



A Comparative Analysis of Deep Learning Models for Facial Expression Recognition at COP28

**التحليل المقارن لنماذج التعلم العميق للتعرف على تعابير الوجه في مؤتمر
COP28**

by

KHALED J. ABDELQADER

**Dissertation submitted in partial fulfilment
of the requirements for the degree of
MSc INFORMATICS
at**

The British University in Dubai

July 2024

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that the content of this research is the direct result of my own work and that any use made in it of published or unpublished copyright material falls within the limits permitted by international copyright conventions.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

Facial expression recognition (FER) has primarily been studied in controlled environments, which do not adequately represent the dynamic conditions of real-world events. This study aims to bridge this gap by conducting a comparative analysis of deep learning models tailored to the context of COP28, a large-scale public event in the UAE. The research evaluated several deep learning architectures, including the standard convolutional neural network (CNN), expanded CNN, multi-path CNN, pre-trained transfer learning Visual Geometry Group 16 (VGG16), recurrent neural network (RNN), long short-term memory (LSTM), gated recurrent unit (GRU), and vision transformer (ViT) ones, using the Karolinska Directed Emotional Faces (KDEF) dataset for training and validation. Performance was then tested on unseen images from COP28. The findings revealed that CNN-based models, particularly the multi-path CNN and VGG16, outperformed sequential models such as the RNN, LSTM, and GRU, with the multi-path CNN achieving the highest accuracy (71.70%) on COP28 test images. The expanded CNN showed a strong performance on the KDEF dataset, achieving a training accuracy of 97.7%. Although most models performed exceptionally well in controlled settings, their performance varied when applied to COP28, indicating areas for improvement in generalization. The limitations included dataset constraints and performance variability across different expressions. Future research should focus on diverse datasets, advanced data augmentation, ensemble methods, newer architectures, multimodal approaches, and understanding cultural impacts on FER while considering ethical implications. This work underscores the importance of adapting FER models to real-world scenarios and provides insights for improving FER accuracy and generalization.

Keywords: facial expression recognition, CNN, COP28

خلاصة

لقد تم دراسة التعرف على تعابير الوجه (FER) بشكل رئيسي في البيانات المسيطر عليها، والتي لا تمثل بشكل كاف الظروف الديناميكية للأحداث الواقعية. تهدف هذه الدراسة إلى سد هذه الفجوة من خلال إجراء تحليل مقارن لنماذج التعلم العميق المصممة خصيصاً لسياق مؤتمر COP28، وهو حدث عام واسع النطاق في الإمارات العربية المتحدة. قامت الدراسة بتقييم العديد من بنى نماذج التعلم العميق، بما في ذلك الشبكة العصبية التلافيفية القياسية (standard CNN)، الشبكة العصبية التلافيفية الموسعة (expanded CNN)، الشبكة العصبية التلافيفية متعددة المسارات (multi-path CNN)، نموذج نقل التعلم المدرب مسبقاً VGG16، الشبكة العصبية التكرارية (RNN)، الذاكرة الطويلة قصيرة الأمد (LSTM)، وحدة الذاكرة التكرارية (GRU)، والمحولات المرئية (ViT)، باستخدام مجموعة بيانات الوجوه التعبيرية الموجهة كارولينسكا (KDEF) للتدريب والتقييم. ثم تم اختبار الأداء على صور غير مدرب عليها مسبقاً من مؤتمر COP28. كشفت النتائج أن النماذج المعتمدة على الشبكات العصبية التلافيفية، ولا سيما الشبكة العصبية التلافيفية متعددة المسارات ونموذج VGG16، تفوقت على النماذج التتابعية مثل الشبكة العصبية التكرارية، الذاكرة الطويلة قصيرة الأمد، ووحدة الذاكرة التكرارية، حيث حققت الشبكة العصبية التلافيفية متعددة المسارات أعلى دقة (71.70%) على صور COP28. أظهرت الشبكة العصبية التلافيفية الموسعة أداءً قوياً على مجموعة بيانات KDEF، حيث حققت دقة تدريب بلغت 97.7%. على الرغم من أن معظم النماذج أدت بشكل استثنائي في البيانات المسيطر عليها، إلا أن أدائها تنوع عند تطبيقها على صور COP28، مما يشير إلى مجالات للتحسين في حالة التعميم. تضمنت القيود قيود مجموعة البيانات وتفاوت الأداء عبر التعبيرات المختلفة. يجب أن تركز الأبحاث المستقبلية على مجموعات بيانات متنوعة، تقنيات تحسين البيانات المتقدمة، طرق التجميع، نماذج تعلم عميق أحدث، النهج المتعدد الوسائط، وفهم التأثيرات الثقافية على التعرف على تعابير الوجه مع مراعاة الأبعاد الأخلاقية. تؤكد هذه الدراسة على أهمية تكيف نماذج التعرف على تعابير الوجه مع السيناريوهات الواقعية وتوفير رؤى لتحسين دقة التعرف والتعميم.

Dedication

To my supporter through every joy and challenge, my dear wife Eman Almaini, and to the light of my life, my children Hisham, Qusai, and Lubna.

Acknowledgments

I would like to express my deepest gratitude to Prof. Khaled Shaalan for his continuous support and guidance throughout this journey. Your encouragement and insights have been invaluable.

I extend my thanks to Dr. Manar Al Khatib for her support and assistance, which have greatly contributed to the success of this work.

I am also grateful to Hisham Abdelqader and Hashim AbuHassan for their valuable advice on new deep learning models.

I would like to thank Mohamed Elrashid for his advice on Python programming.

Finally, I would like to thank Eman Almaini and Lubna Assaf for their continuous motivation and support.

Table of Contents

Chapter 1: Introduction.....	1
1.1 Challenges in FER	1
1.1.1 Challenges in Real-World Application.....	2
1.2 Deep Learning Techniques in FER.....	5
1.2.1 CNNs	6
1.2.2 Recurrent Neural Networks (RNNs) and Long Short-Term Memory Networks (LSTMs)	7
1.2.3 Vision Transformer (ViT)	7
1.2.4 Generative Adversarial Networks (GANs).....	7
1.2.5 Transfer Learning	8
1.3 FER Datasets.....	8
1.3.1 Classification of FER Datasets	8
1.4 Steps of FER by Deep Learning	11
1.5 About This Study	14
1.5.1 Research Questions.....	16
Chapter 2: Literature Review	17
2.1 Datasets	17
2.2 Preprocessing	18
2.3 Deep Learning Models.....	19
2.4 Performance Evaluation.....	20
2.5 Crowd and Public Image Testing.....	21
2.6 Challenges.....	21

2.7 Suggested Future Research Directions	23
2.8 Summary of Literature Review.....	24
Chapter 3: Methodology	25
3.1 Data Collection and Preprocessing	25
3.1.1 Dataset	25
3.1.2 KDEF Preprocessing	25
3.1.3 COP28 Data Collection and Preprocessing	30
3.2 The Deep Learning Models for FER Examined in This Study.....	32
3.2.1 The Standard CNN Model	32
3.2.2 The Expanded CNN Model	39
3.2.3 The Multi-Path CNN-MassAtt-PWFS Model	44
3.2.4 The Transfer Learning VGG16 Model	49
3.2.5 The RNN Model	51
3.2.6 The LSTM Model	54
3.2.7 The GRU Model	57
3.2.8 The Vision Transformer (ViT) Model.....	59
Chapter 4: The Results	64
4.1 KDEF Dataset Training Results	64
4.1.1 Consolidated Training Results for All Models.....	64
4.1.2 Standard CNN 64 × 64 Input Image Size (100 Epochs).....	67
4.1.3 Standard CNN 128 × 128 Input Image Size (100 Epochs).....	69
4.1.4 Standard CNN 224 × 224 Input Image Size (80 Epochs)	71
4.1.5 Expanded CNN 128 × 128 Input Image Size (80 Epochs)	73
4.1.6 Multi-Path CNN 64 × 64 Input Image Size (30 Epochs).....	75

4.1.7 VGG16 224×224 Input Image Size (30 Epochs).....	77
4.1.8 RNN 128×128 Input Image Size (120 Epochs)	79
4.1.9 LSTM 128×128 Input Image Size (120 Epochs).....	81
4.1.10 GRU 128×128 Input Image Size (120 Epochs)	83
4.1.11 ViT 128×128 Input Image Size (120 Epochs)	85
4.2 COP28 FER Results.....	87
4.2.1 Consolidated COP28 FER Prediction Results (Accuracy and F1)	87
4.2.2 Detailed Facial Expression Results for COP28 Test Images.....	89
4.2.3 Examples of COP28 FER Results for All Models.....	95
Chapter 5: Discussion and Evaluation	99
5.1 Consolidation Analysis of KDEF Training and Validation Results	99
5.2 Consolidation Analysis of the COP28 Prediction Results.....	100
5.3 Analysis of the Standard CNN Model Results (Training and COP28)	103
5.4 Analysis of the Expanded CNN Model Results (Training and COP28).....	106
5.5 Analysis of Multi-Path CNN Model Results (Training and COP28)	109
5.6 Transfer Learning VGG16 Model Results (Training and COP28).....	112
5.7 RNN Model Results (Training and COP28).....	115
5.8 LSTM Model Results (Training and COP28).....	117
5.9 GRU Model Results (Training and COP28).....	119
5.10 ViT Model Results (Training and COP28).....	121
Chapter 6: Conclusion, Limitations, and Future Works.....	124
6.1 Summary of Findings.....	124
6.2 Study Limitations.....	125

6.3 Addressing the Research Questions.....	126
6.4 Future Research Directions.....	128
6.5 Final Remarks	129
References	130
Appendices	146
Appendix A: Multi-Path CNN Model Parameters.....	146
Appendix B: Transfer Learning VGG16 Parameters.	149

List of Tables

Table 1: Accuracy and F1 Training Results for All Models on the KDEF Dataset.	65
Table 2: Standard CNN Training Summary for 64×64 Images.	68
Table 3: Standard CNN Training Summary for 128×128 Images.	70
Table 4: Standard CNN Training Summary for 224×224 Images.	72
Table 5: Expanded CNN Training Summary for 128×128 Images.	74
Table 6: Multi-Path CNN Training Summary for 64×64 Images.	76
Table 7: Multi-Path CNN Training Summary for 224×224 Images.	78
Table 8: RNN Training Summary for 128×128 Images.	80
Table 9: LSTM Training Summary for 128×128 Images.	82
Table 10: GRU Training Summary for 128×128 Images.	84
Table 11: ViT Training Summary for 128×128 Images.	86
Table 12: COP28 FER Models Results.	88
Table 13: COP28 Facial Expression per Metric per Model.	92

List of Figures

Figure 1: Examples of Real Instances Featuring Various Types of Facial Occlusion.	2
Figure 2: The Appearance of the Same Person Under Different Lighting Conditions.	3
Figure 3: Mixed and Complex Emotions Beyond the Main Seven Emotions.	4
Figure 4: Model of Convolutional Neural Network (CNN).	6
Figure 5: Steps of FER by Deep Learning.	13
Figure 6: Sample of KDEP.	26
Figure 7: Crop 1:1 of Facial Detection Using the MediaPipe Library.	27
Figure 8: Histogram Equalizer Result by CLAHE.	29
Figure 9: Augmentation of One Image to Five More Images.	30
Figure 10: Samples of COP28 Test Data Preprocessing.	32
Figure 11: The Design of the Standard CNN Model for 128×128 Input Images.	33
Figure 12: 3×3 Kernel on Input CNN Layer.	34
Figure 13: ReLU Function.	36
Figure 14: Example of 3×3 Max Pooling.	37
Figure 15: 2D to 1D Flattening.	38
Figure 16: Example of Fully Connected Dense Layers.	39
Figure 17: The Design of the Expanded CNN Model for 128×128 Input Images.	41
Figure 18: Multi-Path CNN Model.	44
Figure 19: Transfer Learning VGG16 Model.	50
Figure 20: RNN Structure.	52
Figure 21: The Design of the RNN Model for 128×128 Input Images.	53

Figure 22: LSTM Structure.	55
Figure 23: The Design of the LSTM Model for 128×128 Input Images.	56
Figure 24: GRU Structure.	58
Figure 25: The Design of GRU Model for 128×128 Input Images.	59
Figure 26: Vision Transformer (ViT) Architecture.....	60
Figure 27: The Design of ViT Model for 128×128 Input Images.	62
Figure 28: Training Results for the KDEP Dataset.....	64
Figure 29: Standard CNN Training Results for 64×64 Images (100 Epochs).	67
Figure 30: Standard CNN Training Results for 128×128 Images (100 Epochs).	69
Figure 31: Standard CNN Training Results for 224×224 Images (100 Epochs).	71
Figure 32: Expanded CNN Training Results for 128×128 Images (80 Epochs).....	73
Figure 33: Multi-Path CNN Training Results for 64×64 Images (30 Epochs).	75
Figure 34: VGG16 Training Results for 224×224 Images (30 Epochs).	77
Figure 35: RNN Training Results for 128×128 Images (120 Epochs).....	79
Figure 36: LSTM Training Results for 128×128 Images (120 Epochs).	81
Figure 37: GRU Training Results for 128×128 Images (120 Epochs).....	83
Figure 38: ViT Training Results for 128×128 Images (120 Epochs).....	85
Figure 39: The Results for COP28 Unseen Test Images for All Models.....	87
Figure 40: Facial Expression Results for COP28 Test Images (Models-Based).	90
Figure 41: Results for COP28 Test Images (Facial Expressions-Based).....	91
Figure 42: Sample of COP28 Prediction Results for Standard CNN 128×128	95
Figure 43: Sample of COP28 Prediction Results for Expanded CNN Model.	95
Figure 44: Sample of COP28 Prediction Results for Multi-Path CNN Model.	96

Figure 45: Sample of COP28 Prediction Results for VGG16 Model.	96
Figure 46: Sample of COP28 Prediction Results for RNN Model.	97
Figure 47: Sample of COP28 Prediction Results for LSTM Model.	97
Figure 48: Sample of COP28 Prediction Results for GRU Model.	98
Figure 49: Sample of COP28 Prediction Results for ViT Model.	98

Chapter 1: Introduction

Facial expression (emotion) recognition (FER) is a rapidly evolving field within computer vision and artificial intelligence that focuses on the identification and classification of human emotions through facial movements (Appasaheb Borgalli & Surve, 2022; Canedo & Neves, 2019; Kopalidis et al., 2024). This technology has a wide range of applications, from human-computer interaction and healthcare to security and entertainment. FER systems analyze facial features to detect expressions such as happiness, sadness, anger, surprise, fear, and disgust, providing valuable insights into human emotions and behaviors (Mohana & Subashini, 2024).

Traditional approaches to FER relied on handcrafted features and shallow learning methods, such as support vector machines (SVMs) (Cortes & Vapnik, 1995) and k-nearest neighbors (k-NNs) (Cover & Hart, 1967). These methods, while effective to some extent, often struggled with variability in facial expressions due to differences in lighting, pose, and occlusions. Feature extraction techniques such as the histogram of oriented gradients (HOG) (Dalal & Triggs, n.d.) and local binary patterns (LBPs) (Ojala et al., 2002) were commonly used but had limitations in capturing the complexities of facial expressions (Z. Hu, 2023).

