	0.87	0.88	0.89	0.87	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	13.00	0.90	0.86	0.88	0.89	0.88	0.88	0.99	1.00	1.00
bert-cased-768	17.00	0.87	0.83	0.85	0.89	0.83	0.86	0.99	1.00	0.99
bert-uncased-768	14.00	0.90	0.86	0.88	0.89	0.87	0.88	0.99	1.00	1.00
bert-combined-mean-768	14.00	0.90	0.86	0.88	0.89	0.87	0.88	0.99	1.00	1.00
bert-combined-conct-1536	15.40	0.91	0.86	0.88	0.89	0.88	0.88	0.99	1.00	1.00
lexical-300	13.40	0.91	0.86	0.88	0.91	0.86	0.89	0.99	1.00	1.00

C2.2 Further Exploration of Results of Deep Learning Models Using Recurrent Layers

Table C-35

Evaluation Metrics of 1BiLSTM(50)-1BiGRU(50N)-CRF (rmsprop) Per Class

1BiLSTM(50)-1BiGRU(50N)-CRF (rmsprop)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	81.80	0.87	0.89	0.88	0.86	0.88	0.87	1.00	1.00	1.00
glove-twitter-100	60.20	0.87	0.88	0.87	0.87	0.88	0.86	1.00	1.00	1.00
glove-twitter-200	51.60	0.89	0.88	0.88	0.88	0.87	0.88	1.00	1.00	1.00
glove-wiki-gigaword-50	61.00	0.89	0.87	0.88	0.88	0.87	0.87	1.00	1.00	1.00
glove-wiki-gigaword-300	46.80	0.88	0.89	0.88	0.87	0.87	0.87	0.99	1.00	1.00
word2vec-google-news-300	47.60	0.88	0.89	0.88	0.87	0.88	0.86	1.00	1.00	1.00
fasttext-wiki-news- 300	60.20	0.85	0.88	0.87	0.83	0.88	0.85	1.00	0.99	0.99
bert-cased-768	62.80	0.88	0.83	0.86	0.91	0.78	0.84	0.99	1.00	1.00
bert-uncased-768	59.40	0.84	0.89	0.87	0.80	0.89	0.84	1.00	0.99	0.99
bert-combined-mean-768	59.40	0.84	0.89	0.87	0.80	0.89	0.84	1.00	0.99	0.99
bert-combined-conct-1536	41.80	0.83	0.89	0.86	0.82	0.88	0.85	1.00	0.99	0.99
lexical-300	55.40	0.86	0.88	0.87	0.81	0.89	0.85	1.00	0.99	0.99

Table C-36*Evaluation Metrics of 1BiLSTM(25)-1BiGRU(25N)-CRF (Adam) Per Class*

1BiLSTM(25)-1BiGRU(25N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	32.00	0.92	0.88	0.90	0.91	0.87	0.89	1.00	1.00	1.00
glove-twitter-100	27.40	0.92	0.87	0.90	0.92	0.87	0.89	1.00	1.00	1.00
glove-twitter-200	28.00	0.91	0.87	0.90	0.92	0.86	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	30.60	0.92	0.88	0.90	0.91	0.87	0.89	1.00	1.00	1.00
glove-wiki-gigaword-300	29.20	0.92	0.87	0.89	0.92	0.86	0.89	0.99	1.00	1.00
word2vec-google-news-300	28.40	0.91	0.87	0.89	0.91	0.86	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	32.60	0.91	0.87	0.89	0.91	0.86	0.89	0.99	1.00	1.00
bert-cased-768	70.40	0.88	0.82	0.85	0.92	0.77	0.83	0.99	1.00	1.00
bert-uncased-768	29.60	0.91	0.87	0.89	0.89	0.87	0.88	1.00	1.00	1.00
bert-combined-mean-768	29.60	0.91	0.87	0.89	0.89	0.87	0.88	1.00	1.00	1.00
bert-combined-conct-1536	33.00	0.91	0.87	0.89	0.89	0.88	0.89	1.00	1.00	1.00
lexical-300	28.80	0.91	0.87	0.89	0.91	0.86	0.88	0.99	1.00	1.00

Table C-37*Evaluation Metrics of 1BiLSTM(50)-1BiGRU(50N)-CRF (Adam) Per Class*

1BiLSTM(50)-1BiGRU(50N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	24.00	0.91	0.87	0.89	0.89	0.88	0.88	1.00	1.00	1.00
glove-twitter-100	19.60	0.91	0.88	0.89	0.91	0.87	0.89	1.00	1.00	1.00
glove-twitter-200	16.20	0.92	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	23.40	0.91	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	18.60	0.92	0.87	0.89	0.90	0.88	0.89	1.00	1.00	1.00
word2vec-google-news-300	15.00	0.92	0.87	0.89	0.91	0.88	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	18.40	0.92	0.87	0.89	0.91	0.86	0.89	0.99	1.00	1.00
bert-cased-768	31.80	0.89	0.83	0.86	0.91	0.80	0.85	0.99	1.00	1.00
bert-uncased-768	16.00	0.91	0.87	0.89	0.89	0.89	0.89	0.99	1.00	1.00
bert-combined-mean-768	16.00	0.91	0.87	0.89	0.89	0.89	0.89	0.99	1.00	1.00
bert-combined-conct-1536	15.20	0.91	0.88	0.89	0.89	0.89	0.89	1.00	1.00	1.00
lexical-300	17.60	0.92	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00

Table C-38*Evaluation Metrics of 1BiLSTM(75)-1BiGRU(75N)-CRF (Adam) Per Class*

1BiLSTM(75)-1BiGRU(75N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	24.00	0.92	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
glove-twitter-100	18.40	0.92	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
glove-twitter-200	15.60	0.92	0.88	0.90	0.90	0.88	0.89	1.00	1.00	1.00
glove-wiki-gigaword-50	23.00	0.92	0.87	0.90	0.90	0.88	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	15.20	0.92	0.87	0.89	0.91	0.86	0.88	0.99	1.00	1.00
word2vec-google-news-300	19.40	0.92	0.87	0.89	0.91	0.86	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	14.40	0.92	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
bert-cased-768	25.20	0.87	0.82	0.85	0.90	0.75	0.82	0.99	1.00	1.00
bert-uncased-768	14.40	0.91	0.87	0.89	0.90	0.89	0.89	1.00	1.00	1.00
bert-combined-mean-768	14.40	0.91	0.87	0.89	0.90	0.89	0.89	1.00	1.00	1.00
bert-combined-conct-1536	12.40	0.91	0.88	0.89	0.90	0.88	0.89	1.00	1.00	1.00
lexical-300	15.20	0.92	0.87	0.89	0.91	0.86	0.88	0.99	1.00	1.00

Table C-39*Evaluation Metrics of 1BiLSTM(100)-1BiGRU(100N)-CRF (Adam) Per Class*

1BiLSTM(100)-1BiGRU(100N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	40.20	0.91	0.86	0.89	0.89	0.89	0.89	0.99	1.00	1.00
glove-twitter-100	16.80	0.92	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
glove-twitter-200	15.80	0.91	0.88	0.89	0.91	0.88	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	19.80	0.91	0.88	0.90	0.89	0.88	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	13.80	0.91	0.88	0.89	0.89	0.88	0.88	0.99	1.00	1.00
word2vec-google-news-300	12.80	0.92	0.87	0.89	0.91	0.88	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	13.80	0.92	0.87	0.89	0.90	0.88	0.88	0.99	1.00	1.00
bert-cased-768	24.20	0.87	0.82	0.85	0.89	0.76	0.82	0.99	1.00	1.00
bert-uncased-768	13.80	0.91	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
bert-combined-mean-768	13.80	0.91	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
bert-combined-conct-1536	14.60	0.91	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
lexical-300	12.60	0.92	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00

Table C-40*Evaluation Metrics of 2BiLSTM(50)-2BiGRU(50N)-CRF (rmsprop) Per Class*

2BiLSTM(50)-2BiGRU(50N)-CRF (rmsprop)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	70.20	0.86	0.88	0.87	0.86	0.88	0.87	1.00	1.00	1.00
glove-twitter-100	45.40	0.87	0.87	0.87	0.87	0.87	0.87	1.00	1.00	1.00
glove-twitter-200	48.00	0.88	0.88	0.88	0.86	0.88	0.87	1.00	1.00	1.00
glove-wiki-gigaword-50	51.40	0.87	0.89	0.88	0.86	0.88	0.87	1.00	1.00	1.00
glove-wiki-gigaword-300	49.00	0.88	0.87	0.88	0.86	0.88	0.87	1.00	1.00	1.00
word2vec-google-news-300	38.80	0.86	0.87	0.87	0.84	0.88	0.86	1.00	0.99	1.00
fasttext-wiki-news- 300	47.60	0.86	0.88	0.87	0.84	0.89	0.86	1.00	0.99	1.00
bert-cased-768	54.80	0.88	0.82	0.85	0.91	0.81	0.85	0.99	1.00	1.00
bert-uncased-768	54.40	0.86	0.88	0.87	0.83	0.88	0.85	1.00	0.99	0.99
bert-combined-mean-768	45.80	0.87	0.87	0.87	0.84	0.88	0.85	0.99	1.00	0.99
bert-combined-conct-1536	52.60	0.87	0.88	0.87	0.84	0.89	0.86	1.00	0.99	1.00
lexical-300	48.00	0.86	0.88	0.87	0.81	0.89	0.85	1.00	0.99	0.99

Table C-41*Evaluation Metrics of 2BiLSTM(25)-2BiGRU(25)-CRF (Adam) Per Class*

2BiLSTM(25)-2BiGRU(25N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	27.8	0.90	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
glove-twitter-100	22.0	0.91	0.87	0.89	0.91	0.87	0.88	0.99	1.00	1.00
glove-twitter-200	25.8	0.91	0.86	0.89	0.90	0.86	0.88	0.99	1.00	1.00
glove-wiki-gigaword-50	28.2	0.90	0.87	0.88	0.90	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	22.8	0.91	0.86	0.89	0.92	0.86	0.89	0.99	1.00	1.00
word2vec-google-news-300	23.4	0.91	0.86	0.88	0.91	0.86	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	19.0	0.91	0.86	0.88	0.90	0.86	0.88	0.99	1.00	1.00
bert-cased-768	24.8	0.91	0.86	0.88	0.90	0.86	0.88	0.99	1.00	1.00
bert-uncased-768	35.0	0.86	0.81	0.83	0.90	0.78	0.84	0.99	1.00	0.99
bert-combined-mean-768	23.2	0.91	0.86	0.89	0.90	0.88	0.89	0.99	1.00	1.00
bert-combined-conct-1536	23.2	0.91	0.86	0.89	0.90	0.88	0.89	0.99	1.00	1.00
lexical-300	23.0	0.91	0.86	0.88	0.89	0.88	0.89	0.99	1.00	1.00

Table C-42*Evaluation Metrics of 2BiLSTM(50)-2BiGRU(50N)-CRF (Adam) Per Class*

2BiLSTM(50)-2BiGRU(50N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	28.60	0.88	0.87	0.88	0.87	0.89	0.87	1.00	1.00	1.00
glove-twitter-100	29.80	0.88	0.86	0.87	0.87	0.87	0.87	0.99	1.00	1.00
glove-twitter-200	22.00	0.89	0.87	0.88	0.87	0.88	0.88	1.00	1.00	1.00
glove-wiki-gigaword-50	28.20	0.89	0.86	0.88	0.88	0.88	0.88	1.00	1.00	1.00
glove-wiki-gigaword-300	16.20	0.90	0.87	0.88	0.89	0.88	0.88	0.99	1.00	1.00
word2vec-google-news-300	16.00	0.89	0.87	0.88	0.88	0.88	0.88	1.00	1.00	1.00
fasttext-wiki-news- 300	14.20	0.90	0.86	0.88	0.90	0.87	0.88	0.99	1.00	1.00
bert-cased-768	16.40	0.89	0.83	0.86	0.91	0.81	0.86	0.99	1.00	1.00
bert-uncased-768	15.40	0.90	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
bert-combined-mean-768	15.40	0.90	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
bert-combined-conct-1536	16.40	0.91	0.87	0.89	0.89	0.89	0.89	0.99	1.00	1.00
lexical-300	13.80	0.90	0.86	0.88	0.90	0.86	0.88	0.99	1.00	1.00

Table C-43*Evaluation Metrics of 2BiLSTM(75)-2BiGRU(75N)-CRF (Adam) Per Class*

2BiLSTM(75)-2BiGRU(75N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	19.4	0.89	0.88	0.88	0.87	0.89	0.88	1.00	1.00	1.00
glove-twitter-100	13.6	0.90	0.87	0.88	0.89	0.89	0.89	0.99	1.00	1.00
glove-twitter-200	14.4	0.90	0.87	0.89	0.89	0.89	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	17.0	0.90	0.87	0.88	0.89	0.89	0.88	0.99	1.00	1.00
glove-wiki-gigaword-300	13.0	0.91	0.87	0.89	0.90	0.87	0.89	0.99	1.00	1.00
word2vec-google-news-300	14.2	0.90	0.87	0.88	0.88	0.88	0.88	1.00	1.00	1.00
fasttext-wiki-news- 300	13.2	0.90	0.86	0.88	0.90	0.87	0.89	1.00	1.00	1.00
bert-cased-768	14.0	0.89	0.86	0.88	0.89	0.87	0.88	0.99	1.00	1.00
bert-uncased-768	16.2	0.87	0.82	0.85	0.90	0.80	0.85	0.99	1.00	0.99
bert-combined-mean-768	13.6	0.91	0.88	0.89	0.89	0.89	0.89	0.99	1.00	1.00
bert-combined-conct-1536	13.6	0.91	0.88	0.89	0.89	0.89	0.89	0.99	1.00	1.00
lexical-300	14.4	0.91	0.87	0.89	0.90	0.88	0.89	1.00	1.00	1.00