1.1 Challenges in FER

One of the significant challenges in FER is the variability in facial expressions due to differences in lighting, pose, occlusions, and individual differences. Real-world applications, such as recognizing expressions in crowded events or public gatherings, add additional layers

of complexity. Occlusions due to accessories like hats and glasses, varied lighting conditions, and the presence of multiple individuals can significantly impact the performance of FER systems.

1.1.1 Challenges in Real-World Application

Occlusions: Faces are often partially covered by various objects such as accessories (e.g., hats, sunglasses, scarves), hands, or other faces (see Figure 1), hindering FER systems in accurately detecting and interpreting facial expressions. For instance, sunglasses can conceal eye expressions, and hands covering the mouth can obscure expressions like surprise or sadness (Zeng et al., 2021; L. Zhang et al., 2019).

Figure 1

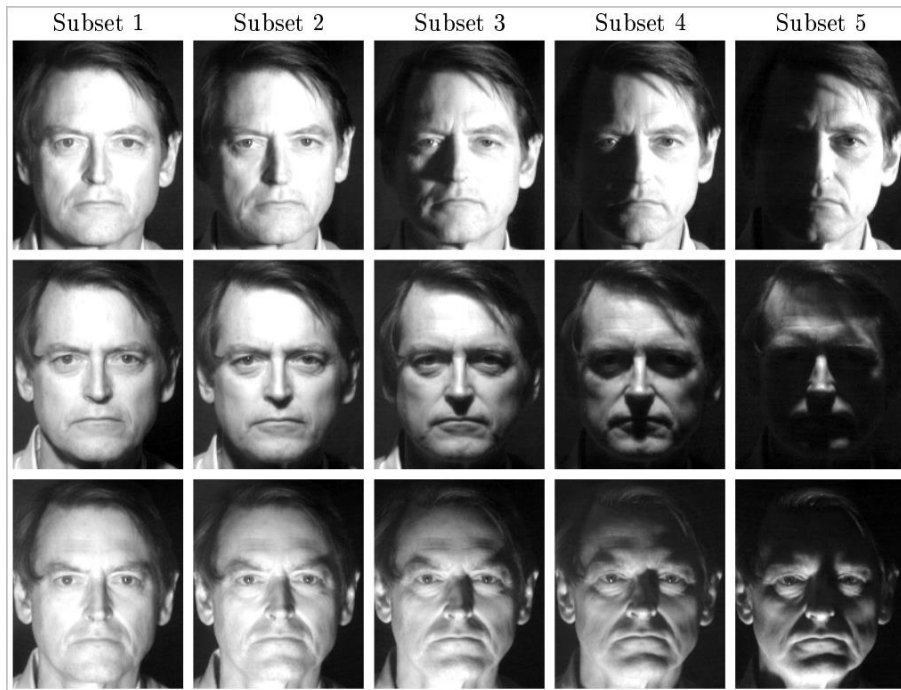
Examples of Real Instances Featuring Various Types of Facial Occlusion (L. Zhang et al., 2019).



Lighting Variability: Lighting conditions can vary significantly across different environments, affecting the appearance of facial features (Belhumeur et al., 1996), see Figure 2. FER systems trained on uniformly lit images may struggle with images taken under different lighting conditions, such as harsh shadows, overexposure, or low light, leading to inaccurate recognition (Stratou et al., 2011).

Figure 2

The Appearance of the Same Person Can Vary Significantly Under Different Lighting Conditions.



Pose Variability: Faces are rarely captured perfectly head-on in real-world scenarios. Different angles can occlude or distort facial features, complicating expression recognition. Advanced FER systems must use pose normalization techniques or train on diverse images showing faces from various angles to improve robustness (Naser et al., 2023).

Individual Differences: Variations in facial structure, age, gender, and cultural background affect how expressions are displayed and recognized. For example, cultural differences may alter the way emotions are expressed, and aging can change the appearance of facial muscles. FER systems must be trained on diverse datasets to accurately recognize expressions across different demographics (Alharbi et al., 2020).

Emotional Subtleties: Human emotions are complex and often subtle. FER systems may struggle to recognize subtle expressions or mixed emotions, where multiple emotions are displayed simultaneously (see Figure 3). Advanced models capable of fine-grained analysis are needed to capture these nuanced expressions (Du et al., 2014).

Figure 3

Mixed and Complex Emotions Beyond the Main Seven Emotions, Labeled from A to G (Du et al., 2014).



Contextual Understanding: Facial expressions are influenced by context. For instance, a smile can indicate happiness, sarcasm, or discomfort depending on the situation. FER systems lacking contextual awareness might misinterpret expressions. Integrating contextual information, such as body language or environmental cues, can improve accuracy (Martinez & Valstar, 2016).

Data Bias and Fairness: Training data for FER systems can be biased, leading to models that perform better on certain demographic groups than others. This can result in unfair or inaccurate outcomes, particularly for underrepresented groups. Addressing data bias and ensuring fairness requires diverse training data and continuous model performance evaluation (Xu et al., 2020).

Adaptability and Personalization: Users may have unique ways of expressing emotions. FER systems that can adapt to individual differences and personalize their responses based on user-specific patterns offer improved accuracy and user satisfaction. Implementing such adaptability requires sophisticated learning algorithms and continuous interaction with users (Rescigno et al., 2020).

1.2 Deep Learning Techniques in FER

To address the various challenges faced by FER systems in real-world applications, deep learning has emerged as part of the solution. Deep learning models, particularly convolutional neural networks (CNNs) (LeCun et al., 1989), have demonstrated remarkable accuracy and robustness in recognizing facial expressions. By leveraging large and diverse datasets, these models can learn intricate patterns and variations in facial features, making them well

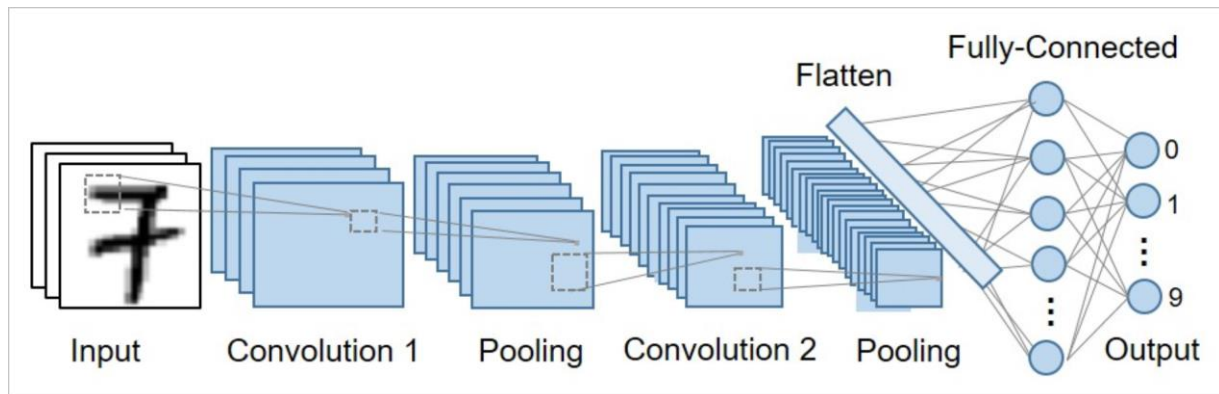
suited to handle occlusions, lighting and pose variability, and individual differences (K. Liu et al., 2016; Wicaksana et al., 2024).

1.2.1 CNNs

CNNs are the backbone of modern FER systems due to their ability to learn and extract features directly from images (K. Liu et al., 2016). They consist of multiple layers, including convolutional layers, pooling layers, and fully connected layers (see Figure 4), which enable them to capture and recognize complex patterns in facial expressions (Wang et al., 2021).

Figure 4

Model of Convolutional Neural Network (CNN) (Bui et al., 2020).



VGGNet: A CNN model known for its simplicity and depth, VGGNet uses small (3x3) convolution filters and has demonstrated high accuracy in FER tasks (Simonyan & Zisserman, 2014).

ResNet: ResNet introduces residual learning, allowing for deeper networks by addressing the vanishing gradient problem. ResNet's skip connections help maintain performance even with increasing depth (He et al., 2015).

Inception: Inception uses a novel architecture that incorporates multiple convolution filters of different sizes in parallel, capturing features at various scales and improving accuracy and efficiency (Szegedy et al., 2014).

1.2.2 Recurrent Neural Networks (RNNs) and Long Short-Term Memory Networks (LSTMs)

RNNs (Williams & Zipser, 1989) and LSTMs (Hochreiter & Schmidhuber, 1997) are employed in FER to capture temporal dependencies in video sequences, processing sequential data and making them suitable for the dynamic analysis of facial expressions over time. LSTMs address the limitations of traditional RNNs by incorporating memory cells that store and retrieve information over long periods, enhancing the recognition of expressions in continuous video streams (Yu et al., 2018).

1.2.3 Vision Transformer (ViT)

The vision transformer (ViT), proposed by Dosovitskiy et al. (2020), applies the transformer architecture to image patches, treating them similarly to word tokens in natural language processing (NLP). This innovative approach adapts the powerful self-attention mechanism for image classification tasks, demonstrating the flexibility and scalability of transformer-based models in computer vision. Transformers, initially introduced in a seminal paper (Vaswani et al., 2017), revolutionized NLP by relying solely on self-attention mechanisms to capture dependencies between input tokens.

1.2.4 Generative Adversarial Networks (GANs)

GANs (Goodfellow et al., 2014) have been explored for FER to augment datasets and generate realistic facial expressions. GANs consist of a generator and a discriminator network

that work in tandem to produce and refine synthetic images, which can be used to enhance training datasets and improve model performance (Rani et al., 2024; Yang et al., 2018).

1.2.5 Transfer Learning

Transfer learning (Donahue et al., 2014) leverages pre-trained models on large datasets (e.g., ImageNet; Krizhevsky et al., 2017) and fine-tunes them for specific FER tasks. This approach reduces training time and improves performance, especially when labeled FER data are limited. Pre-trained models like VGG16, ResNet, and Inception can be adapted for FER, retaining their learned features while being fine-tuned on facial expression datasets (Akhand et al., 2021).

1.3 FER Datasets

Datasets play an essential role in the development and evaluation of FER systems. Commonly used datasets include the Karolinska Directed Emotional Faces (KDEF) (Lundqvist et al., 1998), FER2013 (Goodfellow et al., 2013), Cohn-Kanade AU-Coded Expression Database (CK+) (Lucey et al., 2010), and AffectNet (Mollahosseini et al., 2019). These datasets provide a variety of labeled facial expressions that are essential for training and testing FER models. However, many of these datasets have limitations, such as a lack of diversity in facial features (e.g., facial hair or glasses), controlled environments that do not represent real-world conditions, and insufficient representation of subtle expressions (Mejia-Escobar et al., 2023).

1.3.1 Classification of FER Datasets

1.3.1.1 Controlled Environment Datasets. These datasets consist of images or videos

captured under standardized conditions, such as consistent lighting, background, and camera angles, to ensure uniformity and ease of analysis (Kopalidis et al., 2024).

- **Karolinska Directed Emotional Faces (KDEF):** This dataset consists of 4,900 images of 70 amateur actors, each displaying seven distinct emotional expressions from five different angles (Lundqvist, 1998). While comprehensive, it lacks diversity in facial features and was collected under controlled lighting conditions.
- **Cohn-Kanade AU-Coded Expression Database (CK+):** Comprising 593 sequences from 123 subjects, CK+ provides high-resolution video sequences of facial expressions (Lucey et al., 2010). However, it was also collected in a controlled environment, which may not reflect real-world variability.
- **Japanese Female Facial Expression (JAFPE):** This dataset contains 213 images of 10 Japanese female models displaying six basic emotions and a neutral expression (Lyons et al., 1998). While useful for certain studies, it has a limited number of subjects and controlled settings.
- **Multimedia Understanding Group (MUG) Facial Expression Database:** This dataset includes video sequences and still images of 86 subjects displaying various expressions (Aifanti et al., 2010). It provides detailed annotations but was also collected under controlled conditions.

1.3.1.2 In-the-Wild Datasets. These datasets consist of images or videos captured in natural, uncontrolled environments, which include varying lighting, backgrounds, and spontaneous expressions, providing a more realistic and diverse representation of facial expressions

(Aguilera et al., 2023).

- **FER-2013:** Collected from the internet, this dataset contains 35,887 grayscale images categorized into seven emotion classes (Goodfellow et al., 2013). It represents more natural and varied expressions but still faces challenges in terms of image quality and occlusions.
- **AffectNet:** One of the largest FER datasets, AffectNet contains over 1 million images collected from the internet, annotated with both categorical and continuous emotion labels (Mollahosseini et al., 2019). It offers greater diversity in expressions and conditions, making it valuable for training robust models.
- **SFEW (Static Facial Expressions in the Wild):** This dataset is derived from movies and contains images with various facial expressions in uncontrolled environments, making it useful for real-world applications (Dhall et al., 2011).
- **Real-world Affective Faces Database (RAF-DB):** This dataset includes 29,672 images collected from the internet, with diverse facial expressions and rich annotations. It provides a realistic benchmark for evaluating FER systems in natural settings (Li & Deng, 2019).
- **Expression in-the-Wild (ExpW):** Collected from the internet, ExpW contains approximately 91,793 images (Z. Zhang et al., 2016). However, the dataset includes an uneven distribution of 30 expressions, with some categories overrepresented and others underrepresented. It also has redundant categories like ‘sad’ and ‘unhappy’. Additionally,

the presence of non-human and non-facial images introduces noise, necessitating extra filtration.

1.4 Steps of FER by Deep Learning

The initial phase involves **dataset collection**, where labeled images are gathered. These images must be accurately annotated with the correct facial expression labels to serve as a reliable foundation for the subsequent model training.

Once the dataset is collected, the next step is **data preprocessing**. This phase encompasses several key operations. Firstly, face detection algorithms are applied to locate and isolate faces within the images. After detection, the images are cropped to focus solely on the facial region. Following cropping, the images are converted to grayscale to simplify the data and reduce computational complexity. To enhance the dataset, augmentation techniques are used, which involve generating additional training data by applying transformations such as rotations, shifting, zooming, and flips. Finally, histogram equalization is performed to improve the contrast of the images, thereby enhancing the feature extraction process during model training.

With the data preprocessed, the dataset is then split into two subsets: 80% for training and 20% for validation. This **dataset splitting** ensures that the model can be evaluated on unseen data during training, providing a measure of its generalizability.

The next stage is to **build the model**. This involves selecting an appropriate deep learning architecture and configuring it with the necessary layers and parameters. The model is then trained using the preprocessed training data. During the **train the model** phase, the model

learns to map input images to the corresponding facial expressions by minimizing the error between the predicted and actual labels.

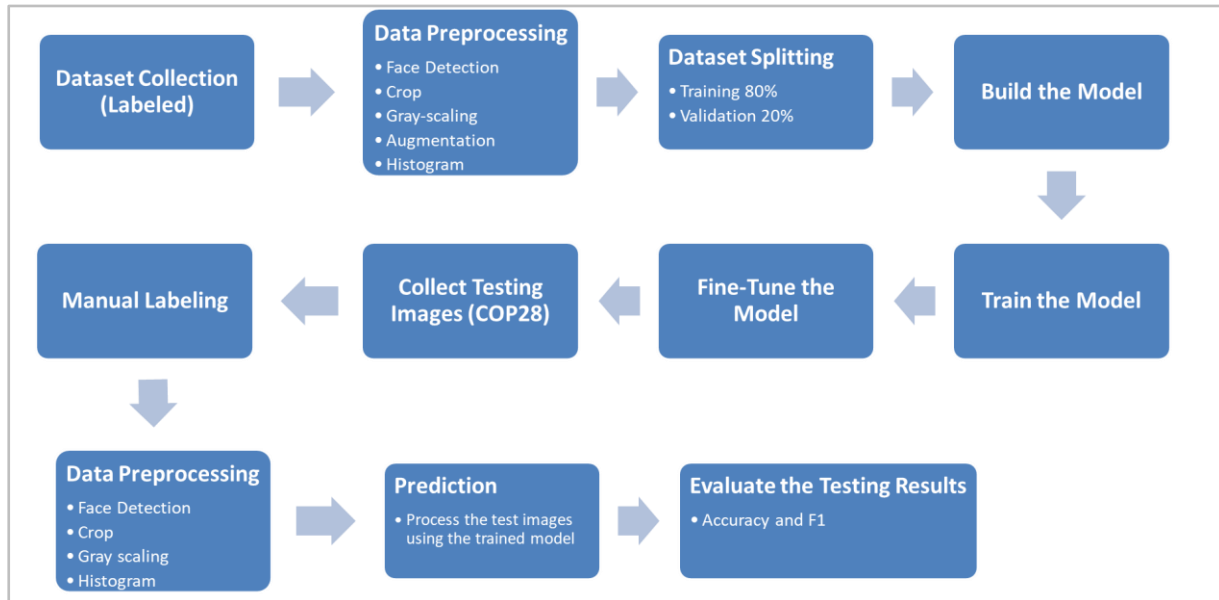
Once the initial training is complete, the model undergoes **fine-tuning**. This involves adjusting the model's hyperparameters and possibly retraining certain layers to improve performance based on the validation set results.

To evaluate the model in a real-world context, testing images are collected, particularly from events such as COP28¹. These images require **manual labeling** to ensure the accuracy of the ground truth labels. The collected testing images undergo similar preprocessing steps as the training data, including face detection, cropping, grayscale conversion, and histogram equalization (see Figure 5).

¹ <https://www.cop28.com/>

Figure 5

Steps of FER by Deep Learning.



In the **prediction** phase, the preprocessed test images are processed by the trained model to predict the facial expressions. Finally, the model's performance is assessed in the **evaluate the testing results** phase, using metrics such as accuracy and the F1 score. These metrics provide a comprehensive evaluation of the model's ability to correctly identify facial expressions in the testing set, reflecting its overall effectiveness and reliability.

This systematic approach to FER using deep learning ensures that each step is carefully executed, from data collection and preprocessing to model training and evaluation, ultimately aiming to achieve high accuracy and robust performance in recognizing facial expressions.

1.5 About this Study

FER has predominantly been explored in controlled environments or using datasets sourced from the internet, which often fail to capture the dynamic and varied conditions of real-world situations. A significant research gap exists in applying FER to large-scale public events, particularly those in the UAE, such as COP28. Events like COP28 present unique challenges for FER systems due to the diversity of attendees, the presence of various cultural expressions, and environmental variability including lighting, pose, and occlusions. Addressing this gap is crucial for enhancing the robustness and applicability of FER technologies in real-world scenarios, where they can provide valuable insights into public sentiment and improve human-computer interaction.

This study aims to bridge this research gap by conducting a comparative analysis of deep learning models for FER specifically tailored to the context of COP28. By focusing on a significant event in the UAE, the work highlights the unique challenges and opportunities presented by such settings.

The main contribution of this research lies in its comprehensive evaluation of multiple deep learning architectures, including the following models:

1. Standard CNN
2. Expanded CNN (Exp-CNN)
3. Multi-path CNN-mass attention (MassAtt) - Point-Wise Feature Selection (PWFS) multi-path
4. Pre-trained learning transfer VGG16
5. RNN
6. LSTM

7. GRU
8. Vision transformer (ViT)

Evaluating and comparing all these deep learning models based on the KDEF dataset and a public event like COP28 is unprecedented. This study is the first to bring together such a diverse range of models for comparison in one comprehensive evaluation. By doing so, it provides novel insights into the strengths and weaknesses of each model in a real-world scenario. This thorough comparison not only advances the understanding of FER in complex environments but also offers practical guidance for selecting the most effective model for similar applications. This extensive and diverse model comparison represents a significant and unique contribution to the field of facial expression recognition research.

The study also addresses the challenge of limited facial expression data by physically augmenting the KDEF dataset by a factor of 6, to improve the robustness and diversity of data for FER training. This enhancement through physical augmentation allows for a more evaluation of various models, providing insights into their respective strengths and limitations and guiding the optimization of architectures and training strategies for FER.

In summary, this study makes two primary contributions: Firstly, it addresses the underexplored area of FER in large-scale public events in the UAE, such as COP28, providing empirical data and analysis that can guide future research and applications. Secondly, the work undertakes an extensive evaluation of various deep learning models, including standard CNN, expanded CNN, multi-path CNN, VGG16, RNN, LSTM, GRU, and ViT models. This first-of-its-kind comparison provides fresh insights into each model's performance in practical, real-world conditions, offering valuable guidance for future FER applications.

1.5.1 Research Questions

- RQ1.** What are the strengths and limitations of deep learning models trained on controlled datasets when applied to real-world images from events like COP28?
- RQ2.** How do different image resolutions (64×64 , 128×128 , and 224×224 pixels) impact the performance of CNN models in FER on controlled datasets like KDEF?
- RQ3.** How does transfer learning with VGG16 compare to training from scratch in terms of accuracy and training time for FER?
- RQ4.** What are the comparative performances of non-CNN models (RNN, LSTM, GRU, and ViT) versus CNN-based models in FER, particularly in the unique settings of events like COP28?

Chapter 2: Literature Review

This literature review examines 15 studies (Akhand et al., 2021; Appasaheb Borgalli & Surve, 2022; Bhagat et al., 2024; Chowdary et al., 2023; Gautam & Seeja, 2023; Holder & Tapamo, 2018; Z. Hu, 2023; Khaireddin & Chen, 2021; Khanzada et al., 2020; K. Liu et al., 2016; Mejia-Escobar et al., 2023; Minaee et al., 2021; Nan et al., 2022; Puthanidam & Moh, 2018; Santoso & Kusuma, 2022) on FER, integrating the findings to provide a comprehensive overview of the **datasets** used, **preprocessing techniques** applied, **deep learning** models employed, **performance metrics** evaluated, **crowd and public images** tested, **challenges** encountered, and **future research directions** suggested.

2.1 Datasets

FER has been extensively studied using a variety of datasets and methodologies to improve the accuracy and applicability of FER systems. Several studies, including those by Appasaheb Borgalli and Surve (2022), Z. Hu (2023), and Puthanidam and Moh (2018), selected the JAFFE dataset, while the KDEF dataset was utilized by Akhand et al. (2021), Appasaheb Borgalli and Surve (2022), and Puthanidam and Moh (2018). The FER2013 dataset was prominently used in research by Bhagat et al. (2024), Khaireddin and Chen (2021), Khanzada et al. (2020), K. Liu et al. (2016), Mejia-Escobar et al. (2023), Minaee et al. (2021), and Santoso and Kusuma (2022). The SFEW dataset was employed by Appasaheb Borgalli and Surve (2022), Nan et al. (2022), and Puthanidam and Moh (2018). The CK+ dataset was adopted in papers by Chowdary et al. (2023), Gautam and Seeja (2023), Khanzada et al. (2020), and Minaee et al. (2021). The RAF-DB dataset was utilized by Appasaheb Borgalli and Surve (2022) and Nan et al. (2022), and the Affectiva-MIT Facial Expression (AMFED+) dataset (McDuff et al., 2013) by Appasaheb Borgalli and Surve (2022). Mejia-Escobar et al. (2023)

chose the natural human face image (NHFI) (Vaidya, 2020) and AffectNet datasets. The Facial Expression Research Group (FERG) dataset (Aneja et al., 2017) was employed by Minaee et al. (2021). These diverse datasets were implemented to explore various FER challenges and solutions.

2.2 Preprocessing

The literature on FER highlights the variety of preprocessing techniques employed. Several studies, including Z. Hu (2023) and Puthanidam and Moh (2018), converted photos to grayscale and applied normalization. Data augmentation was widely used, as seen in works by K. Liu et al. (2016) and Santoso and Kusuma (2022), who implemented techniques such as random cropping, horizontal flipping, and generating synthetic images through small rotations to prevent overfitting. Appasaheb Borgalli and Surve (2022), Holder and Tapamo (2018), and Z. Hu (2023) chose the Viola-Jones framework (Viola & Jones, 2001) for face detection, with the latter also employing multiple preprocessing steps to enhance image quality. Holder and Tapamo (2018) chose the gradient local ternary pattern (GLTP) (Ahmed & Hossain, 2013) for facial feature extraction, while Akhand et al. (2021) and Bhagat et al. (2024) applied standard techniques like cropping, resizing, and normalization. Mejia-Escobar et al. (2023) utilized data-centric refinement techniques and synthetic image generation to balance their dataset, while Khanzada et al. (2020) employed normalization, data augmentation, class weighting, ensembling, and test-time augmentation. Khairuddin and Chen (2021) applied extensive data augmentation techniques, including rescaling, shifting, rotating, and cropping, followed by normalization. Nan et al. (2022) selected a GAN-based feature level super-resolution method for low-resolution images, while Gautam and Seeja (2023) applied HOG and scale-invariant

feature transform (SIFT) (Lowe, 2004) techniques for feature extraction after resizing and normalizing images. Finally, Minaee et al. (2021) utilized typical image augmentation techniques and incorporated a spatial transformer network to focus on salient facial regions.

2.3 Deep Learning Models

The studies reviewed highlight the widespread use of CNNs and their variants. Z. Hu (2023) compared k-NN, SVM, and CNN, underscoring CNN's superior performance due to its ability to extract higher and multi-level features. Puthanidam and Moh (2018) evaluated four CNN architectures: Caffe (Jia et al., 2014), Kahou's structure (Kahou et al., 2013), Tang's structure (Tang, 2013), and Yu's structure (Yu & Zhang, 2015), demonstrating varied accuracy rates across different datasets. Appasaheb Borgalli and Surve (2022) employed both custom and standard CNN architectures, including VGG16 and InceptionResNetV2 (Szegedy et al., 2017), achieving high accuracy on lab-controlled datasets but lower performance on non-lab-controlled datasets. Santoso and Kusuma (2022) introduced the VGGSpinalNet architecture, achieving significant accuracy improvements. K. Liu et al. (2016) applied an ensemble method combining multiple CNN subnets, enhancing accuracy through ensemble learning. Khanzada et al. (2020) leveraged transfer learning with pre-trained models like ResNet50 and SeNet50 (J. Hu et al., 2017), achieving state-of-the-art accuracy through transfer learning and data augmentation. Bhagat et al. (2024) integrated EfficientNet (Tan and Le, 2019) and ResNet into a deep CNN (DCNN) for real-time FER, demonstrating the practical applicability of advanced CNN architectures.

Additionally, Akhand et al. (2021) explored the utilization of DCNN with transfer learning, achieving high accuracy with models like DenseNet-161 (Huang et al., 2017). Khairuddin and Chen (2021) applied extensive data augmentation and optimization techniques

to enhance the performance of the VGGNet architecture. Mejia-Escobar et al. (2023) utilized customized CNN architectures and transfer learning models, improving generalization capabilities. Holder and Tapamo (2018) proposed a novel algorithm combining GLTP and SVM for crowd emotion estimation using the newly created Crowd Emotion dataset. Chowdary et al. (2023) implemented transfer learning with models like ResNet50 and MobileNet (Howard et al., 2018), achieving high accuracies. Nan et al. (2022) introduced a GAN-based feature level super-resolution method to enhance FER for low-resolution images. Gautam and Seeja (2023) applied handcrafted feature extraction techniques like HOG and SIFT with CNNs to improve emotion recognition accuracy. Minaee et al. (2021) employed an attentional conventional network with a spatial transformer to focus on salient facial regions, improving accuracy.

2.4 Performance Evaluation

Performance metrics varied significantly across the studies, with accuracy rates influenced by dataset characteristics and model architectures. Z. Hu (2023) reported higher accuracy for happiness and surprise but lower for fear due to dataset imbalances using the JAFFE dataset. Puthanidam and Moh (2018) achieved 100% accuracy on JAFFE, 89.58% on KDEF, and 71.975% on a combined dataset including JAFFE, KDEF, and SFEW. Appasaheb Borgalli and Surve (2022) achieved near-perfect accuracy on the KDEF dataset but lower accuracy on non-lab-controlled datasets such as SFEW and AMFED+. Santoso and Kusuma (2022) reported a single-model accuracy of 74.45% for VGGSpinalNet on the FER2013 dataset. Khanzada et al. (2020) achieved 75.8% accuracy on FER2013 through transfer learning and data augmentation. Bhagat et al. (2024) achieved 82% training accuracy on FER2013 for real-time FER applications. The various works acknowledged limitations such as overfitting (Z. Hu, 2023; K. Liu et al., 2016; Minaee et al., 2021), class imbalances (Z. Hu, 2023; Khanzada

et al., 2020), and computational resource demands (Z. Hu, 2023; K. Liu et al., 2016; Minaee et al., 2021).

2.5 Crowd and Public Image Testing

Appasaheb Borgalli and Surve (2022) indicated that the test images included those from crowds and public events, emphasizing the diversity and challenge of real-world scenarios. In the study by Khanzada et al. (2020), a mobile web app dataset was created, consisting of images collected from crowd sources like social media posts and direct requests, to train the models for real-time application in natural conditions. The article by Mejia-Escobar et al. (2023) included crowd test images from public events and locations, enhancing the real-world applicability of the models. Nan et al. (2022) presented a novel approach to address the challenges in FER for low-resolution images commonly encountered in crowd scenes such as stations or classrooms. In their investigation, Minaee et al. (2021) leveraged test images from public events and varied crowd settings.

2.6 Challenges

A recurring challenge noted across multiple studies is the need for large, diverse datasets to improve model generalization and performance. For instance, Z. Hu (2023) and Puthanidam and Moh (2018) emphasize the imbalance in datasets like JAFFE, which affects the recognition of certain emotions such as fear. Similarly, Khanzada et al. (2020) and Mejia-Escobar et al. (2023) highlight biases and computational constraints during model training. The issue of dataset quality and generality is also underscored by Holder and Tapamo (2018), who note the challenge of recognizing emotions within non-panic crowd situations. The complexity of preprocessing techniques presents another significant challenge. Many works, in-

cluding those by Appasaheb Borgalli and Surve (2022) and Santoso and Kusuma (2022), discuss the necessity of sophisticated preprocessing steps such as data augmentation, normalization, and image contrast adjustment to enhance image quality and model performance.