Table C-44*Evaluation Metrics of 2BiLSTM(100)-2BiGRU(100N)-CRF (Adam) Per Class*

2BiLSTM(100)-2BiGRU(100N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	17.0	0.89	0.88	0.88	0.87	0.91	0.89	1.00	1.00	1.00
glove-twitter-100	33.2	0.90	0.87	0.88	0.89	0.88	0.89	0.99	1.00	1.00
glove-twitter-200	13.8	0.91	0.87	0.89	0.89	0.88	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	16.8	0.89	0.88	0.89	0.89	0.88	0.88	1.00	1.00	1.00
glove-wiki-gigaword-300	13.2	0.90	0.87	0.88	0.88	0.88	0.88	0.99	1.00	1.00
word2vec-google-news-300	14.4	0.90	0.87	0.88	0.87	0.88	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	11.4	0.90	0.86	0.88	0.90	0.87	0.88	0.99	1.00	1.00
bert-cased-768	11.6	0.90	0.87	0.88	0.90	0.88	0.89	1.00	1.00	1.00
bert-uncased-768	12.2	0.88	0.83	0.85	0.91	0.81	0.86	0.99	1.00	0.99
bert-combined-mean-768	14.0	0.90	0.86	0.88	0.88	0.87	0.88	0.99	1.00	1.00
bert-combined-conct-1536	14.0	0.90	0.86	0.88	0.88	0.87	0.88	0.99	1.00	1.00
lexical-300	13.4	0.90	0.87	0.88	0.88	0.88	0.88	1.00	1.00	1.00

C2.3 Further Exploration of Results of Deep Learning Models Using Recurrent Layers and Convolutional Neural Network Layers

Table C-45

Evaluation Metrics of 1BiLSTM(25N)-1CNN-CRF (Adam) Per Class

1BiLSTM(25N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	25.80	0.90	0.87	0.89	0.90	0.88	0.89	1.00	1.00	1.00
glove-twitter-100	17.20	0.91	0.87	0.89	0.90	0.86	0.88	0.99	1.00	1.00
glove-twitter-200	13.60	0.91	0.86	0.88	0.92	0.85	0.88	0.99	1.00	1.00
glove-wiki-gigaword-50	19.80	0.91	0.87	0.89	0.91	0.86	0.88	0.99	1.00	1.00
glove-wiki-gigaword-300	15.60	0.91	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
word2vec-google-news-300	14.00	0.91	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	16.80	0.90	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
bert-cased-768	64.80	0.81	0.80	0.81	0.87	0.72	0.79	0.99	1.00	0.99
bert-uncased-768	13.80	0.90	0.86	0.88	0.89	0.85	0.87	0.99	1.00	1.00
bert-combined-mean-768	13.80	0.90	0.86	0.88	0.89	0.85	0.87	0.99	1.00	1.00
bert-combined-conct-1536	14.60	0.90	0.87	0.88	0.90	0.84	0.87	0.99	1.00	1.00
lexical-300	17.40	0.90	0.86	0.87	0.91	0.84	0.87	0.99	1.00	1.00

Table C-46*Evaluation Metrics of 1BiLSTM(50N)-1CNN-CRF (Adam) Per Class*

1BiLSTM(50N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	25.40	0.90	0.88	0.89	0.90	0.89	0.89	1.00	1.00	1.00
glove-twitter-100	17.20	0.90	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
glove-twitter-200	13.80	0.90	0.86	0.88	0.90	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	22.00	0.91	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	13.80	0.92	0.87	0.89	0.92	0.86	0.89	0.99	1.00	1.00
word2vec-google-news-300	13.00	0.90	0.86	0.88	0.90	0.85	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	15.40	0.91	0.86	0.88	0.92	0.84	0.88	0.99	1.00	1.00
bert-cased-768	40.00	0.84	0.78	0.81	0.86	0.74	0.80	0.99	1.00	0.99
bert-uncased-768	14.40	0.91	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
bert-combined-mean-768	14.40	0.91	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
bert-combined-conct-1536	15.20	0.90	0.86	0.88	0.90	0.85	0.87	0.99	1.00	1.00
lexical-300	17.00	0.90	0.86	0.87	0.90	0.85	0.87	0.99	1.00	1.00

Table C-47*Evaluation Metrics of 1BiLSTM(75N)-1CNN-CRF (Adam) Per Class*

1BiLSTM(75N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	24.20	0.91	0.87	0.89	0.91	0.85	0.88	0.99	1.00	1.00
glove-twitter-100	16.20	0.91	0.88	0.89	0.91	0.87	0.89	0.99	1.00	1.00
glove-twitter-200	14.60	0.91	0.86	0.89	0.91	0.86	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	20.00	0.90	0.87	0.88	0.91	0.86	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	13.40	0.90	0.87	0.89	0.91	0.87	0.88	0.99	1.00	1.00
word2vec-google-news-300	14.20	0.89	0.87	0.88	0.91	0.85	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	16.20	0.90	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
bert-cased-768	30.40	0.79	0.76	0.78	0.77	0.67	0.71	0.99	1.00	0.99
bert-uncased-768	16.60	0.91	0.86	0.88	0.90	0.85	0.87	0.99	1.00	1.00
bert-combined-mean-768	16.60	0.91	0.86	0.88	0.90	0.85	0.87	0.99	1.00	1.00
bert-combined-conct-1536	14.80	0.91	0.86	0.89	0.90	0.85	0.88	0.99	1.00	1.00
lexical-300	14.40	0.91	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00

Table C-48*Evaluation Metrics of 1BiLSTM(100N)-1CNN-CRF (Adam) Per Class*

1BiLSTM(100N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	22.20	0.90	0.87	0.89	0.89	0.87	0.88	1.00	1.00	1.00
glove-twitter-100	16.00	0.90	0.88	0.89	0.89	0.88	0.88	1.00	1.00	1.00
glove-twitter-200	13.60	0.91	0.87	0.89	0.90	0.85	0.88	0.99	1.00	1.00
glove-wiki-gigaword-50	20.00	0.89	0.88	0.89	0.89	0.89	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	11.80	0.91	0.86	0.89	0.90	0.86	0.88	0.99	1.00	1.00
word2vec-google-news-300	13.20	0.91	0.86	0.88	0.90	0.86	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	16.80	0.90	0.86	0.88	0.90	0.85	0.88	0.99	1.00	1.00
bert-cased-768	31.60	0.82	0.80	0.81	0.86	0.75	0.81	0.99	1.00	0.99
bert-uncased-768	15.60	0.90	0.86	0.88	0.90	0.86	0.88	0.99	1.00	1.00
bert-combined-mean-768	15.60	0.90	0.86	0.88	0.90	0.86	0.88	0.99	1.00	1.00
bert-combined-conct-1536	12.20	0.90	0.87	0.88	0.90	0.86	0.88	0.99	1.00	1.00
lexical-300	15.40	0.91	0.86	0.88	0.91	0.84	0.87	0.99	1.00	1.00

Table C-49*Evaluation Metrics of 1BiGRU(25N)-1CNN-CRF (Adam) Per Class*

1BiGRU(25N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	26.00	0.91	0.87	0.89	0.91	0.87	0.89	1.00	1.00	1.00
glove-twitter-100	17.20	0.91	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
glove-twitter-200	14.80	0.91	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	21.40	0.92	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	14.60	0.91	0.87	0.89	0.92	0.86	0.89	0.99	1.00	1.00
word2vec-google-news-300	13.20	0.91	0.86	0.89	0.91	0.86	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	12.80	0.91	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
bert-cased-768	77.20	0.84	0.76	0.79	0.84	0.74	0.79	0.99	1.00	0.99
bert-uncased-768	12.20	0.89	0.82	0.86	0.88	0.84	0.86	0.99	1.00	1.00
bert-combined-mean-768	12.20	0.90	0.87	0.88	0.90	0.86	0.88	0.99	1.00	1.00
bert-combined-conct-1536	15.00	0.91	0.87	0.88	0.92	0.85	0.88	0.99	1.00	1.00
lexical-300	15.80	0.91	0.86	0.88	0.91	0.84	0.88	0.99	1.00	1.00

Table C-50*Evaluation Metrics of 1BiGRU(50N)-1CNN-CRF (Adam) Per Class*

1BiGRU(50N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	23.80	0.90	0.87	0.88	0.89	0.88	0.89	1.00	1.00	1.00
glove-twitter-100	16.00	0.90	0.87	0.88	0.91	0.87	0.89	0.99	1.00	1.00
glove-twitter-200	13.80	0.92	0.87	0.89	0.92	0.88	0.90	0.99	1.00	1.00
glove-wiki-gigaword-50	18.80	0.90	0.87	0.88	0.90	0.88	0.89	1.00	1.00	1.00
glove-wiki-gigaword-300	12.80	0.91	0.86	0.89	0.91	0.86	0.89	0.99	1.00	1.00
word2vec-google-news-300	12.80	0.91	0.86	0.89	0.90	0.87	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	12.60	0.90	0.87	0.88	0.91	0.86	0.88	0.99	1.00	1.00
bert-cased-768	63.40	0.81	0.80	0.81	0.86	0.71	0.78	0.99	1.00	0.99
bert-uncased-768	12.40	0.90	0.87	0.88	0.91	0.86	0.88	0.99	1.00	1.00
bert-combined-mean-768	12.40	0.90	0.87	0.88	0.91	0.86	0.88	0.99	1.00	1.00
bert-combined-conct-1536	13.40	0.90	0.88	0.89	0.92	0.85	0.88	0.99	1.00	1.00
lexical-300	12.20	0.91	0.86	0.89	0.91	0.84	0.88	0.99	1.00	1.00

Table C-51*Evaluation Metrics of 1BiGRU(75N)-1CNN-CRF (Adam) Per Class*

1BiGRU(75N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	20.80	0.91	0.87	0.89	0.91	0.86	0.89	1.00	1.00	1.00
glove-twitter-100	15.60	0.91	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
glove-twitter-200	32.60	0.92	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	18.40	0.91	0.86	0.89	0.91	0.89	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	13.00	0.91	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
word2vec-google-news-300	11.40	0.92	0.87	0.89	0.91	0.87	0.89	1.00	1.00	1.00
fasttext-wiki-news- 300	13.60	0.91	0.86	0.88	0.91	0.86	0.88	0.99	1.00	1.00
bert-cased-768	34.00	0.81	0.78	0.79	0.86	0.72	0.78	0.99	1.00	0.99
bert-uncased-768	12.80	0.90	0.87	0.88	0.91	0.85	0.88	0.99	1.00	1.00
bert-combined-mean-768	12.80	0.90	0.87	0.88	0.91	0.85	0.88	0.99	1.00	1.00
bert-combined-conct-1536	12.20	0.90	0.87	0.89	0.91	0.85	0.88	0.99	1.00	1.00
lexical-300	12.00	0.91	0.86	0.89	0.92	0.85	0.88	0.99	1.00	1.00

Table C-52*Evaluation Metrics of 1BiGRU(100N)-1CNN-CRF (Adam) Per Class*

1BiGRU(100N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	24.00	0.91	0.88	0.89	0.90	0.89	0.90	1.00	1.00	1.00
glove-twitter-100	15.60	0.91	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
glove-twitter-200	13.20	0.91	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	21.80	0.92	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	14.40	0.90	0.87	0.89	0.89	0.88	0.88	0.99	1.00	1.00
word2vec-google-news-300	12.40	0.92	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	12.20	0.91	0.87	0.89	0.91	0.86	0.88	0.99	1.00	1.00
bert-cased-768	31.80	0.81	0.78	0.80	0.84	0.73	0.79	0.99	1.00	0.99
bert-uncased-768	12.20	0.91	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
bert-combined-mean-768	12.20	0.91	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
bert-combined-conct-1536	13.00	0.90	0.87	0.89	0.91	0.85	0.88	0.99	1.00	1.00
lexical-300	13.20	0.91	0.86	0.88	0.91	0.84	0.88	0.99	1.00	1.00

Table C-53*Evaluation Metrics of 1BiLSTM(25N)-1BiGRU(25N)-1CNN -CRF (Adam) Per Class*

1BiLSTM(25N)-1BiGRU(25N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	26.80	0.91	0.87	0.89	0.90	0.88	0.89	1.00	1.00	1.00
glove-twitter-100	16.20	0.91	0.87	0.89	0.89	0.88	0.89	1.00	1.00	1.00
glove-twitter-200	15.80	0.91	0.86	0.89	0.91	0.86	0.88	0.99	1.00	1.00
glove-wiki-gigaword-50	17.80	0.90	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	13.80	0.91	0.87	0.89	0.92	0.86	0.89	0.99	1.00	1.00
word2vec-google-news-300	13.60	0.91	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	13.20	0.91	0.86	0.89	0.91	0.85	0.88	0.99	1.00	1.00
bert-cased-768	29.00	0.69	0.62	0.64	0.65	0.56	0.60	0.99	1.00	0.99
bert-uncased-768	15.20	0.91	0.86	0.88	0.90	0.85	0.88	0.99	1.00	1.00
bert-combined-mean-768	15.20	0.91	0.86	0.88	0.90	0.85	0.88	0.99	1.00	1.00
bert-combined-conct-1536	15.40	0.90	0.87	0.88	0.90	0.85	0.87	0.99	1.00	1.00
lexical-300	13.40	0.91	0.86	0.89	0.91	0.85	0.88	0.99	1.00	1.00