The difficulty in managing these preprocessing steps is compounded by the need to handle variations in head pose, lighting, and occlusions, as highlighted by Chowdary et al. (2023) and Gautam and Seeja (2023). Additionally, the study by Nan et al. (2022) illustrates the challenges of preprocessing low-resolution images from crowd scenes using GAN-based feature-level super-resolution methods. Algorithm performance, particularly concerning the use of CNNs and their architectures, is another area fraught with challenges. Overfitting remains a persistent issue, especially in complex models, as discussed by K. Liu et al. (2016) and Santoso and Kusuma (2022). The need for extensive computational resources is a common theme, with works like those by Akhand et al. (2021) and Minaee et al. (2021) emphasizing the high demands of training deep CNNs and the risk of misclassification due to class imbalances. Moreover, the performance variability across different datasets, as noted in the papers by Appasaheb Borgalli and Surve (2022) and Khanzada et al. (2020), highlights the difficulties in achieving consistent accuracy, particularly in non-lab-controlled environments.

Real-world application constraints, including the use of crowd-sourced images and real-time FER, pose additional challenges. Bhagat et al. (2024) and Holder and Tapamo (2018) note the ethical concerns and variability in image quality when deploying emotion recognition technology in live video feeds and public sources. The challenge of integrating FER models into mobile and web applications, as discussed by Khanzada et al. (2020), underscores the need for robust real-time processing capabilities. Moreover, the limitations of exist-

ing models in handling real-world scenarios are highlighted in several studies, such as the inability to generalize across diverse environments and the necessity for more robust datasets and model architectures, as suggested by Khairuddin and Chen (2021) and Nan et al. (2022).

2.7 Suggested Future Research Directions

Enhancing dataset diversity and robustness through advanced data augmentation is a common recommendation, noted by Z. Hu (2023), Khanzada et al. (2020), Mejia-Escobar et al. (2023), and Puthanidam and Moh (2018). Exploring integrated strategies and more sophisticated models like deep belief networks (Hinton et al., 2006), generative adversarial networks, and transformer-based architectures is advocated by Appasaheb Borgalli and Surve (2022) and Mejia-Escobar et al. (2023). Additionally, Akhand et al. (2021), Chowdary et al. (2023), Khairuddin and Chen (2021), and K. Liu et al. (2016) suggested optimizing hyperparameters and fine-tuning model architectures to handle diverse real-world scenarios. The integration of synthetic and real datasets during training, as proposed by Mejia-Escobar et al. (2023), and refining feature-level super-resolution techniques to improve low-resolution FER, as advanced by Nan et al. (2022), are also pivotal. Holder and Tapamo (2018) and Minaee et al. (2021) emphasize enhancing the robustness and accuracy of algorithms through improved GLTP methods, 3D facial tracking setups, and sophisticated attention mechanisms. Real-time application optimization and broader applications of FER technology are recommended by Bhagat et al. (2024) and Minaee et al. (2021). Validating models on more challenging datasets to enhance generalizability is a shared suggestion by Chowdary et al. (2023) and Gautam and Seeja (2023). These future directions underscore the importance of diverse datasets, advanced augmentation, architectural innovations, and real-world application validations in advancing FER.

2.8 Summary of Literature Review

The researchers in the reviewed articles have made significant strides in advancing the FER field, demonstrating a deep understanding of the complexities and challenges inherent in this domain. Their work collectively underscores the critical role of diverse and robust datasets, sophisticated preprocessing techniques, and advanced deep learning models in improving FER accuracy. The innovative approaches, such as the use of transfer learning, ensemble methods, and attention mechanisms, reflect the ongoing efforts to push the boundaries of FER capabilities. The emphasis on addressing real-world applicability, particularly through the inclusion of crowd and public images, highlights a commendable focus on practical implementations.

However, despite these advancements, there are noticeable gaps and challenges that persist. Many studies still rely heavily on a limited set of well-known datasets, which can introduce biases and limit the generalizability of their findings. The complexity and computational demands of preprocessing techniques and deep learning models present significant barriers to real-time application, an area that needs more attention. Furthermore, issues like overfitting, class imbalances, and the ethical implications of FER technology are not always thoroughly addressed. Future research should aim to diversify datasets, streamline preprocessing and model training processes, and ensure robust validation across varied real-world conditions. Addressing these challenges will be crucial in translating the impressive theoretical advancements into practical, scalable FER solutions.

Chapter 3: Methodology

3.1 Data Collection and Preprocessing

3.1.1 Dataset

In the exploration of datasets suitable for use with FER, a crucial aspect of the methodology involves a rigorous selection process. The accessibility of datasets remains a significant concern, as many are not openly available for research purposes. Among the available options, the Karolinska Directed Emotional Faces (KDEF) dataset was chosen for this study.

The KDEF dataset is explicitly designed for facial expression, particularly in psychological and medical research (Puthanidam & Moh, 2018), offering a robust framework for this work's requirements. The images in the KDEF dataset have good resolution and exhibit consistent quality in terms of lighting and alignment, ensuring uniformity across the samples. The dataset includes an equal representation of the seven basic expressions, with each subject photographed expressing each emotion twice. This feature of the KDEF dataset facilitates effective augmentation techniques on original images, allowing for subsequent resizing to meet the specifications of various deep neural network models employed in this research. Thus, the KDEF dataset provides a solid foundation for developing and training accurate and reliable FER systems that is crucial for advancing psychological and cognitive studies.

3.1.2 KDEF Preprocessing

The KDEF dataset consists of 4,900 photos of 70 amateur actors (35 males and 35 females), each displaying seven distinct emotional expressions: neutral, happy, angry, afraid, disgusted, sad, and surprised. Each subject was photographed twice for these expressions, from five unique angles, including straight on, 45 degrees and 90 degrees to the left and right

(see Figure 6). The participants, aged between 20 and 30, were carefully selected to exclude any facial obstructions like beards, mustaches, or glasses. Each photography session was rigorously controlled for consistent lighting and camera positioning, with all images standardized to 562×762 pixels in the JPEG format (Lundqvist, 1998).

Figure 6

Sample of KDEF



In the preprocessing phase of the KDEF dataset for this study, several diligent steps were undertaken to refine the dataset for optimal use in training deep learning models for FER.

Further organization involved categorizing these images into seven subfolders. Each subfolder was dedicated to one of the aforementioned expressions, adhering to the naming conventions specified by the KDEF website (Lundqvist, 1998). This careful categorization aids in systematic training and validation processes by ensuring that each expression is equally represented and easily accessible.

Subsequent preprocessing steps involved cropping the images to a 1:1 aspect ratio (see

Figure 7) using the MediaPipe library from Google², which utilizes three sophisticated facial detection technologies: the single-shot detector (SSD) (W. Liu et al., 2016), MobileNetV2 backbone (Sandler et al., 2018), and BlazeFace model (Bazarevsky et al., 2019).

Figure 7

Crop 1:1 of Facial Detection Using the MediaPipe Library



The SSD framework enables efficient object detection in a single pass, accurately identifying facial boundaries without needing a separate region proposal network. It integrates as its backbone MobileNetV2, which is designed for mobile and edge devices, using an inverted residual structure to reduce computational load while maintaining detection robustness.

BlazeFace, another component, specializes in rapid and precise face detection on mobile devices, handling diverse orientations and lighting conditions with minimal computational resources. Together, these technologies optimize datasets by focusing strictly on facial features, enhancing deep learning effectiveness.

To prepare the dataset for training, the images were randomly divided into two groups,

² <https://ai.google.dev/edge/mediapipe/solutions/guide>

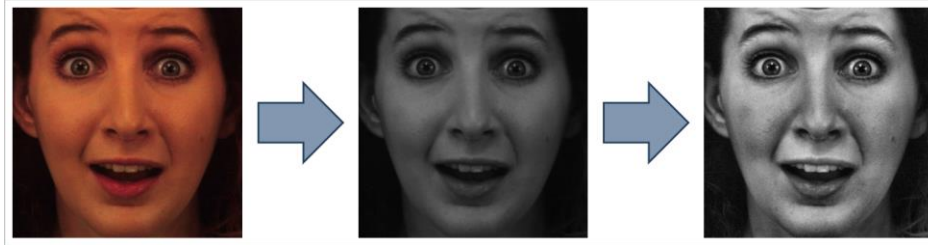
with 20% designated for validation and the remaining 80% reserved for training. This split facilitates the evaluation of the model's performance on unseen data, a crucial step in verifying the generalizability of the trained model. Moreover, all images were converted to grayscale to eliminate color as a variable, which, although visually informative, does not influence facial expressions and introduces unnecessary complexity into the feature set.

Enhancements were further applied through histogram equalization using the contrast-limited adaptive histogram equalization (CLAHE) method (Pizer et al., 1987), a technique used to improve contrast in the images. CLAHE differs from ordinary histogram equalization by its capability to reduce noise amplification, which is particularly useful in homogeneous areas (Sonali et al., 2019). It does this by applying histogram equalization locally to small regions of the image, known as tiles, rather than globally across the entire image. For this dataset, the parameters were set with a clipLimit of 2.0 and a tileGridSize of (8,8) (see Figure 8). The clipLimit parameter controls the contrast limit for localized changes; setting it to 2.0 helps avoid overly contrasting changes that could distort natural facial expressions (Majeed & Isa, 2020). The tileGridSize parameter defines the size of the tile in which the histogram equalization is applied, with each tile being 8x8 pixels.

This size is chosen to ensure that the equalization process maintains enough detail without creating artifacts or over-enhancing the contrast in smaller features. This approach is particularly beneficial for enhancing subtle expressions in images, making it easier for models to detect and learn from these subtle details.

Figure 8

Histogram Equalizer Result by CLAHE (Right Image)



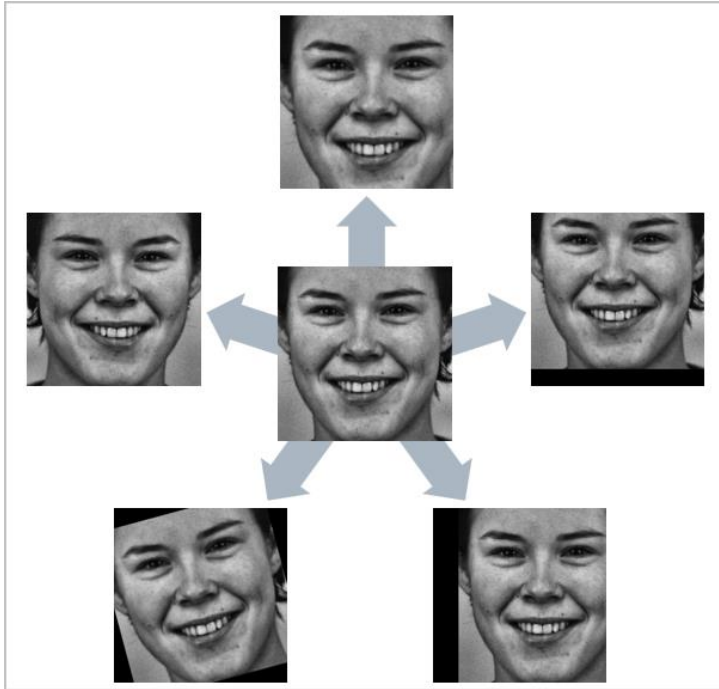
To expand the dataset, the number of images was increased six-fold from 4,900 to 29,400 through the application of five specific augmentation techniques: rotation, width shifting, height shifting, zooming, and horizontal flipping. These techniques were selected for their effectiveness in creating variation while preserving the integrity of the facial expressions, thus avoiding augmentations that could introduce noise and degrade the image quality (see Figure 9).

Finally, the augmented images were resized to three different resolutions— 64×64 , 128×128 , and 224×224 pixels—to accommodate various input specifications of the deep learning models. This resizing ensures that the dataset can be utilized across different network architectures, providing flexibility in model training and experimentation. These image sizes represent the boundaries of face sizes in regular images of events.

Through these detailed and carefully executed preprocessing steps, the dataset was significantly enhanced, not only in terms of quantity but also in quality, providing a robust foundation for training highly accurate and reliable FER models.

Figure 9

Augmentation of One Image (Center) to Five More Images Without Affecting the Quality



3.1.3 COP28 Data Collection and Preprocessing

The collection and preprocessing of test data for this study involved a careful process, primarily utilizing video footage from the official COP28 YouTube channel³. Initially, frames were extracted from the videos at specific intervals to ensure a comprehensive sampling of expressions and contexts. For longer videos, frames were extracted every 5 seconds, while shorter clips underwent a more frequent extraction rate of every 1 second. This strategy was employed to maximize the diversity of facial expressions captured while maintaining a manageable dataset size. From a total of 21 hours of video content, this method yielded an initial

³ <https://www.youtube.com/@COP28UAEOfficial>

collection of 18,700 images.

To refine these images further, the MediaPipe library was utilized in the same way as the training dataset. Despite the high accuracy of the MediaPipe library, manual intervention was necessary to ensure the utmost precision in the dataset. This manual refinement process involved several steps: correcting minor errors that occurred during automated facial detection, excluding images where faces were obscured (e.g., by hands or objects), and removing duplicate images of the same individuals from extended scenes. This careful curation was important to avoid biases in facial recognition due to repeated or unclear data, resulting in a refined testing dataset comprising 1,111 images.

Subsequently, these images were manually labeled into seven expressions: afraid, angry, disgusted, happy, neutral, sad, and surprised, consistent with the classification used in the training dataset. This step ensured that the test dataset mirrored the categorical structure of the training data, facilitating a valid assessment of the model's performance across similar expression categories.

Finally, the test images underwent the same preparation techniques as those applied to the training dataset, with one exception: augmentation. Unlike the training data, where augmentation techniques are employed to artificially expand the dataset and introduce variability, such techniques were omitted for the test data. In testing, it is crucial to evaluate the model's performance on unaltered, real-world data to accurately assess its generalization capabilities and real-time effectiveness (see Figure 10). This approach ensures that the testing conditions

realistically mimic the scenarios in which the FER model will be deployed, providing a genuine measure of its predictive accuracy and robustness. Note that in the study document, the anonymization process involves selectively covering the person's eyes with a gray rectangular overlay, ensuring their identity remains protected and compliant with research ethics. However, during image processing for prediction, the eyes were uncovered.

Figure 10

Samples of COP28 Test Data Preprocessing



3.2 The Deep Learning Models for FER Examined in This Study

The models examined in this study are categorized into two main groups: CNN-based models, which include standard CNN, expanded CNN, multi-path CNN-MassAtt-PWFS, and learning transfer VGG16, and non-CNN models, which include the RNN, LSTM, GRU, and vision transformer (ViT) models.

3.2.1 The Standard CNN Model

This model consists of three convolutional layers followed by two dense (linear) layers. This structure allows the model to learn from the raw pixel data, capturing essential features at various levels of abstraction. The Figure 11 illustrates the complete design for 128×128 input images.

Figure 11

The Design of the Standard CNN Model for 128×128 Input Images

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 32, 126, 126]	320
ReLU-2	[-1, 32, 126, 126]	0
MaxPool2d-3	[-1, 32, 63, 63]	0
Conv2d-4	[-1, 64, 61, 61]	18,496
ReLU-5	[-1, 64, 61, 61]	0
MaxPool2d-6	[-1, 64, 30, 30]	0
Conv2d-7	[-1, 64, 28, 28]	36,928
ReLU-8	[-1, 64, 28, 28]	0
MaxPool2d-9	[-1, 64, 14, 14]	0
Dropout-10	[-1, 64, 14, 14]	0
Flatten-11	[-1, 12544]	0
Linear-12	[-1, 128]	1,606,720
ReLU-13	[-1, 128]	0
Linear-14	[-1, 7]	903
Softmax-15	[-1, 7]	0
=====		
Total params: 1,662,407		
Trainable params: 1,662,407		
Non-trainable params: 0		

3.2.1.1 The Structure of the Standard CNN Model. The standard CNN model utilizes the powerful feature extraction capabilities of convolutional layers. It is constructed using PyTorch⁴, which is a popular library for developing and training deep learning models (Ansel et al., 2024).

Before defining the model, the input data are initialized to rescale the pixel values of images from a grayscale range of 0 to 255 to a normalized range of -1 to 1. This normalization step is necessary for ensuring that the neural network trains efficiently by standardizing the input data, which helps in speeding up the convergence process.

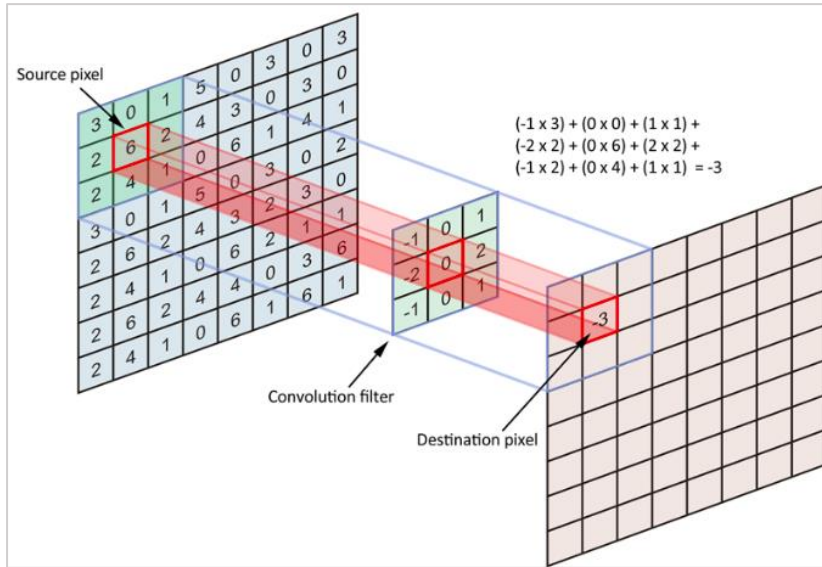
⁴ <https://pytorch.org/>

The data is then organized into training and testing datasets using the ImageFolder class, which structures the image data into a directory-based format. Data loaders are created to efficiently manage the flow of data during training and testing, with a batch size of 32 and shuffling applied to the training data to ensure stochasticity.

The input to the model is defined by the first convolutional layer, which expects images 64×64 , 128×128 , and 224×224 pixels in size with a single channel (grayscale images). This layer applies 32 filters to the input image, each 3×3 in size (see Figure 12).

Figure 12

3×3 Kernel on Input CNN Layer (Rachapudi & Lavanya Devi, 2021)



A filter in a convolutional layer is essentially a small matrix that slides over the input image, performing element-wise multiplications and summing up the results to produce a feature map (Mathew et al., 2021). In this context, 32 filters mean that the layer uses 32 different

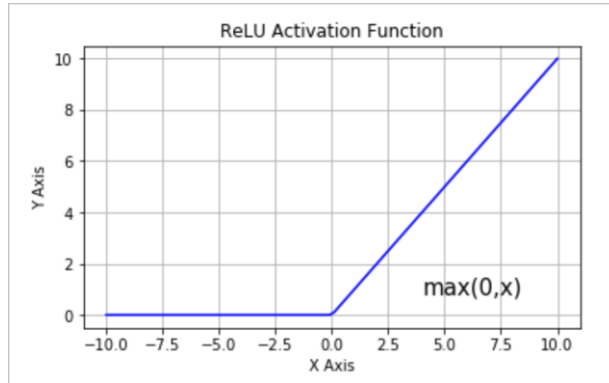
3×3 kernels to scan the image, capturing various features such as edges, textures, and patterns. The 3×3 size refers to the dimensions of each filter, indicating that they consider a 3×3 pixel region of the image at a time.

The stride, which is the number of pixels by which the kernel moves across the image (Khorsheed, 2022), is set to its default value of 1. This means that the kernel shifts one pixel at a time while scanning the image, allowing for detailed feature extraction by examining each possible 3×3 region of the image.

The rectified linear unit (ReLU) activation function is used to introduce non-linearity into the model. The ReLU transforms each input value x to $\max(0, x)$, effectively setting all negative values to zero and keeping positive values unchanged (Pal & Saraswat, 2018). This non-linear operation enables the model to learn and represent complex patterns in the data. Without the ReLU or another non-linear activation function, the model would not be able to capture intricate structures, as the entire network would behave like a single linear transformation regardless of its depth (see Figure 13).

Figure 13

ReLU Function (Pal & Saraswat, 2018)

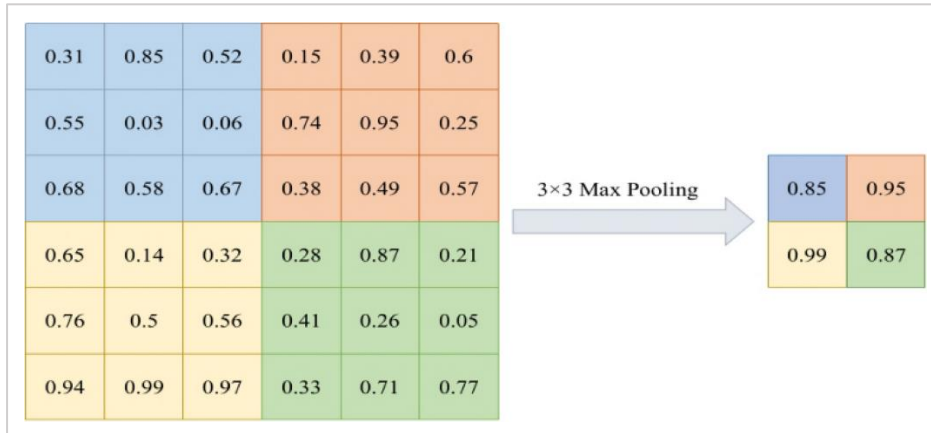


Padding is not applied, meaning the output dimensions will be reduced compared to the input dimensions due to the convolution operation. Padding refers to the process of adding extra pixels around the edges of the input image before applying the convolution operation (Hashemi, 2019).

Following the first convolutional layer, a max pooling layer is used to reduce the spatial dimensions of the feature maps by taking the maximum value in each 2×2 window. Max pooling operates by sliding a 2×2 filter over the input feature map and selecting the maximum value within each region covered by the filter. This process results in a downsampled feature map that retains the most salient features while discarding less significant information (Zhao & Zhang, 2024). This reduction helps in decreasing the computational complexity and mitigating the risk of overfitting by reducing the number of parameters and the spatial size of the feature maps, thereby making the model more efficient and less prone to learning noise from the training data (see Figure 14).

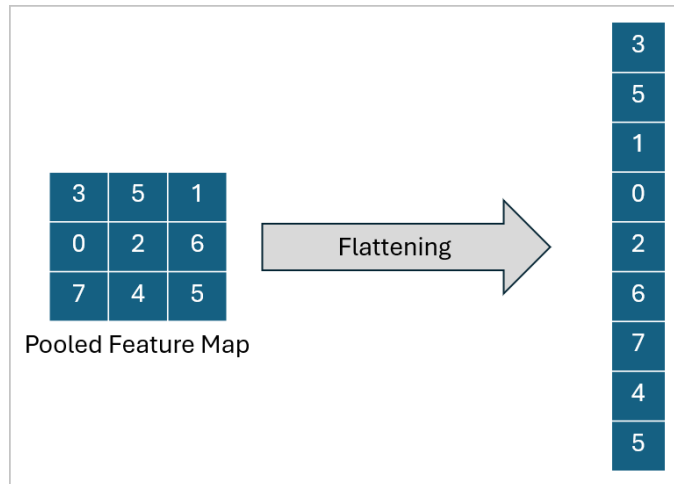
Figure 14

Example of 3x3 Max Pooling (Zhao & Zhang, 2024)



A dropout layer with a dropout rate of 25% is added after the third max-pooling layer. Dropout is a regularization technique used to prevent overfitting by randomly setting a fraction (25% in this case) of input units to zero at each update during training time. This means that during each training iteration, 25% of the neurons in this layer are temporarily ignored, forcing the model to learn redundant representations of the data. This helps ensure that the model does not become overly reliant on any particular set of features, thereby improving its ability to generalize well to new, unseen data (Srivastava et al., 2014). By preventing the co-adaptation of neurons, dropout encourages the network to develop a more robust and diverse set of features, enhancing the overall performance and stability of the model.

The fully connected part of the model begins with a flattening layer that converts the 2D feature map output by the last convolutional layer into 1D feature vectors (see Figure 15).

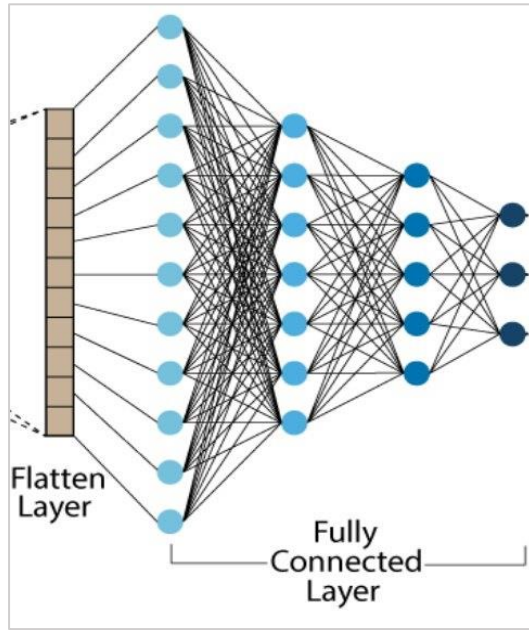
Figure 15*2D to 1D Flattening*

This transformation is crucial as it prepares the data for the dense layers (linear layers as named by PyTorch) by collapsing the spatial dimensions of the feature maps into a single vector. The flattening process allows the subsequent dense layers to process the data in a format suitable for classification (Wang et al., 2021).

The first dense layer consists of 128 neurons and uses the ReLU activation function. Dense layers, also known as fully connected layers (see Figure 16), are fundamental in neural networks as they connect every neuron in the previous layer to every neuron in the current layer.

Figure 16

Example of Fully Connected Dense Layers (Rguibi et al., 2022)



Finally, the output layer consists of seven neurons, corresponding to the seven categories of facial expressions (afraid, angry, disgusted, happy, neutral, sad, and surprised). The softmax activation function is employed in the output layer to produce a probability distribution over the seven classes. Softmax works by converting the raw output scores from the neurons into probabilities that sum to one, effectively normalizing the output (Mathew et al., 2021). This transformation is essential for multi-class classification tasks, as it allows the model to output a likelihood for each class, enabling the identification of the most probable facial expression.

3.2.2 The Expanded CNN Model

This model includes eight convolutional layers that allow for deeper feature extraction

and three dense (linear) layers. This model is designed to handle more complex variations in facial expressions by analyzing a broader context within the image data (see Figure 17 for the complete design).

3.2.2.1 The Structure of the Expanded CNN Model. The model's architecture begins with a sequence of convolutional layers. The initial layer, conv1, consists of 16 filters with a kernel size of 3x3, and applies padding to maintain the input dimensions. This is followed by a batch normalization layer (bn1) that normalizes the output of the convolutional layer, thereby accelerating the training process and improving model stability (Ogundokun et al., 2022). The output is then passed through a ReLU activation function and a max-pooling layer that reduces the spatial dimensions by half.

The second convolutional block repeats this structure with another 16 filters followed by a similar activation and pooling process. This pattern continues with increasing complexity in subsequent layers. The third convolutional block consists of 32 filters followed by another max-pooling operation. This design progressively captures more complex and abstract features as the network depth increases. The fourth and fifth convolutional blocks each contain 32 filters, further enhancing the model's ability to learn intricate patterns within the facial images.

Figure 17

The Design of the Expanded CNN Model for 128×128 Input Images

Layer (type)	Output Shape	Param #
=====		
Conv2d-1	[-1, 16, 128, 128]	160
BatchNorm2d-2	[-1, 16, 128, 128]	32
MaxPool2d-3	[-1, 16, 64, 64]	0
Conv2d-4	[-1, 16, 64, 64]	2,320
BatchNorm2d-5	[-1, 16, 64, 64]	32
Conv2d-6	[-1, 32, 64, 64]	4,640
BatchNorm2d-7	[-1, 32, 64, 64]	64
MaxPool2d-8	[-1, 32, 32, 32]	0
Conv2d-9	[-1, 32, 32, 32]	9,248
BatchNorm2d-10	[-1, 32, 32, 32]	64
MaxPool2d-11	[-1, 32, 16, 16]	0
Conv2d-12	[-1, 32, 16, 16]	9,248
BatchNorm2d-13	[-1, 32, 16, 16]	64
MaxPool2d-14	[-1, 32, 8, 8]	0
Conv2d-15	[-1, 64, 8, 8]	18,496
BatchNorm2d-16	[-1, 64, 8, 8]	128
MaxPool2d-17	[-1, 64, 4, 4]	0
Conv2d-18	[-1, 64, 4, 4]	36,928
BatchNorm2d-19	[-1, 64, 4, 4]	128
Conv2d-20	[-1, 64, 4, 4]	36,928
BatchNorm2d-21	[-1, 64, 4, 4]	128
Flatten-22	[-1, 1024]	0
Linear-23	[-1, 256]	262,400
BatchNorm1d-24	[-1, 256]	512
Dropout-25	[-1, 256]	0
Linear-26	[-1, 128]	32,896
BatchNorm1d-27	[-1, 128]	256
Dropout-28	[-1, 128]	0
Linear-29	[-1, 7]	903
=====		
Total params: 415,575		
Trainable params: 415,575		
Non-trainable params: 0		
=====		

After the convolutional layers, the model includes a flattened layer. The first fully connected layer reduces the dimensionality to 256 neurons, followed by a dropout with a 50% dropout rate.

The final fully connected layer outputs a vector of size 7, corresponding to the seven

facial expression categories. This output is then passed through a softmax activation function.

3.2.2.2 Training Process of the Standard and Expanded CNN Models. The process begins with the definition of the optimizer and the loss function. The criterion used is the cross-entropy loss, a standard loss function for classification tasks that calculates the difference between the predicted probabilities and the actual labels. The optimizer selected is adaptive moment estimation (Adam), which is known for its efficiency in handling sparse gradients on noisy problems (Kingma & Ba, 2014). It is initialized with a learning rate of 0.00005, a conservative choice to ensure stable convergence.