Table C-54*Evaluation Metrics of 1BiLSTM(50N)-1BiGRU(50N)-1CNN -CRF (Adam) Per Class*

1BiLSTM(50N)-1BiGRU(50N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	22.40	0.91	0.87	0.89	0.90	0.88	0.90	0.99	1.00	1.00
glove-twitter-100	14.20	0.91	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
glove-twitter-200	13.80	0.91	0.87	0.89	0.92	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	19.40	0.90	0.88	0.89	0.89	0.89	0.89	1.00	1.00	1.00
glove-wiki-gigaword-300	12.40	0.91	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
word2vec-google-news-300	13.00	0.91	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	12.80	0.90	0.87	0.88	0.91	0.86	0.89	0.99	1.00	1.00
bert-cased-768	33.40	0.81	0.82	0.80	0.85	0.73	0.77	0.99	1.00	0.99
bert-uncased-768	12.20	0.91	0.87	0.89	0.91	0.85	0.88	0.99	1.00	1.00
bert-combined-mean-768	12.20	0.91	0.87	0.89	0.91	0.85	0.88	0.99	1.00	1.00
bert-combined-conct-1536	13.40	0.90	0.87	0.88	0.91	0.86	0.88	0.99	1.00	1.00
lexical-300	13.40	0.91	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00

Table C-55*Evaluation Metrics of 1BiLSTM(75N)-1BiGRU(75N)-1CNN -CRF (Adam) Per Class*

1BiLSTM(75N)-1BiGRU(75N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	20.00	0.91	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
glove-twitter-100	16.20	0.92	0.87	0.89	0.92	0.88	0.90	0.99	1.00	1.00
glove-twitter-200	14.40	0.92	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	19.00	0.91	0.87	0.89	0.89	0.89	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	13.40	0.91	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
word2vec-google-news-300	12.60	0.92	0.86	0.89	0.91	0.87	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	11.80	0.91	0.87	0.89	0.93	0.86	0.89	0.99	1.00	1.00
bert-cased-768	28.80	0.83	0.79	0.81	0.84	0.74	0.79	0.99	1.00	0.99
bert-uncased-768	12.60	0.90	0.87	0.89	0.91	0.85	0.88	0.99	1.00	1.00
bert-combined-mean-768	12.60	0.90	0.87	0.89	0.91	0.85	0.88	0.99	1.00	1.00
bert-combined-conct-1536	11.40	0.90	0.87	0.89	0.91	0.86	0.88	0.99	1.00	1.00
lexical-300	10.80	0.91	0.86	0.89	0.91	0.85	0.88	0.99	1.00	1.00

Table C-56*Evaluation Metrics of 1BiLSTM(100N)-1BiGRU(100N)-1CNN -CRF (Adam) Per Class*

1BiLSTM(100N)-1BiGRU(100N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	20.40	0.89	0.88	0.89	0.88	0.88	0.88	1.00	1.00	1.00
glove-twitter-100	15.00	0.91	0.88	0.90	0.90	0.89	0.89	0.99	1.00	1.00
glove-twitter-200	13.20	0.90	0.87	0.89	0.89	0.88	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	17.20	0.92	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	13.00	0.91	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
word2vec-google-news-300	13.00	0.91	0.87	0.89	0.89	0.89	0.89	1.00	1.00	1.00
fasttext-wiki-news- 300	13.80	0.91	0.87	0.89	0.91	0.87	0.89	0.99	1.00	1.00
bert-cased-768	32.80	0.86	0.79	0.82	0.85	0.74	0.79	0.99	1.00	0.99
bert-uncased-768	11.00	0.90	0.86	0.88	0.90	0.86	0.88	0.99	1.00	1.00
bert-combined-mean-768	11.00	0.90	0.86	0.88	0.90	0.86	0.88	0.99	1.00	1.00
bert-combined-conct-1536	20.80	0.90	0.88	0.89	0.90	0.87	0.89	1.00	1.00	1.00
lexical-300	13.00	0.90	0.87	0.89	0.91	0.85	0.88	0.99	1.00	1.00

Table C-57*Evaluation Metrics of 2BiGRU(75N)-1CNN-CRF (rmsprop) Per Class*

2BiGRU(75N)-1CNN-CRF (rmsprop)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	58	0.88	0.89	0.88	0.87	0.89	0.88	1.00	1.00	1.00
glove-twitter-100	42	0.88	0.88	0.88	0.86	0.89	0.87	1.00	0.99	1.00
glove-twitter-200	45	0.88	0.88	0.88	0.86	0.88	0.87	1.00	0.99	1.00
glove-wiki-gigaword-50	56	0.87	0.88	0.88	0.85	0.89	0.87	1.00	1.00	1.00
glove-wiki-gigaword-300	40	0.87	0.89	0.88	0.87	0.88	0.87	1.00	1.00	1.00
word2vec-google-news-300	36	0.87	0.88	0.87	0.85	0.89	0.87	1.00	0.99	0.99
fasttext-wiki-news- 300	44	0.85	0.88	0.87	0.84	0.89	0.86	1.00	0.99	1.00
bert-cased-768	35	0.88	0.87	0.88	0.86	0.87	0.86	1.00	1.00	1.00
bert-uncased-768	116	0.80	0.67	0.65	0.83	0.65	0.66	0.98	0.99	0.99
bert-combined-mean-768	37	0.85	0.89	0.87	0.86	0.86	0.86	1.00	1.00	0.99
bert-combined-conct-1536	37	0.85	0.89	0.87	0.86	0.86	0.86	1.00	1.00	0.99
lexical-300	37	0.85	0.89	0.87	0.86	0.89	0.87	1.00	0.99	1.00

Table C-58*Evaluation Metrics of 2BiGRU(25N)-1CNN-CRF (Adam) Per Class*

2BiGRU(25N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	18.80	0.90	0.86	0.88	0.88	0.89	0.88	0.99	1.00	1.00
glove-twitter-100	15.80	0.90	0.87	0.88	0.89	0.88	0.88	1.00	1.00	1.00
glove-twitter-200	14.80	0.90	0.87	0.89	0.90	0.88	0.90	1.00	1.00	1.00
glove-wiki-gigaword-50	17.00	0.90	0.87	0.89	0.90	0.89	0.89	1.00	1.00	1.00
glove-wiki-gigaword-300	12.40	0.91	0.87	0.89	0.91	0.88	0.89	0.99	1.00	1.00
word2vec-google-news-300	12.60	0.90	0.86	0.88	0.89	0.89	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	14.60	0.90	0.86	0.88	0.91	0.87	0.89	0.99	1.00	1.00
bert-cased-768	20.40	0.73	0.71	0.72	0.74	0.66	0.70	0.99	0.99	0.99
bert-uncased-768	14.00	0.90	0.87	0.88	0.90	0.87	0.88	0.99	1.00	1.00
bert-combined-mean-768	14.00	0.90	0.87	0.88	0.90	0.87	0.88	0.99	1.00	1.00
bert-combined-conct-1536	14.60	0.90	0.86	0.89	0.89	0.87	0.88	0.99	1.00	1.00
lexical-300	13.00	0.91	0.86	0.88	0.91	0.86	0.89	0.99	1.00	1.00

Table C-59*Evaluation Metrics of 2BiGRU(50N)-1CNN-CRF (Adam) Per Class*

2BiGRU(50N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	19.40	0.90	0.87	0.89	0.89	0.89	0.89	1.00	1.00	1.00
glove-twitter-100	13.80	0.91	0.87	0.89	0.91	0.88	0.89	1.00	1.00	1.00
glove-twitter-200	13.00	0.91	0.87	0.89	0.90	0.88	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	16.40	0.91	0.87	0.89	0.89	0.89	0.89	0.99	1.00	1.00
glove-wiki-gigaword-300	12.20	0.91	0.87	0.89	0.90	0.88	0.89	1.00	1.00	1.00
word2vec-google-news-300	11.40	0.90	0.86	0.88	0.90	0.87	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	13.20	0.90	0.86	0.88	0.90	0.87	0.88	0.99	1.00	1.00
bert-cased-768	22.60	0.84	0.78	0.81	0.87	0.77	0.81	0.99	1.00	1.00
bert-uncased-768	11.20	0.91	0.86	0.88	0.90	0.87	0.88	0.99	1.00	1.00
bert-combined-mean-768	11.20	0.91	0.86	0.88	0.90	0.87	0.88	0.99	1.00	1.00
bert-combined-conct-1536	12.00	0.91	0.87	0.89	0.91	0.88	0.89	0.99	1.00	1.00
lexical-300	12.20	0.91	0.85	0.88	0.91	0.84	0.88	0.99	1.00	1.00

Table C-60*Evaluation Metrics of 2BiGRU(75N)-1CNN-CRF (Adam) Per Class*

2BiGRU(75N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	22.00	0.90	0.87	0.88	0.89	0.89	0.89	1.00	1.00	1.00
glove-twitter-100	14.80	0.91	0.87	0.89	0.90	0.87	0.89	0.99	1.00	1.00
glove-twitter-200	13.20	0.90	0.87	0.88	0.89	0.89	0.89	1.00	1.00	1.00
glove-wiki-gigaword-50	16.60	0.91	0.86	0.88	0.91	0.87	0.89	1.00	1.00	1.00
glove-wiki-gigaword-300	12.20	0.90	0.87	0.88	0.89	0.88	0.89	0.99	1.00	1.00
word2vec-google-news-300	10.40	0.92	0.87	0.89	0.89	0.89	0.89	0.99	1.00	1.00
fasttext-wiki-news- 300	13.40	0.90	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
bert-cased-768	19.20	0.84	0.76	0.80	0.85	0.79	0.81	0.99	1.00	0.99
bert-uncased-768	12.60	0.90	0.86	0.88	0.89	0.87	0.88	0.99	1.00	1.00
bert-combined-mean-768	12.60	0.90	0.86	0.88	0.89	0.87	0.88	0.99	1.00	1.00
bert-combined-conct-1536	11.60	0.91	0.85	0.88	0.89	0.87	0.88	0.99	1.00	1.00
lexical-300	14.20	0.90	0.86	0.88	0.91	0.84	0.87	0.99	1.00	1.00

Table C-61*Evaluation Metrics of 2BiGRU(100N)-1CNN-CRF (Adam) Per Class*

2BiGRU(100N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	21.2	0.92	0.87	0.89	0.90	0.88	0.89	1.00	1.00	1.00
glove-twitter-100	13.2	0.90	0.87	0.89	0.88	0.89	0.88	1.00	1.00	1.00
glove-twitter-200	13.6	0.91	0.87	0.89	0.89	0.88	0.89	0.99	1.00	1.00
glove-wiki-gigaword-50	15.8	0.90	0.87	0.88	0.89	0.89	0.89	1.00	1.00	1.00
glove-wiki-gigaword-300	13.0	0.90	0.87	0.88	0.89	0.88	0.88	0.99	1.00	1.00
word2vec-google-news-300	12.4	0.90	0.86	0.88	0.89	0.87	0.88	0.99	1.00	1.00
fasttext-wiki-news- 300	11.8	0.90	0.86	0.88	0.91	0.86	0.89	0.99	1.00	1.00
bert-cased-768	10.8	0.89	0.85	0.87	0.91	0.82	0.86	0.99	1.00	1.00
bert-uncased-768	18.4	0.84	0.79	0.81	0.87	0.79	0.82	0.99	1.00	0.99
bert-combined-mean-768	12.2	0.90	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
bert-combined-conct-1536	12.2	0.90	0.86	0.88	0.91	0.85	0.88	0.99	1.00	1.00
lexical-300	11.8	0.90	0.86	0.88	0.91	0.86	0.88	0.99	1.00	1.00

C2.4 Best Results Per Experimented Model on *MenuNER* Dataset

Table C-62 presents the optimal results for each evaluated model on the *MenuNER* dataset, detailing the highest F1-score achieved, the number of trainable parameters involved, and the word embedding model utilized to attain these results.

Table C-62

Best Achieved Results Per Experimented Model on MenuNER Dataset

Model Architectures		F1-Score	Trainable Parameters	Word Embeddings
1BiLSTM-CRF	BiLSTM(50N)-CRF (rmsprop) (Non-Trainable)	86.40	177,480	glove-wiki-gigaword-300
	BiLSTM(50N)-CRF (rmsprop)	91.13	619,730	glove-wiki-gigaword-50
	BiLSTM(25N)-CRF (Adam)	91.53	579,030	glove-wiki-gigaword-50
			1,131,280	glove-twitter-100
	BiLSTM(50N)-CRF (Adam)	92.53	338,605	glove-twitter-25
			619,730	glove-wiki-gigaword-50
	BiLSTM(75N)-CRF (Adam)	92.73	1,242,680	glove-twitter-100
	BiLSTM(100N)-CRF (Adam)	92.93	2,477,880	glove-twitter-200
	BiLSTM(125N)-CRF (Adam)	92.60	1,394,080	glove-twitter-100
			2,578,580	glove-twitter-200
	BiLSTM(150N)-CRF (Adam)	92.80	3,893,780	glove-wiki-gigaword-300
1BiGRU-CRF	BiGRU(50N)-CRF (rmsprop) (Non-Trainable)	88.53	61,180	glove-wiki-gigaword-50
	BiGRU(50N)-CRF (rmsprop)	91.40	324,805	glove-twitter-25
	BiGRU(25N)-CRF (Adam)	92.53	297,255	glove-twitter-25

Model Architectures		F1-Score	Trainable Parameters	Word Embeddings
	BiGRU(50N)-CRF (Adam)	92.73	8,605,540	bert-uncased-768
	BiGRU(75N)-CRF (Adam)	92.87	3,465,980	glove-wiki-gigaword-300
	BiGRU(100N)-CRF (Adam)	93.07	1,260,780	glove-twitter-100
	BiGRU(125N)-CRF (Adam)	93.00	742,330	glove-wiki-gigaword-50
	BiGRU(150N)-CRF (Adam)	92.60	1,394,080	glove-twitter-100
			3,763,080	glove-wiki-gigaword-300
2BiLSTM-CRF	2BiLSTM(50N)-CRF (rmsprop)	91.53	1,242,380	glove-twitter-100
			3,491,380	glove-wiki-gigaword-300
	2BiLSTM(25N)-CRF (Adam)	91.53	17,007,100	bert-combined-concct-1536
	2BiLSTM(50N)-CRF (Adam)	92.07	2,366,880	glove-twitter-200
	2BiLSTM(75N)-CRF (Adam)	92.33	1,378,280	glove-twitter-100
	2BiLSTM(100N)-CRF (Adam)	92.33	971,930	glove-wiki-gigaword-50
2BiGRU-CRF	2BiGRU(50N)-CRF (rmsprop)	91.80	3,435,280	glove-wiki-gigaword-300
	2BiGRU(25N)-CRF (Adam)	92.07	16,922,250	bert-combined-concct-1536
	2BiGRU(50N)-CRF (Adam)	91.73	3,435,280	fasttext-300
	2BiGRU(75N)-CRF (Adam)	92.53	462,005	glove-twitter-25
	2BiGRU(100N)-CRF (Adam)	92.40	2,586,480	glove-twitter-200
1BiLSTM-1BiGRU-CRF	1BiLSTM(50N)-1BiGRU(50N)-CRF (rmsprop)	91.93	2,406,580	glove-twitter-200
1BiLSTM-1BiGRU-CRF	1BiLSTM(25N)-1BiGRU(25N)-CRF (Adam)	92.93	1,162,580	glove-twitter-100
			2,282,080	glove-twitter-200
			602,830	glove-wiki-gigaword-50
	1BiLSTM(50N)-1BiGRU(50N)-CRF	92.80	1,252,080	glove-twitter-100

Model Architectures		F1-Score	Trainable Parameters	Word Embeddings
	(Adam)	92.80	3,561,080	word2vec-300
	1BiLSTM(75N)-1BiGRU(75N)-CRF	92.87	2,548,580	glove-twitter-200
	(Adam)			
	1BiLSTM(100N)-1BiGRU(100N)-CRF	92.80	1,483,580	glove-twitter-100
	(Adam)		871,330	glove-wiki-gigaword-50
			19,067,400	bert-combined-concct-1536
2BiLSTM-	2BiLSTM(50N)-2BiGRU(50N)-CRF	91.47	2,512,580	glove-twitter-200,
2BiGRU-	(rmsprop)		3,667,080	glove-wiki-gigaword-300
CRF	2BiLSTM(25N)-2BiGRU(25N)-CRF	92.40	3,428,330	glove-wiki-gigaword-300
	(Adam)			
	2BiLSTM(50N)-2BiGRU(50N)-CRF	92.60	9,070,140	bert-uncased-768
	(Adam)		9,070,140	bert-combined-mean-768
	2BiLSTM(75N)-2BiGRU(75N)-CRF	92.67	9,542,690	bert-uncased-768
	(Adam)		9,542,690	bert-combined-mean-768
	2BiLSTM(100N)-2BiGRU(100N)-CRF	92.47	3,130,080	glove-twitter-200
	(Adam)			
1BiLSTM-	1BiLSTM(25N)-1CNN-CRF (Adam)	92.47	337,555	glove-twitter-25
1CNN-	1BiLSTM(50N)-1CNN-CRF (Adam)	92.67	373,255	glove-twitter-25
CRF	1BiLSTM(75N)-1CNN-CRF (Adam)	92.53	1,297,280	glove-twitter-100
	1BiLSTM(100N)-1CNN-CRF (Adam)	92.33	730,680	glove-wiki-gigaword-50
1BiGRU-	1BiGRU(25N)-1CNN-CRF (Adam)	92.67	3,438,030	glove-wiki-gigaword-300
1CNN-	1BiGRU(50N)-1CNN-CRF (Adam)	92.87	2,362,330	glove-twitter-200
CRF	1BiGRU(75N)-1CNN-CRF (Adam)	92.67	2,421,080	glove-twitter-200

Model Architectures		F1-Score	Trainable Parameters	Word Embeddings
			681,830	glove-wiki-gigaword-50
	1BiGRU(100N)-1CNN-CRF (Adam)	92.87	437,055	glove-twitter-25
1BiLSTM-	1BiLSTM(25N)-1BiGRU(25N)-1CNN-	92.67	357,605	glove-twitter-25
1BiGRU-	CRF (Adam)		3,518,730	word2vec
1CNN-	1BiLSTM(50N)-1BiGRU(50N)-1CNN-	92.93	420,855	glove-twitter-25
CRF	CRF (Adam)			
	1BiLSTM(75N)-1BiGRU(75N)-1CNN-	92.87	1,418,780	glove-twitter-100
	CRF (Adam)			
	1BiLSTM(100N)-1BiGRU(100N)-	92.67	916,030	glove-wiki-gigaword-50
	1CNN-CRF (Adam)			
2BiGRU-	2BiGRU(75N)-1CNN-CRF (rmsprop)	92.00	496,655	glove-twitter-25
1CNN-	2BiGRU(25N)-1CNN-CRF (Adam)	92.67	2,320,080	glove-twitter-200
CRF	2BiGRU(50N)-1CNN-CRF (Adam)	92.53	1,263,430	glove-twitter-100
			3,552,430	glove-wiki-gigaword-300
			17,698,450	bert-combined-concat
	2BiGRU(75N)-1CNN-CRF (Adam)	92.53	3,685,280	word2vec-300
	2BiGRU(100N)-1CNN-CRF (Adam)	92.60	618,255	glove-twitter-25

C.3 Further Analysis of Deep Learning Models With Trainable Word Embeddings on *ReviewsDB* Dataset

This section provides an in-depth analysis of the evaluation metrics used to assess the performance of Named Entity Recognition (NER) deep learning models. These models were trained using five-fold cross-validation on the *ReviewsDB* dataset.

C3.1 Further Exploration on Results of Models With Single Type of Recurrent Layers

Table C-63

Evaluation Metrics of 1BiLSTM(50)-CRF (rmsprop) With Fixed Embeddings Per Class

1BiLSTM(50)-CRF (rmsprop) - Fixed Word Embeddings										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	154.8	0.61	0.57	0.59	0.64	0.68	0.65	0.98	0.98	0.98
glove-twitter-100	146.2	0.59	0.57	0.58	0.64	0.58	0.62	0.98	0.98	0.98
glove-twitter-200	121.8	0.58	0.58	0.58	0.64	0.66	0.64	0.98	0.98	0.98
glove-wiki-gigaword-50	135.0	0.62	0.58	0.59	0.65	0.64	0.64	0.98	0.98	0.98
glove-wiki-gigaword-300	127.8	0.60	0.56	0.58	0.68	0.59	0.62	0.98	0.98	0.98
word2vec-google-news-300	155.6	0.61	0.56	0.59	0.66	0.64	0.65	0.98	0.98	0.98
fasttext-wiki-news-subwords-300	137.0	0.68	0.52	0.58	0.69	0.60	0.64	0.97	0.99	0.98

Table C-64*Evaluation Metrics of 1BiLSTM(50)-CRF (rmsprop) Per Class*

1BiLSTM(50)-CRF (rmsprop)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	102.2	0.65	0.76	0.69	0.72	0.73	0.72	0.99	0.98	0.98
glove-twitter-100	77.0	0.67	0.73	0.70	0.75	0.68	0.71	0.99	0.98	0.99
glove-twitter-200	74.4	0.64	0.76	0.69	0.73	0.70	0.71	0.99	0.98	0.98
glove-wiki-gigaword-50	89.6	0.67	0.75	0.70	0.74	0.70	0.72	0.99	0.98	0.98
glove-wiki-gigaword-300	70.0	0.66	0.75	0.70	0.73	0.71	0.72	0.99	0.98	0.98
word2vec-google-news-300	71.2	0.64	0.73	0.69	0.74	0.70	0.71	0.99	0.98	0.98
fasttext-wiki-news-subwords-300	72.8	0.64	0.74	0.68	0.71	0.72	0.71	0.99	0.98	0.98
Lexical-300	70.2	0.63	0.73	0.68	0.70	0.70	0.70	0.99	0.98	0.98

Table C-65*Evaluation Metrics of 1BiLSTM(25)-CRF (Adam) Per Class*

1BiLSTM(25)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	158.0	0.71	0.69	0.70	0.79	0.66	0.71	0.98	0.99	0.99
glove-twitter-100	170.2	0.71	0.66	0.68	0.80	0.61	0.69	0.98	0.99	0.99
glove-twitter-200	144.8	0.71	0.68	0.69	0.78	0.60	0.67	0.98	0.99	0.99
glove-wiki-gigaword-50	170.8	0.70	0.67	0.68	0.78	0.60	0.68	0.98	0.99	0.99
glove-wiki-gigaword-300	128.6	0.70	0.64	0.67	0.79	0.58	0.67	0.98	0.99	0.99
word2vec-google-news-300	116.4	0.71	0.66	0.69	0.78	0.59	0.67	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	89.8	0.71	0.66	0.68	0.80	0.59	0.67	0.98	0.99	0.99
Lexical-300	83.6	0.71	0.69	0.70	0.79	0.62	0.69	0.98	0.99	0.99

Table C-66*Evaluation Metrics of 1BiLSTM(50)-CRF (Adam) Per Class*

1BiLSTM(50)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	71.2	0.72	0.72	0.72	0.78	0.69	0.73	0.99	0.99	0.99
glove-twitter-100	57.2	0.72	0.71	0.71	0.79	0.67	0.72	0.98	0.99	0.99
glove-twitter-200	49.0	0.72	0.70	0.70	0.82	0.64	0.71	0.98	0.99	0.99
glove-wiki-gigaword-50	83.6	0.72	0.70	0.71	0.79	0.66	0.71	0.98	0.99	0.99
glove-wiki-gigaword-300	49.6	0.71	0.68	0.70	0.80	0.63	0.71	0.98	0.99	0.99
word2vec-google-news-300	55.8	0.73	0.70	0.71	0.80	0.63	0.71	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	45.8	0.72	0.70	0.71	0.80	0.65	0.71	0.98	0.99	0.99
Lexical-300	39.4	0.72	0.70	0.70	0.80	0.65	0.71	0.98	0.99	0.99

Table C-67*Evaluation Metrics of 1BiLSTM(75)-CRF (Adam) Per Class*

1BiLSTM(75)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	60.2	0.72	0.72	0.72	0.76	0.69	0.73	0.98	0.99	0.99
glove-twitter-100	44.0	0.72	0.71	0.71	0.79	0.66	0.72	0.98	0.99	0.99
glove-twitter-200	40.4	0.71	0.71	0.71	0.79	0.67	0.72	0.98	0.99	0.99
glove-wiki-gigaword-50	43.0	0.73	0.74	0.73	0.79	0.71	0.74	0.99	0.99	0.99
glove-wiki-gigaword-300	35.8	0.72	0.71	0.71	0.78	0.67	0.72	0.98	0.99	0.99
word2vec-google-news-300	32.4	0.72	0.71	0.71	0.80	0.66	0.72	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	31.2	0.72	0.71	0.71	0.78	0.67	0.72	0.98	0.99	0.99
Lexical-300	34.8	0.72	0.70	0.71	0.81	0.63	0.72	0.98	0.99	0.99

Table C-68*Evaluation Metrics of 1BiLSTM(100)-CRF (Adam) Per Class*

1BiLSTM(100)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	57.4	0.72	0.73	0.72	0.75	0.71	0.73	0.99	0.99	0.99
glove-twitter-100	35.4	0.71	0.73	0.72	0.77	0.69	0.72	0.99	0.98	0.99
glove-twitter-200	27.8	0.72	0.74	0.72	0.79	0.68	0.72	0.98	0.99	0.99
glove-wiki-gigaword-50	51.4	0.71	0.73	0.72	0.77	0.69	0.73	0.98	0.99	0.99
glove-wiki-gigaword-300	26.8	0.72	0.72	0.72	0.78	0.69	0.73	0.98	0.99	0.99
word2vec-google-news-300	28.4	0.72	0.71	0.72	0.80	0.66	0.72	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	28.6	0.72	0.71	0.71	0.79	0.67	0.72	0.98	0.99	0.99
Lexical-300	28.4	0.73	0.72	0.72	0.78	0.70	0.73	0.98	0.99	0.99

Table C-69*Evaluation Metrics of 1BiGRU(50)-CRF (rmsprop) With Fixed Word Embedding*