Metric calculation functions are defined to evaluate the model's performance beyond simple accuracy. These metrics include precision, recall, and the F1 score, all calculated using a weighted average to account for any class imbalances. The precision metric measures the proportion of true positive predictions among all positive predictions, recall measures the proportion of true positive predictions among all actual positives, and the F1 score provides a harmonic mean of precision and recall. Additionally, overall accuracy is computed as the mean of correct predictions.

The training loop spans several epochs, during which the model iteratively learns from the training data and is evaluated on the validation data. The loop begins by setting the model to the training mode and initializing variables to track the running loss and store all predictions and labels for the current epoch. The training data are loaded in batches using a data loader.

For each batch, images and labels are transferred to the computing device (CPU or GPU), and the model's gradients are reset using `optimizer.zero_grad()`. The forward pass is

performed by feeding the images through the model to obtain predictions, which are then compared to the actual labels using the loss function. The computed loss is backpropagated through the network using `loss.backward()`, and the optimizer updates the model parameters to minimize the loss. The running loss is accumulated for reporting purposes, and predictions are collected for metric calculations.

After completing the iterations over the training data, the average loss for the epoch is computed, and custom metrics (precision, recall, F1 score, and accuracy) are calculated using the collected predictions and labels. These metrics are recorded for further analysis.

The model is then switched to evaluation mode to validate its performance on the test data. In this mode, the model's parameters are not updated. Instead, the test data is processed similarly to the training data, but with no gradient calculations to save memory and computations. The validation loss and metrics are computed and recorded.

Throughout the training process, the best model parameters are saved based on the validation loss and accuracy. If the current epoch's validation loss is lower than the best recorded loss, the model's state is saved to a file for future use. Similarly, if the current epoch's accuracy surpasses the best recorded accuracy, the model's state is saved separately.

This comprehensive training process ensures that the model is not only learning effectively but also being rigorously evaluated to prevent overfitting and ensure generalization. The inclusion of custom metrics provides a deeper insight into the model's performance across different aspects of classification, making the evaluation more robust and informative.

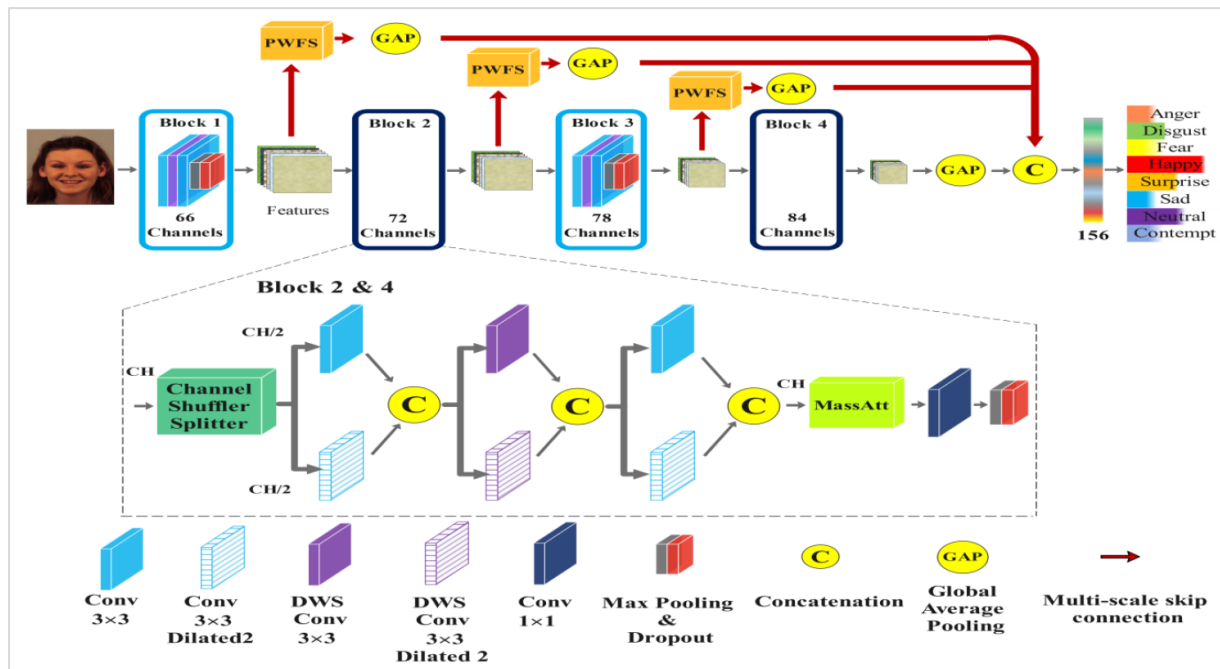
3.2.3 The Multi-Path CNN-MassAtt-PWFS Model

This model represents a significant escalation in complexity, featuring 27 hidden layers distributed among three intertwined models, based on the work of Ezati et al. (2024). It includes 23 convolutional layers and four dense layers, employing a multi-path architecture to focus attention more effectively across the input space. This design is aimed at enhancing the model's sensitivity to subtle emotional cues by leveraging massive attention (MassAtt) mechanisms and point-wise feature selection (PWFS).

Figure 18 shows the design of the multi-path CNN model, incorporating all the mentioned layers (see Appendix A for the total model parameters).

Figure 18

Multi-Path CNN (Lightweight Attentional Network Incorporating Multi-Scale Feature Fusion) Model (Ezati et al., 2024)



3.2.3.1 The Structure of the Multi-Path Model. At the heart of the model is the two-path convolutional module, which enhances the feature extraction process by dividing the input tensor into two separate pathways. The input tensor is first shuffled and split into two groups of channels, each processed independently. This division allows the model to capture a broader range of features by leveraging different convolutional strategies. The first pathway employs standard convolutional operations, which are effective for capturing fine-grained local features. In contrast, the second pathway uses dilated convolutions, which expand the receptive field without increasing the computational burden, allowing the network to capture more contextual information from the input images. Both pathways include batch normalization and ReLU activation, which help stabilize the training process and introduce non-linearity, respectively. The outputs of these two pathways are then concatenated, merging local and contextual features into a single, rich representation.

3.2.3.1.1 Mass Attention (MassAtt) Module. The MassAtt module is a critical component of the model, designed to recalibrate feature maps by focusing on the most important aspects of the data. This module generates both channel and spatial attention maps. The channel attention mechanism begins with global average pooling, which reduces each feature map to a single value representing the average activation (Hsiao et al., 2019). This pooled representation is passed through a dense network with a reduction ratio, typically set to 4, to learn the importance of each channel. The network produces a channel attention map that scales the original feature maps, emphasizing significant channels while suppressing less important ones.

Simultaneously, the spatial attention mechanism processes the input tensor to produce a spatial attention map. This involves reducing the feature maps along the channel dimension

to generate a single-channel spatial representation, followed by convolutional and transposed convolutional operations to highlight critical spatial regions. The final spatial attention map is combined with the channel attention map to produce a comprehensive attention map that enhances the most informative features across both dimensions.

3.2.3.1.2 Point-Wise Feature Selection (PWFS) Module. The PWFS module further refines the feature selection process by employing a comparative approach. This module splits the input feature maps into two sub-groups and performs a channel-wise comparison to retain the most meaningful features. By computing the minimum, maximum, and median values across the channels, the PWFS module effectively filters out noise and emphasizes important features. The result is a set of average values that represent the most relevant information, which is then used in subsequent layers of the network.

3.2.3.1.3 Convolutional Blocks. The architecture of the model includes four main convolutional blocks, each designed to extract and refine features from the input images progressively:

First Block: The first block employs two types of convolutional layers: standard Conv2D with 32 filters 3×3 in size and SeparableConv2D also with 32 filters 3×3 in size. The use of separable convolutions helps to reduce the model's parameter count and computational complexity, followed with batch normalization, then the ReLU activation function. The block concludes with a MaxPooling2D layer with a pool size of 2×2 , and a dropout layer at 40%.

Second, Third, and Fourth Blocks: These blocks follow a similar structure but with

an increase in the number of filters, enhancing the model's ability to capture more detailed features as the depth increases. Each block integrates the MassAtt module. The blocks conclude with a PWFS layer. Global Average Pooling 2D is applied at the end of each block to reduce data dimensionality. The total count of convolutional hidden layers in the model amounts to 23, including both Conv2D and SeparableConv2D layers, making the model both deep and capable of intricate feature extraction. The model incorporates four dense hidden layers within its architecture. Each dense layer is followed by a dropout at 50%. The final layer of the model uses a softmax activation function with seven units.

By integrating these advanced architectural elements and techniques, the multi-path CNN model achieves a high level of performance in FER, making it a powerful tool for applications requiring accurate and efficient FER.

3.2.3.2 Training Process of the Multi-Path CNN Model. The input images of 64x64 are stored in NumPy arrays, and the labels are mapped to numerical values representing the seven facial expression categories. The learning rate in the Adam optimizer is set to 0.001, which controls the step size for each iteration as the model moves toward minimizing the loss function. The Adam optimizer also uses two parameters, β_1 and β_2 , to adjust how the model learns over time.

The β_1 parameter is set at 0.9 and controls the exponential decay rate for the moving average of past gradients. Essentially, this means that the optimizer calculates a running average of the gradients, giving more importance to the most recent while still considering the past

ones. This helps to stabilize the updates by smoothing out the changes in the gradients, reducing oscillations during the training process.

The β_2 parameter is set at 0.999 and controls the exponential decay rate for the moving average of past squared gradients. This parameter helps to smooth out the updates even further by considering the variance of the gradients. By giving more weight to past squared gradients, the optimizer can adjust the learning rate more effectively, ensuring that the updates are neither too large nor too small, which helps in achieving a smoother and more stable convergence. An epsilon value of 10^{-7} is included in the Adam optimizer to avoid dividing by zero, preventing extremely large updates that could destabilize the training process.

The evaluation metrics applied, such as precision, recall, F1 score, and accuracy, are the same as those used for previous CNN models.

Several callbacks are employed to optimize and monitor the training process. The Model Checkpoint callback saves the model weights whenever there is an improvement in validation accuracy, ensuring that the best-performing model is preserved. The ReduceLROnPlateau callback monitors the validation loss and reduces the learning rate by a factor of 0.1 if no improvement is observed for nine consecutive epochs. This adaptive learning rate strategy helps refine the model's weights more effectively in the later stages of training, preventing the model from getting stuck in local minima and enhancing overall optimization.

Throughout the training, the callbacks ensure that the learning rate is adapted appropriately, the best model weights are saved, and the training progress is visually tracked. This combination of strategies optimizes the training process, reducing computational demands

while maintaining a high performance in FER tasks.

3.2.4 The Transfer Learning VGG16 Model

The VGG16 model, based on the work of Akhand et al. (2021), is a deep convolutional neural network that has been pre-trained on the extensive ImageNet dataset, encompassing millions of images across a diverse range of categories. This extensive pre-training equips VGG16 with highly effective and generalizable features for various image recognition tasks, making it a prominent choice in the field of computer vision. By leveraging the power of these pre-trained features, VGG16 offers a solid foundation for fine-tuning on specific tasks such as FER, where the nuances and subtleties of facial features are critical for accurate classification.

VGG16 model features 16 layers, including 13 convolutional and three dense layers. For the purpose of FER, this model was adapted by replacing its last three dense layers with two ones specifically designed to classify seven distinct facial expressions.

3.2.4.1 The Structure of the Transfer Learning VGG16 Model. The input layer of the VGG16 model accepts images of 224×224 pixels in size with three color channels (RGB), ensuring a standardized input format that aligns with common practices in image recognition tasks. The original top classification layer of VGG16, which is designed for the ImageNet dataset, has been excluded. This modification allows for the integration of custom output layers specific to the task of FER.

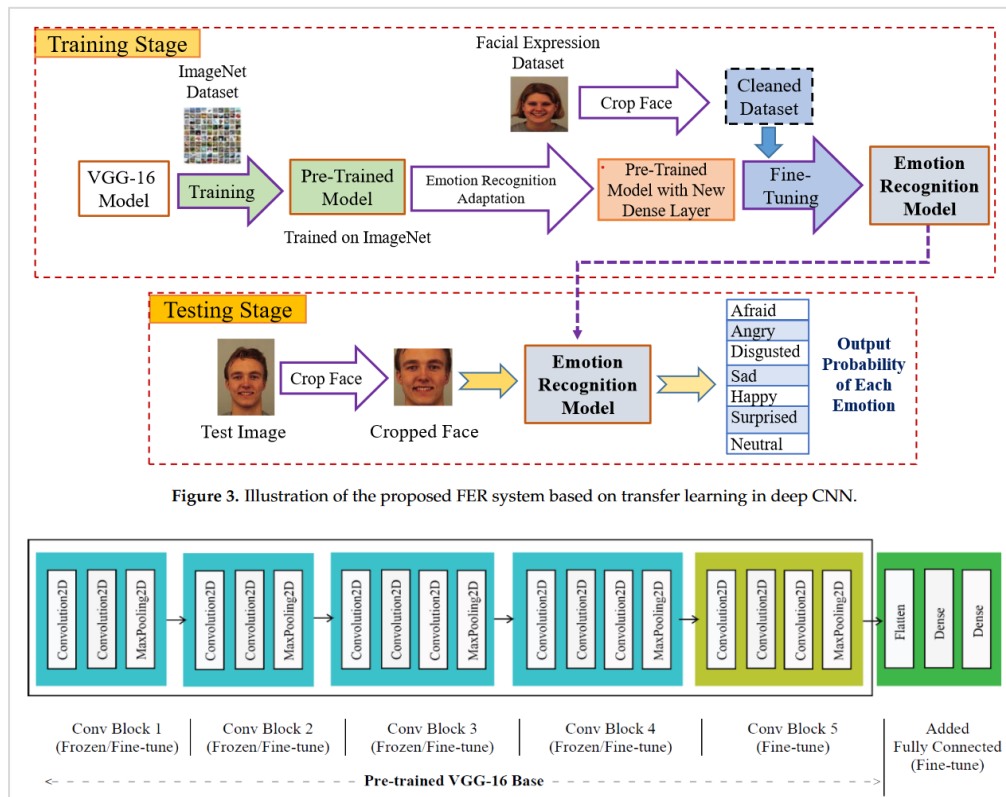
The convolutional base of the model comprises 13 convolutional layers organized into blocks, each followed by a max-pooling layer. These layers utilize 3×3 convolution kernels. The convolutional layers employ ReLU activation functions and max pooling.

Following the convolutional base, the model incorporates custom output layers tailored for FER. The first step in this adaptation is the flattening layer. The first dense layer comprises 512 units and employs ReLU activation. The final output layer consists of seven units with softmax activation. By excluding the original top classification layer and integrating custom dense layers, the model's output dimensions are tailored to the FER task.

Figure 19 shows the design of the transfer learning VGG16 model, incorporating all mentioned layers (see Appendix B for the total model parameters).

Figure 19

Transfer Learning VGG16 Model (Akhand et al., 2021)



3.2.4.2 Training Process of the VGG16 Model. For the training process, the Adam optimizer is utilized with a learning rate of 0.00005. To measure the discrepancy between the predicted probabilities and the actual class labels, the categorical cross-entropy loss function is used. The performance of the model is monitored using precision, recall, F1 score, and accuracy. During the training process, the model is trained using batches of images generated by the training generator. Similarly, the validation generator provides batches of images for validation, allowing the model's performance to be assessed on unseen data after each epoch.