1BiGRU(50)-CRF (rmsprop) with Fixed Word Embeddings										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	146.4	0.64	0.57	0.60	0.67	0.67	0.66	0.98	0.98	0.98
glove-twitter-100	135.8	0.62	0.61	0.61	0.67	0.68	0.68	0.98	0.98	0.98
glove-twitter-200	147.8	0.61	0.58	0.59	0.70	0.64	0.67	0.98	0.98	0.98
glove-wiki-gigaword-50	144.8	0.63	0.62	0.63	0.68	0.69	0.68	0.98	0.98	0.98
glove-wiki-gigaword-300	129.2	0.59	0.60	0.60	0.67	0.65	0.65	0.98	0.98	0.98
word2vec-google-news-300	140.6	0.61	0.61	0.61	0.66	0.67	0.66	0.98	0.98	0.98
fasttext-wiki-news-subwords-300	114.8	0.63	0.63	0.63	0.66	0.67	0.66	0.98	0.98	0.98

Table C-70*Evaluation Metrics of 1BiGRU(50)-CRF (rmsprop) Per Class*

1BiGRU(50N)-CRF (rmsprop)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	102.6	0.67	0.74	0.70	0.74	0.70	0.72	0.99	0.98	0.99
glove-twitter-100	100.4	0.68	0.73	0.71	0.74	0.70	0.71	0.99	0.98	0.99
glove-twitter-200	87.2	0.67	0.73	0.70	0.75	0.67	0.70	0.99	0.98	0.99
glove-wiki-gigaword-50	113.8	0.68	0.74	0.71	0.75	0.68	0.71	0.98	0.98	0.99
glove-wiki-gigaword-300	94.0	0.68	0.72	0.70	0.76	0.68	0.71	0.99	0.99	0.99
word2vec-google-news-300	87.0	0.67	0.71	0.69	0.73	0.68	0.70	0.98	0.98	0.98
fasttext-wiki-news-subwords-300	89.8	0.65	0.72	0.68	0.70	0.66	0.68	0.98	0.98	0.98
Lexical-300	91.4	0.64	0.71	0.68	0.67	0.70	0.68	0.99	0.98	0.98

Table C-71*Evaluation Metrics of 1BiGRU(25)-CRF (Adam) Per Class*

1BiGRU(25N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	159.2	0.72	0.71	0.71	0.78	0.66	0.71	0.98	0.99	0.99
glove-twitter-100	117.0	0.70	0.68	0.70	0.78	0.64	0.70	0.98	0.99	0.99
glove-twitter-200	121.8	0.70	0.68	0.69	0.79	0.64	0.70	0.98	0.99	0.99
glove-wiki-gigaword-50	142.8	0.69	0.68	0.69	0.78	0.63	0.69	0.98	0.99	0.99
glove-wiki-gigaword-300	102.0	0.72	0.67	0.69	0.80	0.62	0.70	0.98	0.99	0.99
word2vec-google-news-300	95.8	0.71	0.68	0.71	0.79	0.63	0.70	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	88.6	0.71	0.67	0.68	0.79	0.62	0.69	0.98	0.99	0.99
Lexical-300	85.6	0.70	0.68	0.69	0.79	0.64	0.70	0.98	0.99	0.99

Table C-72*Evaluation Metrics of 1BiGRU(50)-CRF (Adam) Per Class*

1BiGRU(50N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	63.0	0.71	0.73	0.72	0.75	0.71	0.73	0.99	0.99	0.98
glove-twitter-100	49.2	0.73	0.71	0.72	0.78	0.68	0.73	0.98	0.99	0.99
glove-twitter-200	48.4	0.72	0.70	0.71	0.78	0.66	0.72	0.98	0.99	0.99
glove-wiki-gigaword-50	63.2	0.72	0.72	0.72	0.77	0.70	0.73	0.99	0.99	0.99
glove-wiki-gigaword-300	42.4	0.72	0.70	0.71	0.79	0.66	0.72	0.98	0.99	0.99
word2vec-google-news-300	47.8	0.71	0.70	0.70	0.79	0.65	0.71	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	49.8	0.72	0.68	0.70	0.80	0.63	0.70	0.98	0.99	0.99
Lexical-300	50.4	0.71	0.70	0.71	0.79	0.65	0.71	0.98	0.99	0.99

Table C-73*Evaluation Metrics of 1BiGRU(75)-CRF (Adam) Per Class*

BiGRU(75N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	53.6	0.71	0.74	0.73	0.76	0.71	0.73	0.99	0.98	0.99
glove-twitter-100	36.2	0.72	0.73	0.72	0.78	0.69	0.73	0.98	0.99	0.99
glove-twitter-200	37.6	0.73	0.71	0.72	0.77	0.69	0.73	0.98	0.99	0.99
glove-wiki-gigaword-50	41.0	0.74	0.71	0.73	0.79	0.70	0.74	0.98	0.99	0.99
glove-wiki-gigaword-300	30.6	0.71	0.71	0.71	0.79	0.67	0.72	0.98	0.99	0.99
word2vec-google-news-300	32.8	0.72	0.71	0.71	0.79	0.67	0.72	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	37.4	0.72	0.70	0.71	0.79	0.66	0.72	0.98	0.99	0.99
Lexical-300	40.6	0.72	0.68	0.70	0.79	0.65	0.69	0.98	0.99	0.99

Table C-74*Evaluation Metrics of 1BiGRU(100)-CRF (Adam) Per Class*

1BiGRU(100N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	46.0	0.71	0.73	0.72	0.75	0.71	0.73	0.99	0.99	0.99
glove-twitter-100	34.4	0.73	0.71	0.72	0.77	0.69	0.72	0.98	0.99	0.99
glove-twitter-200	30.6	0.72	0.73	0.72	0.78	0.68	0.72	0.98	0.99	0.99
glove-wiki-gigaword-50	35.4	0.73	0.73	0.73	0.76	0.72	0.73	0.99	0.99	0.99
glove-wiki-gigaword-300	29.0	0.72	0.71	0.71	0.79	0.66	0.71	0.98	0.99	0.99
word2vec-google-news-300	29.4	0.72	0.71	0.72	0.78	0.68	0.72	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	27.4	0.72	0.71	0.72	0.76	0.69	0.72	0.99	0.99	0.99
Lexical-300	33.8	0.72	0.70	0.71	0.77	0.67	0.71	0.98	0.99	0.99

Table C-75*Evaluation Metrics of 2BiLSTM(50)-CRF (rmsprop) Per Class*

2BiLSTM(50N)-CRF (rmsprop)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	74.0	0.67	0.74	0.71	0.76	0.70	0.72	0.99	0.99	0.99
glove-twitter-100	90.4	0.66	0.73	0.69	0.77	0.68	0.72	0.99	0.99	0.98
glove-twitter-200	56.6	0.65	0.77	0.70	0.75	0.71	0.72	0.99	0.98	0.98
glove-wiki-gigaword-50	74.6	0.70	0.70	0.70	0.80	0.65	0.72	0.98	0.99	0.99
glove-wiki-gigaword-300	54.0	0.68	0.75	0.71	0.78	0.69	0.73	0.99	0.99	0.99
word2vec-google-news-300	54.4	0.65	0.74	0.69	0.73	0.71	0.72	0.99	0.98	0.99
fasttext-wiki-news-subwords-300	54.0	0.65	0.73	0.69	0.72	0.70	0.71	0.99	0.98	0.98
Lexical-300	47.8	0.62	0.74	0.67	0.69	0.72	0.70	0.99	0.98	0.98

Table C-76*Evaluation Metrics of 2BiLSTM(25)-CRF (Adam) Per Class*

2BiLSTM(25N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	113.4	0.68	0.68	0.68	0.74	0.67	0.70	0.98	0.99	0.99
glove-twitter-100	71.8	0.70	0.68	0.69	0.76	0.65	0.70	0.98	0.99	0.99
glove-twitter-200	68.2	0.69	0.69	0.69	0.78	0.65	0.70	0.98	0.99	0.99
glove-wiki-gigaword-50	85.8	0.68	0.68	0.68	0.77	0.63	0.69	0.98	0.99	0.98
glove-wiki-gigaword-300	58.2	0.69	0.68	0.69	0.80	0.62	0.69	0.98	0.99	0.99
word2vec-google-news-300	58.8	0.69	0.69	0.69	0.81	0.63	0.70	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	58.6	0.70	0.67	0.69	0.80	0.62	0.69	0.98	0.99	0.99
Lexical-300	57.0	0.70	0.68	0.69	0.83	0.62	0.70	0.98	0.99	0.99

Table C-77*Evaluation Metrics of 2BiLSTM(50)-CRF (Adam) Per Class*

2BiLSTM(50N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	90.6	0.70	0.70	0.70	0.78	0.68	0.72	0.98	0.99	0.99
glove-twitter-100	62.0	0.71	0.70	0.71	0.81	0.65	0.72	0.98	0.99	0.99
glove-twitter-200	51.0	0.71	0.70	0.70	0.79	0.66	0.72	0.98	0.99	0.99
glove-wiki-gigaword-50	39.6	0.72	0.71	0.71	0.78	0.68	0.73	0.98	0.99	0.99
glove-wiki-gigaword-300	27.2	0.71	0.71	0.71	0.78	0.67	0.72	0.98	0.99	0.99
word2vec-google-news-300	54.2	0.70	0.71	0.70	0.79	0.66	0.72	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	56.2	0.70	0.70	0.70	0.79	0.65	0.71	0.98	0.99	0.99
Lexical-300	55.2	0.69	0.72	0.70	0.78	0.67	0.72	0.99	0.99	0.99

Table C-78*Evaluation Metrics of 2BiLSTM(75)-CRF (Adam) Per Class*

2BiLSTM(75N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	84.2	0.69	0.71	0.70	0.75	0.68	0.71	0.98	0.99	0.99
glove-twitter-100	23.8	0.70	0.75	0.72	0.75	0.73	0.74	0.99	0.98	0.99
glove-twitter-200	22.6	0.72	0.73	0.72	0.79	0.69	0.73	0.98	0.99	0.99
glove-wiki-gigaword-50	78.2	0.69	0.71	0.70	0.75	0.70	0.72	0.99	0.99	0.99
glove-wiki-gigaword-300	21.8	0.72	0.72	0.72	0.81	0.68	0.74	0.98	0.99	0.99
word2vec-google-news-300	24.6	0.72	0.72	0.72	0.79	0.68	0.73	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	25.8	0.71	0.71	0.71	0.77	0.68	0.72	0.99	0.99	0.99
Lexical-300	25.6	0.69	0.73	0.71	0.76	0.69	0.72	0.99	0.99	0.99

Table C-79*Evaluation Metrics of 2BiLSTM(100)-CRF (Adam) Per Class*

2BiLSTM(100N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	83.8	0.69	0.72	0.70	0.75	0.70	0.72	0.99	0.99	0.99
glove-twitter-100	98.8	0.69	0.71	0.70	0.77	0.68	0.71	0.98	0.99	0.99
glove-twitter-200	73.6	0.70	0.69	0.70	0.79	0.66	0.71	0.98	0.99	0.99
glove-wiki-gigaword-50	78.6	0.69	0.70	0.70	0.76	0.67	0.72	0.98	0.99	0.99
glove-wiki-gigaword-300	74.8	0.71	0.71	0.71	0.79	0.67	0.72	0.98	0.99	0.99
word2vec-google-news-300	50.0	0.71	0.70	0.70	0.77	0.68	0.71	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	53.2	0.70	0.69	0.69	0.76	0.68	0.71	0.98	0.99	0.99
Lexical-300	104.4	0.69	0.70	0.69	0.79	0.65	0.71	0.98	0.99	0.99

Table C-80*Evaluation Metrics of 2BiGRU(50)-CRF (rmsprop) Per Class*

2BiGRU(50N)-CRF (rmsprop)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	72.4	0.71	0.71	0.70	0.80	0.67	0.72	0.98	0.99	0.99
glove-twitter-100	59.4	0.67	0.74	0.70	0.76	0.69	0.72	0.99	0.99	0.99
glove-twitter-200	58.4	0.68	0.74	0.71	0.77	0.57	0.72	0.99	0.99	0.99
glove-wiki-gigaword-50	64.2	0.69	0.74	0.71	0.76	0.70	0.72	0.99	0.98	0.99
glove-wiki-gigaword-300	50.8	0.68	0.72	0.70	0.75	0.68	0.71	0.98	0.99	0.99
word2vec-google-news-300	58.8	0.67	0.72	0.69	0.75	0.67	0.70	0.99	0.98	0.98
fasttext-wiki-news-subwords-300	57.2	0.65	0.72	0.68	0.73	0.68	0.70	0.98	0.98	0.98
Lexical-300	54.8	0.66	0.70	0.68	0.73	0.67	0.70	0.98	0.98	0.98

Table C-81*Evaluation Metrics of 2BiGRU(25)-CRF (Adam) Per Class*

2BiGRU(25N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	95.0	0.69	0.71	0.70	0.76	0.68	0.71	0.98	0.99	0.99
glove-twitter-100	60.2	0.71	0.71	0.71	0.78	0.65	0.71	0.98	0.99	0.99
glove-twitter-200	59.4	0.70	0.68	0.69	0.78	0.65	0.71	0.98	0.99	0.99
glove-wiki-gigaword-50	64.2	0.70	0.71	0.71	0.79	0.67	0.72	0.98	0.99	0.99
glove-wiki-gigaword-300	54.6	0.70	0.68	0.69	0.78	0.64	0.70	0.98	0.99	0.99
word2vec-google-news-300	51.2	0.71	0.67	0.69	0.78	0.61	0.69	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	47.6	0.70	0.68	0.69	0.79	0.61	0.69	0.98	0.99	0.99
Lexical-300	49.6	0.70	0.68	0.69	0.78	0.63	0.70	0.98	0.99	0.99