The training duration is set for 30 epochs, which provides a sufficient number of iterations for the model to learn from the training data while also allowing for periodic evaluation against the validation data.

3.2.5 The RNN Model

Recurrent neural networks (RNNs), introduced by Williams and Zipser (1989), are a class of artificial neural networks that are particularly well suited for processing sequential data, where the current output depends not only on the current input but also on previous inputs (Das et al., 2023). This characteristic makes RNNs powerful for tasks involving time series, language modeling, and speech recognition, as will be explained in the coming LSTM model.

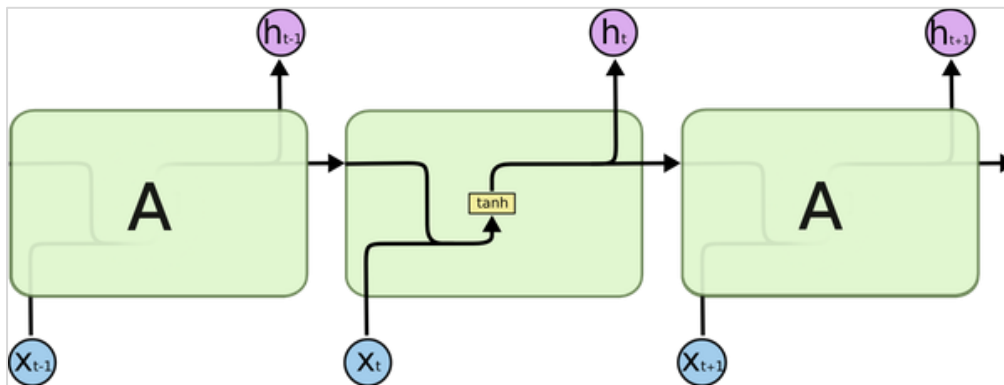
The fundamental structure of an RNN consists of a standard neural network layer that is looped upon itself. This loop allows information to persist, essentially enabling the network to have a 'memory' of previous inputs. At each time step, the hidden state of the RNN is updated based on the current input and the previous hidden state (see Figure 20). This structure

enables the network to process data sequentially, making it well suited for applications where the order of data is crucial (Das et al., 2023).

Despite their theoretical capabilities, RNNs suffer from practical limitations, particularly the vanishing gradient problem. As the gradient of the loss function is backpropagated through many time steps, it tends to either vanish (become very small) or explode (grow exponentially), making training difficult. This issue limits the ability of RNNs to learn long-term dependencies in the data (Noh, 2021).

Figure 20

RNN Structure (Olah, 2015)



3.2.5.1 The Structure of the RNN Model. The preprocessing pipeline begins with several transformations applied to the input images. Random rotations of up to 20 degrees and horizontal flipping are applied to augment the dataset, introducing variability that helps the model generalize better to different facial orientations and expressions. The images are then converted to tensors and normalized to have a mean of 0.5 and a standard deviation of 0.5, to a normalized range of -1 to 1. The dataset is organized using PyTorch's 'ImageFolder' class,

and data loaders are created for both the training and testing datasets with a batch size of 32. Shuffling is applied to the training data to ensure that the model is exposed to a diverse set of images in each epoch, promoting robust learning.

The core of the model is defined in the ‘RNNModel’ class, which extends ‘nn.Module’. The model’s architecture begins with a ‘flatten’ layer, which reshapes the input images. Initially, the images are three-dimensional tensors of shape (batch_size, channels, height, width), which for grayscale images is (32, 1, 128, 128). The flatten layer transforms this into a two-dimensional tensor of shape (batch_size, channels, height * width), resulting in (32, 1, 16384). This transformation makes the data suitable for input into the RNN layers by flattening each image into a sequence of 16,384 features. A second RNN layer follows, processing the output of the first RNN and maintaining the same dimensionality for its output (see Figure 21).

Figure 21

The Design of the RNN Model for 128×128 Input Images

Layer (type)	Output Shape	Param #
RNNModel	[32, 7]	0
└Flatten: 1-1	[32, 1, 16384]	0
└RNN: 1-2	[32, 1, 128]	2,113,792
└RNN: 1-3	[32, 1, 128]	33,024
└Linear: 1-4	[32, 128]	16,512
└Dropout: 1-5	[32, 128]	0
└Linear: 1-6	[32, 7]	903
=====		
Total params: 2,164,231		
Trainable params: 2,164,231		
Non-trainable params: 0		

The final output of the second RNN layer is then passed through two fully connected (dense) layers. The first dense layer reduces the dimensionality to 128 neurons and applies the ReLU activation function. A dropout layer with a dropout rate of 50% is used to prevent overfitting. The final dense layer maps the 128-dimensional input to the seven output classes, corresponding to the seven facial expressions. The model summary indicates that it has a total of 2,164,231 trainable parameters, as shown in Figure 21.

3.2.6 The LSTM Model

Long short-term memory (LSTM) networks are a specialized form of RNNs designed to model temporal sequences and capture long-term dependencies. Introduced by Hochreiter and Schmidhuber (1997), LSTMs address the limitations of traditional RNNs, such as the vanishing gradient problem, making them particularly effective for tasks involving sequential data.

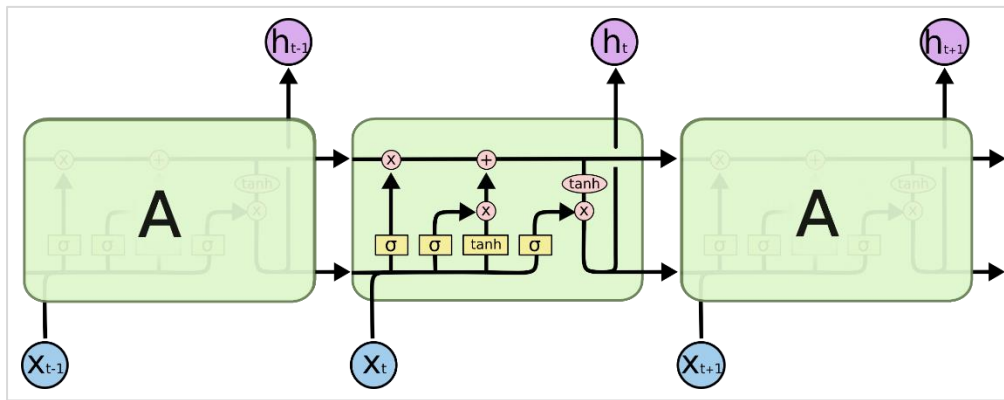
An LSTM network's core component is its memory cell, which maintains information over long periods. This cell is regulated by three gates: the input, forget, and output gates. The input gate determines how much new information should be added to the cell state using a sigmoid function for updates and a tanh function for new candidate values. The forget gate decides which parts of the previous cell state to retain, using a sigmoid function to output values between 0 (forget) and 1 (retain). The output gate controls the cell state's output, filtering it through a sigmoid function and squashing the values between -1 and 1 with a tanh function (Olah, 2015; Ye, 2024) (see Figure 22).

LSTM networks are highly effective for tasks requiring contextual understanding over

extended sequences. They excel in time series forecasting, capturing trends and seasonal patterns for tasks like stock market prediction and weather forecasting (Siami-Namini et al., 2019). In natural language processing, LSTMs are used for language modeling, machine translation, and sentiment analysis, leveraging the context provided by previous words (Yao & Guan, 2018). For speech recognition, LSTMs model the temporal dynamics of speech signals, improving spoken word and phrase recognition (Graves et al., 2013).

Figure 22

LSTM Structure (Olah, 2015)



3.2.6.1 The Structure of the LSTM Model. The structure of the FER LSTM model is not different from the previous RNN model (see Figure 23). However, the LSTM model has a total of 8,604,679 trainable parameters for 128×128 input images, which is four times the number of parameters in the RNN model, giving an advantage to the LSTM model over the RNN one. This change refers to the formula for calculating the parameters in an LSTM. LSTM networks have more parameters compared to standard RNNs because each LSTM cell includes additional gates (input, forget, and output gates) and internal states that require extra weights and biases (see Figure 22).

Figure 23

The Design of the LSTM Model for 128×128 Input Images

Layer (type)	Output Shape	Param #
=====		
LSTMModel	[32, 7]	0
└Flatten: 1-1	[32, 1, 16384]	0
└LSTM: 1-2	[32, 1, 128]	8,455,168
└LSTM: 1-3	[32, 1, 128]	132,096
└Linear: 1-4	[32, 128]	16,512
└Dropout: 1-5	[32, 128]	0
└Linear: 1-6	[32, 7]	903
=====		
Total params: 8,604,679		
Trainable params: 8,604,679		
Non-trainable params: 0		
=====		

The number of trainable parameters can be calculated using the following formula (Olah, 2015; Sak et al., 2014):

$$LSTM\ Parameters = 4 \times ((n_{input} \times n_{units}) + (n_{units}^2) + n_{units})$$

Where:

- n_{input} is the number of input features.
- n_{units} is the number of LSTM units (or neurons) in the layer.

This formula accounts for the weights associated with the input, the recurrent connections (within the LSTM units), and the biases. The factor 4 comes from the four sets of weights and biases required for the input gate, forget gate, cell state, and output gate in each LSTM cell. In contrast, an RNN's parameter count is given by Olah (2015) and Shewalkar (2018):

$$RNN\ Parameters = (n_{input} \times n_{units}) + (n_{units}^2) + n_{units}$$

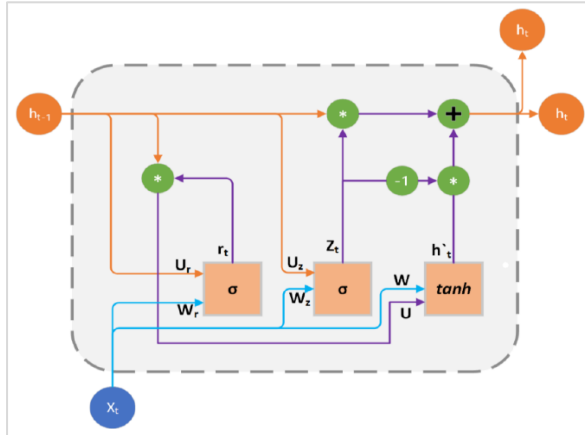
Where:

- n_{input} is the number of input features.
- n_{units} is the number of RNN units (or neurons) in the layer.

3.2.7 The GRU Model

The gated recurrent unit (GRU) neural network, introduced by Cho et al. (2014), addresses the limitations of traditional RNNs, particularly the vanishing and exploding gradient problems, by simplifying the structure of LSTM networks. GRUs combine the functionalities of the input and forget gates into a single update gate, reducing computational complexity while capturing long-term dependencies in sequential data (Shewalkar, 2018).

A GRU unit comprises two main gates: the reset gate, which controls the flow of past information, and the update gate, which determines the extent to which the previous hidden state is carried forward. This architecture efficiently maintains and updates the hidden state using element-wise operations and activation functions, enabling effective learning and adaptation to sequences of varying lengths (see Figure 24).

Figure 24*GRU Structure (Zheng et al., 2024)*

3.2.7.1 The Structure of the GRU Model. The structure of the FER GRU model does not differ from the previous RNN and LSTM models. However, the GRU model has a total of 6,457,863 trainable parameters, which is three times the number of parameters in the RNN model for the same images size (128×128). This increment in parameters compared to standard RNNs is because each GRU cell includes additional gates (reset and update gates) and internal states that require extra weights and biases (Shewalkar, 2018).

On the other hand, the GRU model has three-quarters of the parameters of an LSTM one because it combines the forget and input gates into a single update gate, and merges the cell state and hidden state, simplifying the architecture (see Figure 25). This reduction in the number of gates and states results in fewer parameters to train while still offering a significant improvement in performance over traditional RNNs by better capturing dependencies in the data and mitigating issues like the vanishing gradient problem (Shewalkar et al., 2019).

Figure 25

The Design of GRU Model for 128×128 Input Images

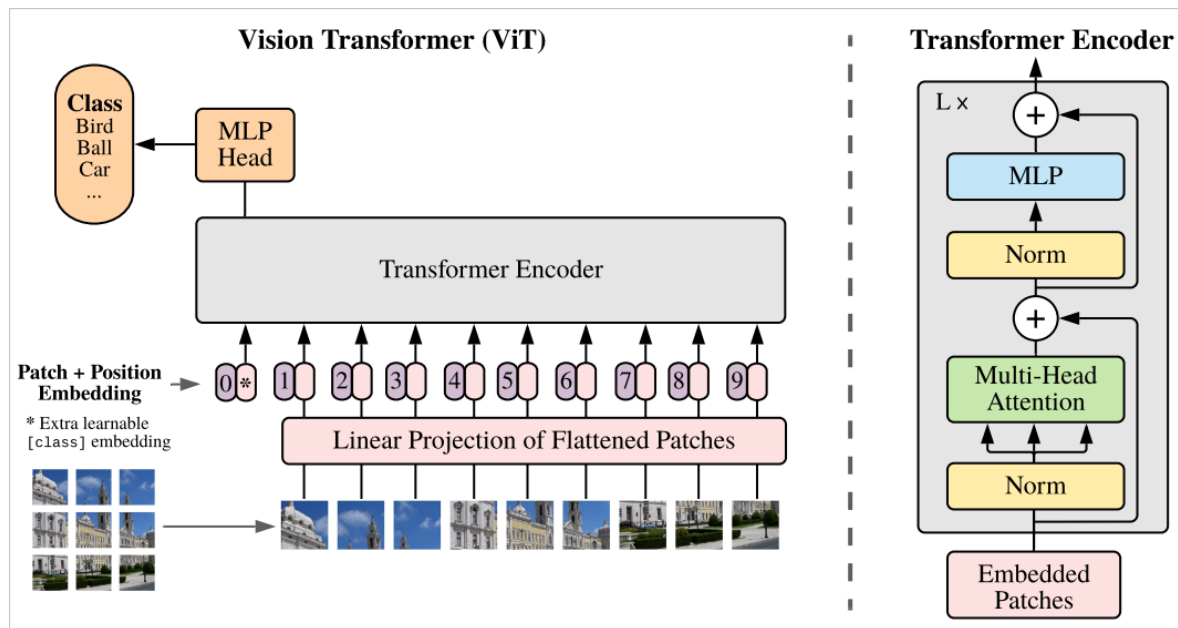
Layer (type)	Output Shape	Param #
GRUModel	[32, 7]	0
└Flatten: 1-1	[32, 1, 16384]	0
└GRU: 1-2	[32, 1, 128]	6,341,376
└GRU: 1-3	[32, 1, 128]	99,072
└Linear: 1-4	[32, 128]	16,512
└Dropout: 1-5	[32, 128]	0
└Linear: 1-6	[32, 7]	903
Total params: 6,457,863		
Trainable params: 6,457,863		
Non-trainable params: 0		

3.2.8 The Vision Transformer (ViT) Model

The vision transformer (ViT) was first introduced by Dosovitskiy et al. (2020), unlike CNNs, which rely on convolutional layers to capture local spatial hierarchies in images, the ViT employs a transformer (Vaswani et al., 2017) architecture to model long-range dependencies directly (see Figure 26).