Table C-82*Evaluation Metrics of 2BiGRU(50)-CRF (Adam) Per Class*

2BiGRU(50N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	52.8	0.71	0.72	0.71	0.76	0.70	0.73	0.99	0.99	0.99
glove-twitter-100	27.2	0.71	0.72	0.72	0.77	0.71	0.73	0.99	0.99	0.99
glove-twitter-200	27.2	0.72	0.71	0.71	0.78	0.68	0.72	0.99	0.99	0.99
glove-wiki-gigaword-50	31.4	0.71	0.71	0.71	0.77	0.72	0.74	0.99	0.99	0.99
glove-wiki-gigaword-300	27.2	0.71	0.71	0.71	0.77	0.69	0.73	0.99	0.99	0.99
word2vec-google-news-300	29.0	0.71	0.71	0.70	0.80	0.67	0.72	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	27.6	0.71	0.69	0.70	0.78	0.66	0.71	0.98	0.99	0.99
Lexical-300	29.8	0.70	0.69	0.70	0.79	0.66	0.71	0.98	0.99	0.99

Table C-83*Evaluation Metrics of 2BiGRU(75N)-CRF (Adam) Per Class*

2BiGRU(75N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	38.2	0.72	0.71	0.71	0.75	0.70	0.72	0.99	0.99	0.99
glove-twitter-100	25.4	0.71	0.72	0.72	0.79	0.70	0.74	0.99	0.99	0.99
glove-twitter-200	22.2	0.71	0.71	0.71	0.78	0.70	0.73	0.99	0.99	0.99
glove-wiki-gigaword-50	29.0	0.73	0.72	0.72	0.79	0.71	0.75	0.99	0.99	0.99
glove-wiki-gigaword-300	22.6	0.72	0.71	0.71	0.79	0.69	0.73	0.98	0.99	0.99
word2vec-google-news-300	21.6	0.71	0.71	0.71	0.79	0.67	0.72	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	20.6	0.71	0.71	0.71	0.77	0.68	0.73	0.99	0.99	0.99
Lexical-300	20.4	0.70	0.71	0.70	0.78	0.66	0.71	0.98	0.99	0.99

Table C-84*Evaluation Metrics of 2BiGRU(100)-CRF (Adam) Per Class*

2BiGRU(100N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	32.0	0.72	0.73	0.72	0.76	0.71	0.73	0.99	0.99	0.99
glove-twitter-100	19.6	0.73	0.71	0.72	0.79	0.70	0.72	0.98	0.99	0.99
glove-twitter-200	22.2	0.71	0.70	0.70	0.80	0.67	0.73	0.98	0.99	0.99
glove-wiki-gigaword-50	25.6	0.71	0.73	0.72	0.78	0.68	0.72	0.98	0.99	0.99
glove-wiki-gigaword-300	19.4	0.72	0.72	0.72	0.81	0.69	0.74	0.98	0.99	0.99
word2vec-google-news-300	21.2	0.72	0.71	0.71	0.77	0.68	0.72	0.99	0.99	0.99
fasttext-wiki-news-subwords-300	19.2	0.71	0.71	0.71	0.79	0.66	0.72	0.98	0.99	0.99
Lexical-300	17.4	0.70	0.69	0.69	0.79	0.65	0.71	0.98	0.99	0.99

C3.2 Further Exploration of Results of Deep Learning Models Using Recurrent Layers

Table C-85

Evaluation Metrics of 1BiLSTM(50)-1BiGRU(50)-CRF (rmsprop) Per Class

1BiLSTM(50)-1BiGRU(50)-CRF (rmsprop)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	116.6	0.69	0.73	0.71	0.78	0.65	0.71	0.99	0.99	0.99
glove-twitter-100	76.2	0.70	0.73	0.71	0.77	0.68	0.72	0.98	0.99	0.99
glove-twitter-200	77.2	0.69	0.73	0.71	0.77	0.69	0.73	0.99	0.99	0.99
glove-wiki-gigaword-50	96.2	0.68	0.74	0.71	0.76	0.67	0.71	0.98	0.98	0.99
glove-wiki-gigaword-300	70.8	0.69	0.73	0.71	0.76	0.68	0.71	0.98	0.99	0.99
word2vec-google-news-300	70.8	0.69	0.73	0.71	0.76	0.69	0.74	0.99	0.99	0.99
fasttext-wiki-news-subwords-300	72.4	0.65	0.73	0.69	0.72	0.70	0.71	0.99	0.98	0.98
Lexical-300	79.0	0.66	0.72	0.69	0.73	0.70	0.71	0.99	0.98	0.98

Table C-86*Evaluation Metrics of 1BiLSTM(25)-1BiGRU(25)-CRF (Adam) Per Class*

1BiLSTM(25N)-1BiGRU(25N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	80.6	0.73	0.71	0.72	0.76	0.70	0.73	0.98	0.99	0.99
glove-twitter-100	54.6	0.72	0.70	0.71	0.79	0.67	0.72	0.98	0.99	0.99
glove-twitter-200	50.2	0.74	0.70	0.72	0.80	0.67	0.73	0.98	0.99	0.99
glove-wiki-gigaword-50	54.8	0.74	0.73	0.73	0.81	0.68	0.74	0.98	0.99	0.99
glove-wiki-gigaword-300	43.6	0.73	0.70	0.71	0.81	0.67	0.73	0.98	0.99	0.99
word2vec-google-news-300	42.8	0.73	0.69	0.71	0.80	0.65	0.72	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	43.6	0.73	0.70	0.71	0.81	0.67	0.73	0.98	0.99	0.99
Lexical-300	46.4	0.73	0.72	0.72	0.80	0.67	0.72	0.98	0.99	0.99

Table C-87*Evaluation Metrics of 1BiLSTM(50)-1BiGRU(50)-CRF (Adam) Per Class*

1BiLSTM(50N)-1BiGRU(50N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	48.6	0.74	0.73	0.73	0.77	0.72	0.74	0.99	0.99	0.99
glove-twitter-100	29.8	0.73	0.71	0.72	0.78	0.68	0.72	0.99	0.99	0.99
glove-twitter-200	29.2	0.73	0.71	0.72	0.79	0.68	0.73	0.98	0.99	0.99
glove-wiki-gigaword-50	35.0	0.74	0.72	0.73	0.78	0.70	0.74	0.98	0.99	0.99
glove-wiki-gigaword-300	25.8	0.73	0.72	0.73	0.80	0.68	0.73	0.98	0.99	0.99
word2vec-google-news-300	29.2	0.72	0.72	0.72	0.81	0.66	0.72	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	29.0	0.72	0.70	0.71	0.78	0.66	0.71	0.98	0.99	0.99
Lexical-300	31.6	0.72	0.72	0.72	0.77	0.68	0.72	0.98	0.99	0.99

Table C-88*Evaluation Metrics of 1BiLSTM(75)-1BiGRU(75)-CRF (Adam) Per Class*

1BiLSTM(75N)-1BiGRU(75N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	33.6	0.73	0.73	0.73	0.76	0.72	0.73	0.99	0.99	0.99
glove-twitter-100	30.8	0.73	0.75	0.74	0.79	0.71	0.74	0.98	0.99	0.99
glove-twitter-200	22.8	0.74	0.72	0.73	0.79	0.71	0.73	0.98	0.99	0.99
glove-wiki-gigaword-50	37.2	0.73	0.72	0.72	0.76	0.70	0.72	0.98	0.99	0.99
glove-wiki-gigaword-300	23.8	0.74	0.72	0.72	0.79	0.67	0.72	0.98	0.99	0.99
word2vec-google-news-300	26.2	0.73	0.73	0.73	0.78	0.69	0.73	0.99	0.99	0.99
fasttext-wiki-news-subwords-300	27.8	0.73	0.71	0.72	0.78	0.68	0.72	0.98	0.99	0.99
Lexical-300	24.0	0.73	0.71	0.72	0.78	0.68	0.72	0.99	0.99	0.99

Table C-89*Evaluation Metrics of 1BiLSTM(100)-1BiGRU(100)-CRF (Adam) Per Class*

1BiLSTM(100N)-1BiGRU(100N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	34.0	0.72	0.75	0.73	0.76	0.73	0.74	0.99	0.98	0.99
glove-twitter-100	22.6	0.74	0.72	0.73	0.80	0.69	0.73	0.99	0.99	0.99
glove-twitter-200	24.6	0.73	0.73	0.73	0.79	0.70	0.74	0.99	0.99	0.99
glove-wiki-gigaword-50	31.0	0.74	0.73	0.74	0.78	0.71	0.74	0.99	0.99	0.99
glove-wiki-gigaword-300	21.6	0.72	0.72	0.72	0.79	0.68	0.73	0.98	0.99	0.99
word2vec-google-news-300	23.4	0.73	0.72	0.72	0.77	0.69	0.72	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	22.8	0.73	0.71	0.72	0.79	0.67	0.72	0.98	0.99	0.99
Lexical-300	23.4	0.73	0.71	0.72	0.78	0.68	0.73	0.98	0.99	0.99

Table C-90*Evaluation Metrics of 2BiLSTM(50)-2BiGRU(50)-CRF (rmsprop) Per Class*

2BiLSTM(50N)-2BiGRU(50N)-CRF (rmsprop)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	76.2	0.68	0.74	0.71	0.76	0.71	0.73	0.99	0.98	0.99
glove-twitter-100	58.4	0.69	0.73	0.71	0.79	0.67	0.72	0.98	0.99	0.99
glove-twitter-200	49.2	0.71	0.72	0.71	0.80	0.69	0.74	0.98	0.99	0.99
glove-wiki-gigaword-50	68.6	0.68	0.73	0.71	0.78	0.71	0.74	0.99	0.99	0.99
glove-wiki-gigaword-300	50.6	0.69	0.72	0.70	0.79	0.69	0.73	0.99	0.99	0.99
word2vec-google-news-300	55.6	0.69	0.72	0.70	0.78	0.65	0.71	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	48.4	0.67	0.72	0.69	0.74	0.67	0.70	0.99	0.99	0.99
Lexical-300	46.0	0.68	0.72	0.69	0.75	0.68	0.71	0.98	0.98	0.99

Table C-91*Evaluation Metrics of 2BiLSTM(25)-2BiGRU(25)-CRF (Adam) Per Class*

2BiLSTM(25N)-2BiGRU(25N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	54.4	0.70	0.70	0.70	0.76	0.68	0.72	0.98	0.99	0.99
glove-twitter-100	34.8	0.72	0.72	0.72	0.81	0.69	0.74	0.98	0.99	0.99
glove-twitter-200	46.4	0.70	0.70	0.70	0.78	0.67	0.72	0.98	0.99	0.99
glove-wiki-gigaword-50	38.8	0.71	0.72	0.72	0.79	0.70	0.74	0.98	0.99	0.99
glove-wiki-gigaword-300	33.6	0.72	0.71	0.71	0.79	0.67	0.72	0.98	0.99	0.99
word2vec-google-news-300	35.0	0.71	0.69	0.70	0.78	0.66	0.72	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	31.2	0.72	0.70	0.71	0.79	0.65	0.71	0.98	0.99	0.99
Lexical-300	32.8	0.72	0.70	0.71	0.79	0.66	0.71	0.98	0.99	0.99

Table C-92*Evaluation Metrics of 2BiLSTM(50)-2BiGRU(50)-CRF (Adam) Per Class*

2BiLSTM(50N)-2BiGRU(50N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	32.8	0.72	0.74	0.73	0.78	0.72	0.75	0.99	0.99	0.99
glove-twitter-100	24.4	0.72	0.72	0.72	0.78	0.70	0.74	0.98	0.99	0.99
glove-twitter-200	22.6	0.72	0.72	0.72	0.77	0.70	0.73	0.99	0.99	0.99
glove-wiki-gigaword-50	28.8	0.73	0.71	0.72	0.80	0.67	0.73	0.98	0.99	0.99
glove-wiki-gigaword-300	23.2	0.72	0.73	0.72	0.79	0.68	0.73	0.99	0.99	0.99
word2vec-google-news-300	21.4	0.71	0.72	0.71	0.79	0.68	0.73	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	22.6	0.70	0.70	0.70	0.77	0.67	0.71	0.98	0.99	0.99
Lexical-300	21.8	0.71	0.71	0.71	0.78	0.68	0.72	0.99	0.99	0.99

Table C-93*Evaluation Metrics of 2BiLSTM(75)-2BiGRU(75)-CRF (Adam) Per Class*

2BiLSTM(75N)-2BiGRU(75N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	34.8	0.71	0.72	0.71	0.77	0.71	0.74	0.98	0.99	0.99
glove-twitter-100	21.8	0.71	0.72	0.71	0.79	0.69	0.73	0.98	0.99	0.99
glove-twitter-200	16.2	0.72	0.73	0.72	0.78	0.71	0.73	0.98	0.99	0.99
glove-wiki-gigaword-50	22.8	0.73	0.73	0.73	0.79	0.71	0.74	0.99	0.99	0.99
glove-wiki-gigaword-300	18.0	0.72	0.73	0.72	0.79	0.69	0.73	0.98	0.99	0.99
word2vec-google-news-300	15.0	0.72	0.72	0.72	0.79	0.70	0.74	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	17.2	0.71	0.69	0.70	0.78	0.66	0.71	0.98	0.99	0.99
Lexical-300	16.8	0.70	0.70	0.70	0.79	0.67	0.71	0.98	0.99	0.99

Table C-94*Evaluation Metrics of 2BiLSTM(100)-2BiGRU(100)-CRF (Adam) Per Class*

2BiLSTM(100N)-2BiGRU(100N)-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	27.4	0.71	0.72	0.72	0.77	0.70	0.73	0.99	0.99	0.99
glove-twitter-100	24.4	0.72	0.71	0.71	0.78	0.69	0.73	0.98	0.99	0.99
glove-twitter-200	16.8	0.72	0.72	0.72	0.79	0.68	0.72	0.99	0.99	0.99
glove-wiki-gigaword-50	22.6	0.74	0.71	0.72	0.79	0.70	0.74	0.99	0.99	0.99
glove-wiki-gigaword-300	16.6	0.72	0.73	0.73	0.78	0.71	0.74	0.99	0.99	0.99
word2vec-google-news-300	17.0	0.71	0.72	0.72	0.78	0.69	0.72	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	15.4	0.71	0.72	0.71	0.78	0.68	0.73	0.98	0.99	0.99
Lexical-300	16.0	0.70	0.71	0.70	0.79	0.66	0.71	0.98	0.99	0.99

C3.3 Further Exploration of Results of Deep Learning Models Using Recurrent and Convolutional Neural Network Layers

Table C-95

Evaluation Metrics of 1BiLSTM(25N)-1CNN-CRF (Adam) Per Class

1BiLSTM(25)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	41.4	0.71	0.71	0.71	0.80	0.68	0.73	0.99	0.99	0.99
glove-twitter-100	25.2	0.73	0.71	0.72	0.79	0.68	0.72	0.98	0.99	0.99
glove-twitter-200	20.6	0.73	0.72	0.72	0.81	0.67	0.73	0.99	0.99	0.99
glove-wiki-gigaword-50	34.0	0.72	0.71	0.72	0.79	0.68	0.73	0.99	0.99	0.99
glove-wiki-gigaword-300	19.8	0.73	0.71	0.72	0.81	0.66	0.72	0.99	0.99	0.99
word2vec-google-news-300	19.6	0.73	0.71	0.72	0.79	0.64	0.71	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	19.8	0.71	0.71	0.71	0.79	0.65	0.71	0.98	0.99	0.99
Lexical-300	22.0	0.71	0.71	0.71	0.78	0.66	0.71	0.98	0.99	0.99

Table C-96*Evaluation Metrics of 1BiLSTM(50)-1CNN-CRF (Adam) Per Class*

1BiLSTM(50N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	44.0	0.73	0.71	0.71	0.80	0.69	0.73	0.98	0.99	0.99
glove-twitter-100	23.8	0.73	0.72	0.72	0.80	0.68	0.73	0.98	0.99	0.99
glove-twitter-200	17.2	0.74	0.72	0.73	0.80	0.68	0.73	0.98	0.99	0.99
glove-wiki-gigaword-50	26.6	0.71	0.73	0.72	0.79	0.68	0.73	0.97	0.99	0.99
glove-wiki-gigaword-300	17.8	0.72	0.73	0.73	0.79	0.68	0.73	0.98	0.99	0.99
word2vec-google-news-300	17.8	0.73	0.72	0.73	0.81	0.66	0.73	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	21.2	0.71	0.73	0.72	0.79	0.65	0.71	0.99	0.99	0.99
Lexical-300	22.2	0.72	0.69	0.70	0.80	0.65	0.71	0.98	0.99	0.99

Table C-97*Evaluation Metrics of 1BiLSTM(75)-1CNN-CRF (Adam) Per Class*

1BiLSTM(75N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	41.4	0.73	0.72	0.72	0.77	0.71	0.73	0.98	0.99	0.99
glove-twitter-100	22.6	0.73	0.70	0.71	0.80	0.65	0.72	0.98	0.99	0.99
glove-twitter-200	19.0	0.71	0.74	0.73	0.80	0.67	0.73	0.99	0.99	0.99
glove-wiki-gigaword-50	30.0	0.73	0.71	0.72	0.80	0.66	0.72	0.98	0.99	0.99
glove-wiki-gigaword-300	15.8	0.74	0.73	0.73	0.80	0.68	0.73	0.98	0.99	0.99
word2vec-google-news-300	17.8	0.73	0.72	0.72	0.80	0.65	0.71	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	22.2	0.72	0.71	0.72	0.78	0.67	0.72	0.98	0.99	0.99
Lexical-300	23.6	0.72	0.71	0.72	0.77	0.67	0.71	0.98	0.99	0.99

Table C-98*Evaluation Metrics of 1BiLSTM(100)-1CNN-CRF (Adam) Per Class*

1BiLSTM(100N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	41.4	0.73	0.70	0.72	0.80	0.67	0.73	0.98	0.99	0.99
glove-twitter-100	23.4	0.73	0.73	0.72	0.80	0.68	0.73	0.98	0.99	0.99
glove-twitter-200	19.4	0.74	0.72	0.73	0.82	0.67	0.74	0.99	0.99	0.99
glove-wiki-gigaword-50	29.8	0.74	0.72	0.73	0.79	0.70	0.74	0.99	0.99	0.99
glove-wiki-gigaword-300	16.8	0.73	0.72	0.73	0.82	0.65	0.72	0.98	0.99	0.99
word2vec-google-news-300	18.4	0.73	0.72	0.73	0.80	0.70	0.74	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	24.2	0.72	0.71	0.71	0.79	0.66	0.72	0.98	0.99	0.99
Lexical-300	24.4	0.72	0.72	0.72	0.78	0.67	0.72	0.99	0.99	0.99

Table C-99*Evaluation Metrics of 1BiGRU(25)-1CNN-CRF (Adam) Per Class*

1BiGRU(25N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	49.0	0.72	0.71	0.72	0.79	0.68	0.73	0.99	0.99	0.99
glove-twitter-100	29.4	0.73	0.70	0.72	0.78	0.69	0.73	0.98	0.99	0.99
glove-twitter-200	23.8	0.74	0.71	0.73	0.80	0.68	0.73	0.99	0.99	0.99
glove-wiki-gigaword-50	35.0	0.74	0.73	0.73	0.81	0.68	0.74	0.98	0.99	0.99
glove-wiki-gigaword-300	21.2	0.74	0.72	0.73	0.80	0.69	0.74	0.98	0.99	0.99
word2vec-google-news-300	21.4	0.74	0.72	0.73	0.80	0.69	0.74	0.99	0.99	0.99
fasttext-wiki-news-subwords-300	21.0	0.72	0.71	0.72	0.79	0.67	0.72	0.98	0.99	0.99
Lexical-300	20.6	0.72	0.70	0.71	0.78	0.67	0.71	0.98	0.99	0.99

Table C-100*Evaluation Metrics of 1BiGRU(50)-1CNN-CRF (Adam) Per Class*

1BiGRU(50N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	37.0	0.71	0.74	0.72	0.76	0.71	0.73	0.99	0.99	0.99
glove-twitter-100	22.8	0.72	0.72	0.72	0.78	0.68	0.72	0.98	0.99	0.99
glove-twitter-200	19.2	0.73	0.72	0.73	0.79	0.69	0.73	0.98	0.99	0.99
glove-wiki-gigaword-50	29.2	0.73	0.71	0.72	0.80	0.69	0.74	0.98	0.99	0.99
glove-wiki-gigaword-300	15.8	0.74	0.72	0.73	0.79	0.69	0.73	0.98	0.99	0.99
word2vec-google-news-300	17.8	0.73	0.71	0.72	0.78	0.68	0.72	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	21.4	0.72	0.71	0.71	0.78	0.67	0.72	0.98	0.99	0.99
Lexical-300	20.4	0.73	0.71	0.71	0.79	0.65	0.71	0.98	0.99	0.99

Table C-101*Evaluation Metrics of 1BiGRU(75)-1CNN-CRF (Adam) Per Class*

1BiGRU(75N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	38.2	0.73	0.72	0.73	0.78	0.71	0.75	0.99	0.99	0.99
glove-twitter-100	24.4	0.74	0.72	0.73	0.81	0.70	0.74	0.98	0.99	0.99
glove-twitter-200	22.2	0.73	0.71	0.72	0.81	0.70	0.75	0.98	0.99	0.99
glove-wiki-gigaword-50	28.6	0.74	0.73	0.74	0.80	0.69	0.74	0.99	0.99	0.99
glove-wiki-gigaword-300	17.0	0.75	0.71	0.72	0.80	0.68	0.73	0.98	0.99	0.99
word2vec-google-news-300	17.6	0.74	0.71	0.72	0.81	0.65	0.72	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	20.0	0.72	0.72	0.72	0.79	0.67	0.72	0.98	0.99	0.99
Lexical-300	17.8	0.72	0.71	0.72	0.79	0.66	0.71	0.98	0.99	0.99

Table C-102*Evaluation Metrics of 1BiGRU(100)-1CNN-CRF (Adam) Per Class*

1BiGRU(100N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	32.8	0.73	0.73	0.73	0.77	0.73	0.75	0.99	0.99	0.99
glove-twitter-100	22.8	0.74	0.72	0.73	0.80	0.69	0.74	0.98	0.99	0.99
glove-twitter-200	20.2	0.74	0.73	0.73	0.79	0.68	0.73	0.99	0.99	0.99
glove-wiki-gigaword-50	27.4	0.74	0.71	0.73	0.79	0.70	0.74	0.98	0.99	0.99
glove-wiki-gigaword-300	17.0	0.74	0.74	0.74	0.80	0.69	0.74	0.98	0.99	0.99
word2vec-google-news-300	19.4	0.73	0.72	0.72	0.79	0.68	0.73	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	21.2	0.72	0.71	0.71	0.79	0.65	0.71	0.98	0.99	0.99
Lexical-300	17.4	0.71	0.71	0.71	0.79	0.66	0.71	0.98	0.99	0.99

Table C-103*Evaluation Metrics of 1BiLSTM(25)-1BiGRU(25)-1CNN-CRF (Adam) Per Class*

1BiLSTM(25N)-1BiGRU(25N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	36.6	0.74	0.73	0.73	0.79	0.70	0.74	0.99	0.99	0.99
glove-twitter-100	22.4	0.74	0.71	0.73	0.81	0.67	0.73	0.98	0.99	0.99
glove-twitter-200	20.4	0.73	0.72	0.73	0.81	0.67	0.73	0.98	0.99	0.99
glove-wiki-gigaword-50	32.2	0.73	0.71	0.72	0.80	0.67	0.73	0.98	0.99	0.99
glove-wiki-gigaword-300	18.4	0.73	0.72	0.73	0.78	0.70	0.73	0.98	0.99	0.99
word2vec-google-news-300	20.2	0.74	0.72	0.73	0.80	0.68	0.73	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	19.4	0.72	0.70	0.71	0.80	0.66	0.72	0.98	0.99	0.99
Lexical-300	21.2	0.71	0.70	0.71	0.78	0.65	0.71	0.98	0.99	0.99

Table C-104*Evaluation Metrics of 1BiLSTM(50)-1BiGRU(50)-1CNN-CRF (Adam) Per Class*

1BiLSTM(50N)-1BiGRU(50N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	33.6	0.73	0.72	0.73	0.78	0.72	0.74	0.99	0.99	0.99
glove-twitter-100	22.0	0.75	0.72	0.73	0.80	0.70	0.74	0.98	0.99	0.99
glove-twitter-200	18.2	0.73	0.72	0.72	0.79	0.68	0.72	0.99	0.99	0.99
glove-wiki-gigaword-50	27.2	0.75	0.72	0.73	0.80	0.70	0.74	0.98	0.99	0.99
glove-wiki-gigaword-300	18.0	0.73	0.71	0.72	0.82	0.67	0.73	0.98	0.99	0.99
word2vec-google-news-300	20.2	0.72	0.71	0.71	0.80	0.66	0.72	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	33.0	0.72	0.70	0.71	0.77	0.68	0.72	0.98	0.99	0.99
Lexical-300	31.8	0.72	0.70	0.71	0.79	0.66	0.71	0.98	0.99	0.99

Table C-105*Evaluation Metrics of 1BiLSTM(75)-1BiGRU(75)-1CNN-CRF (Adam) Per Class*

1BiLSTM(75N)-1BiGRU(75N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	35.6	0.73	0.71	0.72	0.80	0.71	0.75	0.99	0.99	0.99
glove-twitter-100	22.8	0.73	0.73	0.73	0.79	0.70	0.74	0.99	0.99	0.99
glove-twitter-200	20.6	0.74	0.72	0.73	0.80	0.69	0.73	0.99	0.99	0.99
glove-wiki-gigaword-50	29.2	0.72	0.73	0.73	0.79	0.69	0.73	0.99	0.99	0.99
glove-wiki-gigaword-300	18.4	0.73	0.72	0.73	0.80	0.68	0.73	0.99	0.99	0.99
word2vec-google-news-300	18.8	0.73	0.72	0.72	0.79	0.68	0.72	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	18.4	0.73	0.70	0.71	0.78	0.66	0.71	0.98	0.99	0.99
Lexical-300	19.8	0.72	0.70	0.71	0.80	0.64	0.71	0.98	0.99	0.99

Table C-106*Evaluation Metrics of 1BiLSTM(100)-1BiGRU(100)-1CNN-CRF (Adam) Per Class*

1BiLSTM(100N)-1BiGRU(100N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	35.0	0.74	0.71	0.72	0.80	0.69	0.74	0.98	0.99	0.99
glove-twitter-100	18.8	0.74	0.73	0.74	0.81	0.69	0.74	0.98	0.99	0.99
glove-twitter-200	16.4	0.73	0.72	0.72	0.79	0.69	0.73	0.98	0.99	0.99
glove-wiki-gigaword-50	25.8	0.73	0.73	0.73	0.79	0.71	0.74	0.99	0.99	0.99
glove-wiki-gigaword-300	18.4	0.74	0.70	0.73	0.79	0.70	0.73	0.98	0.99	0.99
word2vec-google-news-300	17.0	0.74	0.71	0.72	0.80	0.69	0.73	0.99	0.99	0.99
fasttext-wiki-news-subwords-300	18.8	0.72	0.72	0.72	0.78	0.67	0.72	0.98	0.99	0.99
Lexical-300	19.0	0.72	0.70	0.71	0.79	0.64	0.71	0.98	0.99	0.99

Table C-107*Evaluation Metrics of 2BiGRU(75)-1CNN-CRF (rmsprop) Per Class*

2BiGRU(75N)-1CNN-CRF (rmsprop)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	83.8	0.69	0.76	0.73	0.76	0.73	0.74	0.99	0.98	0.99
glove-twitter-100	63.4	0.71	0.72	0.71	0.76	0.71	0.73	0.99	0.99	0.99
glove-twitter-200	71.4	0.70	0.74	0.72	0.77	0.70	0.74	0.99	0.99	0.99
glove-wiki-gigaword-50	73.6	0.70	0.73	0.72	0.76	0.72	0.74	0.99	0.99	0.99
glove-wiki-gigaword-300	57.4	0.69	0.75	0.72	0.75	0.70	0.72	0.99	0.98	0.98
word2vec-google-news-300	55.2	0.70	0.73	0.71	0.78	0.68	0.72	0.99	0.99	0.99
fasttext-wiki-news-subwords-300	56.0	0.69	0.72	0.71	0.75	0.69	0.71	0.99	0.98	0.99
Lexical-300	49.0	0.69	0.73	0.71	0.75	0.72	0.72	0.99	0.98	0.99

Table C-108*Evaluation Metrics of 2BiGRU(25)-1CNN-CRF (Adam) Per Class*

2BiGRU(25N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	33.2	0.72	0.72	0.72	0.76	0.70	0.73	0.98	0.99	0.99
glove-twitter-100	24.8	0.72	0.71	0.72	0.79	0.68	0.73	0.99	0.99	0.99
glove-twitter-200	20.2	0.73	0.71	0.72	0.80	0.65	0.71	0.98	0.99	0.99
glove-wiki-gigaword-50	25.8	0.73	0.72	0.72	0.79	0.70	0.74	0.99	0.99	0.99
glove-wiki-gigaword-300	19.4	0.73	0.72	0.73	0.81	0.68	0.73	0.98	0.99	0.99
word2vec-google-news-300	21.0	0.73	0.72	0.73	0.79	0.68	0.73	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	21.6	0.72	0.70	0.71	0.78	0.65	0.71	0.98	0.99	0.99
Lexical-300	20.0	0.72	0.71	0.72	0.79	0.65	0.71	0.98	0.99	0.99

Table C-109*Evaluation Metrics of 2BiGRU(50)-1CNN-CRF (Adam) Per Class*

2BiGRU(50N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	29.2	0.73	0.72	0.72	0.79	0.69	0.74	0.98	0.99	0.99
glove-twitter-100	21.8	0.73	0.72	0.72	0.78	0.70	0.74	0.99	0.99	0.99
glove-twitter-200	18.0	0.72	0.73	0.73	0.80	0.70	0.74	0.99	0.99	0.99
glove-wiki-gigaword-50	28.0	0.72	0.72	0.72	0.81	0.68	0.73	0.99	0.99	0.99
glove-wiki-gigaword-300	17.0	0.73	0.73	0.73	0.78	0.71	0.74	0.99	0.99	0.99
word2vec-google-news-300	17.0	0.73	0.73	0.73	0.80	0.70	0.74	0.99	0.99	0.99
fasttext-wiki-news-subwords-300	18.0	0.72	0.71	0.71	0.81	0.66	0.72	0.98	0.99	0.99
Lexical-300	18.8	0.72	0.72	0.72	0.78	0.68	0.72	0.98	0.99	0.99

Table C-110*Evaluation Metrics of 2BiGRU(75)-1CNN-CRF (Adam) Per Class*

2BiGRU(75N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	35.0	0.73	0.72	0.73	0.81	0.72	0.76	0.98	0.99	0.99
glove-twitter-100	17.6	0.73	0.73	0.73	0.78	0.73	0.75	0.99	0.99	0.99
glove-twitter-200	17.6	0.72	0.77	0.75	0.78	0.71	0.74	0.99	0.99	0.99
glove-wiki-gigaword-50	25.2	0.73	0.71	0.72	0.77	0.71	0.74	0.99	0.99	0.99
glove-wiki-gigaword-300	17.2	0.73	0.71	0.72	0.79	0.68	0.73	0.98	0.99	0.99
word2vec-google-news-300	17.8	0.73	0.73	0.73	0.79	0.69	0.73	0.97	0.99	0.99
fasttext-wiki-news-subwords-300	21.0	0.71	0.71	0.71	0.78	0.69	0.72	0.98	0.99	0.99
Lexical-300	19.8	0.71	0.70	0.71	0.78	0.66	0.72	0.98	0.99	0.99

Table C-111*Evaluation Metrics of 2BiGRU(100)-1CNN-CRF (Adam) Per Class*

2BiGRU(100N)-1CNN-CRF (Adam)										
Embeddings	Epochs	B-FOOD			I-FOOD			O		
		P	R	F1	P	R	F1	P	R	F1
glove-twitter-25	33.2	0.71	0.73	0.72	0.77	0.70	0.72	0.99	0.98	0.99
glove-twitter-100	21.0	0.73	0.72	0.72	0.79	0.70	0.74	0.99	0.99	0.99
glove-twitter-200	18.0	0.73	0.73	0.73	0.78	0.72	0.74	0.99	0.99	0.99
glove-wiki-gigaword-50	24.2	0.72	0.73	0.73	0.79	0.71	0.75	0.98	0.99	0.99
glove-wiki-gigaword-300	17.8	0.74	0.71	0.72	0.79	0.69	0.74	0.98	0.99	0.99
word2vec-google-news-300	20.0	0.73	0.72	0.72	0.79	0.69	0.74	0.98	0.99	0.99
fasttext-wiki-news-subwords-300	17.4	0.72	0.71	0.72	0.81	0.69	0.74	0.98	0.99	0.99
Lexical-300	18.4	0.70	0.72	0.71	0.78	0.67	0.71	0.99	0.99	0.99

C2.6 Best Results Per Experimented Model on *ReviewsDB* Dataset

The table below presents the optimal results for each evaluated model on the *ReviewsDB*, detailing the highest F1-score achieved, the number of trainable parameters involved, and the word embedding model utilized to attain these results.

Table C-112

Best Achieved Results Per Experimented Model on ReviewsDB Dataset

	Model Architectures	F1-score	Trainable Parameters	Word Embeddings
1BiLSTM-CRF	BiLSTM(50N)-CRF (rmsprop) (non-Trainable)	74.07	312,575	glove-twitter-25
	BiLSTM(50N)-CRF (rmsprop)	80.00	571,500	glove-wiki-gigaword-50
			3,160,750	glove-wiki-gigaword-300
	BiLSTM(25N)-CRF (Adam)	80.00	278,805	glove-twitter-25
	BiLSTM(50N)-CRF (Adam)	81.27	312,505	glove-twitter-25
	BiLSTM(75N)-CRF (Adam)	82.07	620,130	glove-wiki-gigaword-50
	BiLSTM(100N)-CRF (Adam)	81.53	3,368,080	Lexical-300
1BiGRU-CRF	BiGRU(50N)-CRF (rmsprop) (non-Trainable)	76.20	556,130	glove-wiki-gigaword-50
	BiGRU(50N)-CRF (rmsprop)	80.27	1,069,050	glove-twitter-100
	BiGRU(25N)-CRF (Adam)	80.40	273,655	glove-twitter-25
	BiGRU(50N)-CRF (Adam)	81.33	1,068,980	glove-twitter-100

Model Architectures		F1- score	Trainable Parameters	Word Embeddings
2BiLSTM- CRF	BiGRU(75N)-CRF (Adam)	81.75	593,430	glove-wiki-gigaword- 50
	BiGRU(100N)-CRF (Adam)	81.73	638,300	glove-wiki-gigaword- 50
	2BiLSTM(50N)-CRF (rmsprop)	80.87	3,221,080	glove-wiki-gigaword- 300
	2BiLSTM(25N)-CRF (Adam)	79.47	2,071,550	glove-twitter-200
	2BiLSTM(50N)-CRF (Adam)	80.87	631,830	glove-wiki-gigaword- 50
	2BiLSTM(75N)-CRF (Adam)	81.67	1,283,650	glove-twitter-100
	2BiLSTM(100N)-CRF (Adam)	80.40	3,608,880	glove-wiki-gigaword- 300
	2BiGRU(50N)-CRF (rmsprop)	80.53	2,140,350	glove-twitter-200
			601,800	glove-wiki-gigaword- 50
	2BiGRU(25N)-CRF (Adam)	80.33	537,950	glove-wiki-gigaword- 50
2BiGRU- CRF	2BiGRU(50N)-CRF (Adam)	81.20	1,114,580	glove-twitter-100
	2BiGRU(75N)-CRF (Adam)	81.80	695,650	glove-wiki-gigaword- 50
	2BiGRU(100N)-CRF (Adam)	81.47	3,458,680	glove-wiki-gigaword- 300

Model Architectures		F1- score	Trainable Parameters	Word Embeddings
1BiLSTM- 1BiGRU- CRF	1BiLSTM(50N)-1BiGRU(50N)-CRF (rmsprop)	81.27	3,287,850	word2vec-google- news-300
	1BiLSTM(25N)-1BiGRU(25N)-CRF (Adam)	81.93	555,100	glove-wiki-gigaword- 50
	1BiLSTM(50N)-1BiGRU(50N)-CRF (Adam)	81.93	357,105	glove-twitter-25
	1BiLSTM(75N)-1BiGRU(75N)-CRF (Adam)	82.20	1,259,880	glove-twitter-100
	1BiLSTM(100N)-1BiGRU(100N)-CRF (Adam)	82.20	813,100	glove-wiki-gigaword- 50
2BiLSTM- 2BiGRU- CRF	2BiLSTM(50N)-2BiGRU(50N)-CRF (rmsprop)	81.27	2,328,150	glove-twitter-200
	2BiLSTM(25N)-2BiGRU(25N)-CRF (Adam)	81.67	1,097,130	glove-twitter-100
	2BiLSTM(50N)-2BiGRU(50N)-CRF (Adam)	82.20	463,105	glove-twitter-25
	2BiLSTM(75N)-2BiGRU(75N)-CRF (Adam)	81.93	947,350	glove-wiki-gigaword- 50
	2BiLSTM(100N)-2BiGRU(100N)-CRF (Adam)	81.80	4,074,350	glove-wiki-gigaword- 300
1BiLSTM- 1CNN- CRF	1BiLSTM(25N)-1CNN-CRF (Adam)	81.33	2,140,430	glove-twitter-200
	1BiLSTM(50N)-1CNN-CRF (Adam)	81.73	2,209,130	glove-twitter-200
	1BiLSTM(75N)-1CNN-CRF (Adam)	81.60	3,373,530	glove-wiki-gigaword- 300
	1BiLSTM(100N)-1CNN-CRF (Adam)	82.00	718,050	glove-wiki-gigaword- 50

Model Architectures		F1- score	Trainable Parameters	Word Embeddings
1BiGRU- 1CNN- CRF	1BiGRU(25N)-1CNN-CRF (Adam)	82.00	3,167,300	glove-wiki-gigaword- 300
	1BiGRU(50N)-1CNN-CRF (Adam)	81.67	595,280	glove-wiki-gigaword- 50
	1BiGRU(75N)-1CNN-CRF (Adam)	82.13	632,650	glove-wiki-gigaword- 50
	1BiGRU(100N)-1CNN-CRF (Adam)	82.07	405,955 3,391,630	glove-twitter-25 glove-wiki-gigaword- 300
1BiLSTM- 1BiGRU- 1CNN- CRF	1BiLSTM(25N)-1BiGRU(25N)-1CNN-CRF (Adam)	81.73	331,625 3,248,550	glove-twitter-25 glove-wiki-gigaword- 300
	1BiLSTM(50N)-1BiGRU(50N)-1CNN-CRF (Adam)	82.13	665,230	glove-wiki-gigaword- 50
	1BiLSTM(75N)-1BiGRU(75N)-1CNN-CRF (Adam)	82.00	374,955 1,177,980	glove-twitter-25 glove-twitter-100
	1BiLSTM(100N)-1BiGRU(100N)-1CNN- CRF (Adam)	82.27	1,232,780	glove-twitter-100
2BiGRU- 1CNN- CRF	2BiGRU(75N)-1CNN-CRF (rmsprop)	81.73	467,125	glove-twitter-25
	2BiGRU(25N)-1CNN-CRF (Adam)	81.73	577,030 3,178,780	glove-wiki-gigaword- 50 glove-wiki-gigaword- 300

Model Architectures	F1- score	Trainable Parameters	Word Embeddings
2BiGRU(50N)-1CNN-CRF (Adam)	82.00	3,280,200	glove-wiki-gigaword-300
2BiGRU(75N)-1CNN-CRF (Adam)	82.40	467,125	glove-twitter-25
		1,270,150	glove-twitter-100
2BiGRU(100N)-1CNN-CRF (Adam)	82.13	858,580	glove-wiki-gigaword-50