Figure 26

Vision Transformer (ViT) Architecture (Dosovitskiy et al., 2020)



The ViT uses self-attention to evaluate different parts of an input image, dividing it into patches and embedding them with positional encodings. These embeddings are processed by a transformer encoder with multi-head self-attention and feed-forward networks, capturing global context and understanding the entire image comprehensively (Dosovitskiy et al., 2020). The ViT's scalability allows for efficient handling of high-resolution images, making it suitable for image classification, object detection, and segmentation (P. Zhang et al., 2021).

Despite its promising approach, the ViT presents certain issues, including the need to adapt to transformer-based paradigms and develop new techniques. Additionally, the model's reliance on large-scale pre-training datasets is a significant hurdle, as it requires substantial computational resources and extensive data collection efforts (Dosovitskiy et al., 2020; Ruan

et al., 2022). The ViT's advantages drive further research in transformer architectures and computer vision.

3.2.8.1 The Structure of the ViT Model. The model structure begins with a patch embedding layer that divides each image into smaller patches 16x16 pixels in size. These patches are then projected into a higher dimensional space of 256 features using a convolutional layer, effectively transforming the spatial structure of the image into a sequence suitable for the transformer encoder. A critical component of the model is the addition of positional encodings to the patch embeddings, which allow the transformer to retain information about the relative positions of patches within the image (see Figure 27).

The core of the model consists of six layers of transformer encoders, each featuring multi-head self-attention mechanisms and feed-forward networks. These layers are designed to capture intricate dependencies and interactions between different patches, enabling the model to learn complex patterns that are essential for distinguishing between different facial expressions. The output from the transformer encoder is then globally averaged, reducing the sequence dimension to a single vector that encapsulates the learned features from the entire image. Finally, a fully connected layer maps the aggregated features to the seven facial expression classes.

The summary of the model reveals a total of approximately 7.97 million parameters, all of which are trainable. The model's significant parameter count underscores its capacity to capture detailed features necessary for high-performance facial expression recognition. The

transformer layers alone contribute to the majority of these parameters, highlighting their complexity and depth.

Figure 27

The Design of ViT Model for 128×128 Input Images

Layer (type)	Output Shape	Param #
TransformerModel	[32, 7]	16,384
└─Conv2d: 1-1	[32, 256, 8, 8]	65,792
└─TransformerEncoder: 1-2	[32, 64, 256]	0
└─ModuleList: 2-1	--	0
└─TransformerEncoderLayer: 3-1	[32, 64, 256]	1,315,072
└─TransformerEncoderLayer: 3-2	[32, 64, 256]	1,315,072
└─TransformerEncoderLayer: 3-3	[32, 64, 256]	1,315,072
└─TransformerEncoderLayer: 3-4	[32, 64, 256]	1,315,072
└─TransformerEncoderLayer: 3-5	[32, 64, 256]	1,315,072
└─TransformerEncoderLayer: 3-6	[32, 64, 256]	1,315,072
└─Linear: 1-3	[32, 7]	1,799
=====		
Total params: 7,974,407		
Trainable params: 7,974,407		
Non-trainable params: 0		
=====		

3.2.8.2 Training Process of the RNN, LSTM, GRU, and ViT Models. The training process for the RNN, LSTM, GRU, and ViT models in FER was the same. The input images resized to 128×128 pixels. The training batch size was set to 32, and the models were trained over 120 epochs to ensure thorough learning. The Adam optimizer, with a learning rate of 0.00005, was used to update the model weights.

The computed evaluation metrics were the F1 score, precision, recall, and accuracy for both the training and validation sets. Checkpoints were set up to save the model states based on the best validation accuracy and loss. This involved initializing variables to track the best metrics. By ensuring that the models were saved at their optimal performance, the training

process maintained a high standard of model evaluation and reproducibility.

The training loop encompassed an iterative process over the specified number of epochs. During each epoch, the models were set to training mode, and the cumulative training loss was calculated by iterating over batches from the training data loader. The optimizer gradients were reset, a forward pass was conducted, the loss was computed, a backward pass was performed, and the optimizer was stepped to update the weights. Predictions and true labels were collected throughout this process to compute the training metrics.

In the validation phase, the models were set to evaluation mode, and the cumulative validation loss was calculated by iterating over batches from the validation data loader. Predictions and true labels were collected to compute the validation metrics, ensuring that the models' performance was thoroughly assessed on unseen data. The models were saved based on the best-recorded validation loss and accuracy, ensuring that the final models represented the peak performance during training.

Chapter 4: The Results

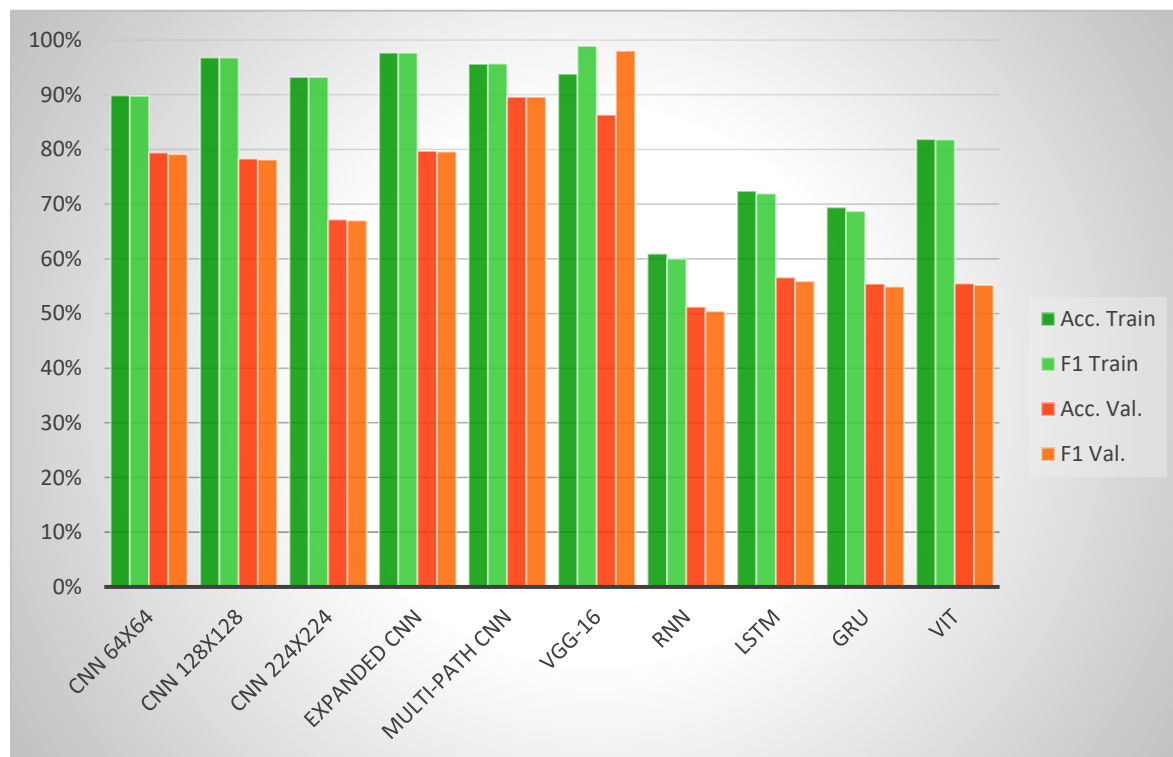
4.1 KDEF Dataset Training Results

The following sections present the training results of the KDEF dataset for both the training and validation phases across various models, including standard CNN for the 64-, 128-, and 224-pixel sizes, expanded CNN, multi-path CNN, VGG16, RNN, LSTM, GRU, and ViT. The performance of these models is evaluated based on several key metrics: accuracy, F1 score, precision, recall, and loss. The consolidated results will be shown first, followed by detailed outcomes for each model.

4.1.1 Consolidated Training Results for All Models

Figure 28

Training Results for the KDEF Dataset



Error! Not a valid bookmark self-reference. provides a comprehensive overview of the training and validation performance of models tested on the KDEF dataset, specifically focusing on accuracy and F1 scores. The results indicate significant variability in performance across the models, with certain models demonstrating superior accuracy and F1 scores in both the training and validation phases.

Table 1

Accuracy and F1 Training Results for All Models on the KDEF Dataset

	Accuracy Train	F1 Train	Accuracy Val.	F1 Val.
CNN 64x64	89.9%	89.80%	79.4%	79.10%
CNN 128x128	96.8%	96.80%	78.3%	78.10%
CNN 224x224	93.2%	93.20%	67.2%	67.00%
Expanded CNN	97.7%	97.70%	79.7%	79.60%
Multi-Path CNN	95.6%	95.70%	89.6%	89.60%
VGG16	93.8%	98.90%	86.3%	98.00%
RNN	60.9%	60.00%	51.2%	50.40%
LSTM	72.4%	71.90%	56.6%	55.90%
GRU	69.4%	68.70%	55.4%	54.90%
ViT	81.9%	81.80%	55.5%	55.20%

Black cells indicate the highest percentage of the metric among all models.

For the standard CNN model, the 128×128 pixel images achieved the highest training

accuracy and F1 score at 96.8%, while its validation metrics were slightly lower at 78.3% accuracy and a 78.1% F1 score. Interestingly, the 224×224 image size, despite its high training accuracy of 93.2%, showed a notable drop in validation accuracy and F1 score to 67.2% and 67.0%, respectively. The 64×64 size performed reasonably well with 89.9% training accuracy and 79.4% validation accuracy.

The expanded CNN model outperformed the standard CNN models with a training accuracy of 97.7% and a validation accuracy of 79.7%. This model also demonstrated high consistency between its training and validation metrics. The multi-path CNN model showed a remarkable validation performance, achieving the highest validation accuracy and F1 score among all the models at 89.6%, while maintaining a solid training accuracy of 95.6%.

The VGG16 model stood out with an exceptional F1 score of 98.9% in training and 98.0% in validation. However, its training accuracy was 93.8%, slightly lower than its F1 score, suggesting the model's strong ability to correctly classify the various facial expressions.

Among the sequential models, the RNN exhibited the lowest performance with a training accuracy of 60.9% and a validation accuracy of 51.2%. The LSTM model performed better, with a training accuracy of 72.4% and a validation accuracy of 56.6%. The GRU model showed similar trends with a training accuracy of 69.4% and a validation accuracy of 55.4%.

Finally, the vision transformer (ViT) model demonstrated balanced but moderate performance, achieving 81.9% training accuracy and 55.5% validation accuracy. Its F1 scores were closely aligned with its accuracy metrics, indicating consistent performance.

The following sections present detailed training results for each individual model.

4.1.2 Standard CNN 64×64 Input Image Size (100 Epochs)

Figure 29

Standard CNN Training Results for 64×64 Images (100 Epochs)

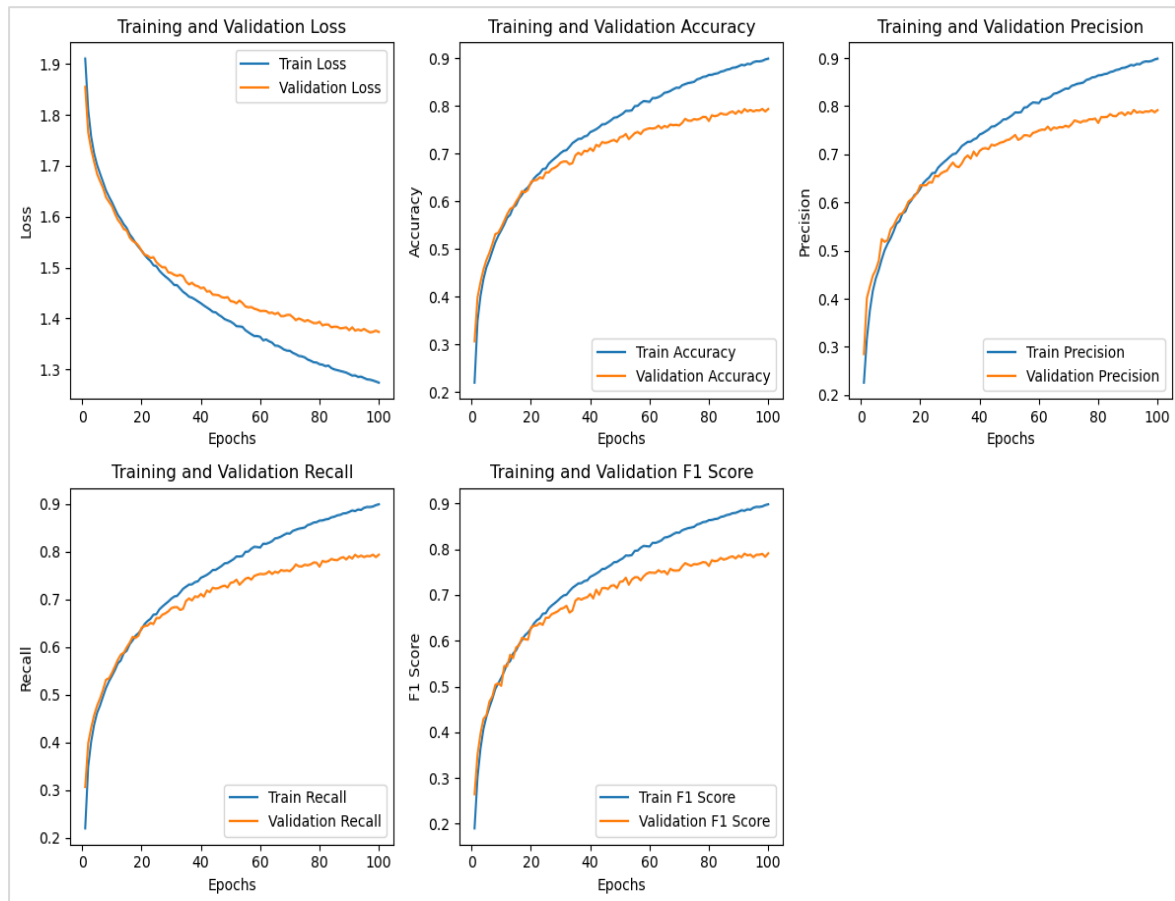


Table 2*Standard CNN Training Summary for 64×64 Images***Accuracy**

training (min: 0.220, max: 0.899, cur: 0.899)

validation (min: 0.307, max: 0.794, cur: 0.794)

F1

training (min: 0.190, max: 0.898, cur: 0.898)

validation (min: 0.264, max: 0.791, cur: 0.791)

Precision

training (min: 0.226, max: 0.899, cur: 0.899)

validation (min: 0.285, max: 0.792, cur: 0.792)

Recall

training (min: 0.220, max: 0.899, cur: 0.899)

validation (min: 0.307, max: 0.794, cur: 0.794)

Loss

training (min: 1.274, max: 1.911, cur: 1.274)

validation (min: 1.373, max: 1.855, cur: 1.373)

4.1.3 Standard CNN 128×128 Input Image Size (100 Epochs)

Figure 30

Standard CNN Training Results for 128×128 Images (100 Epochs)

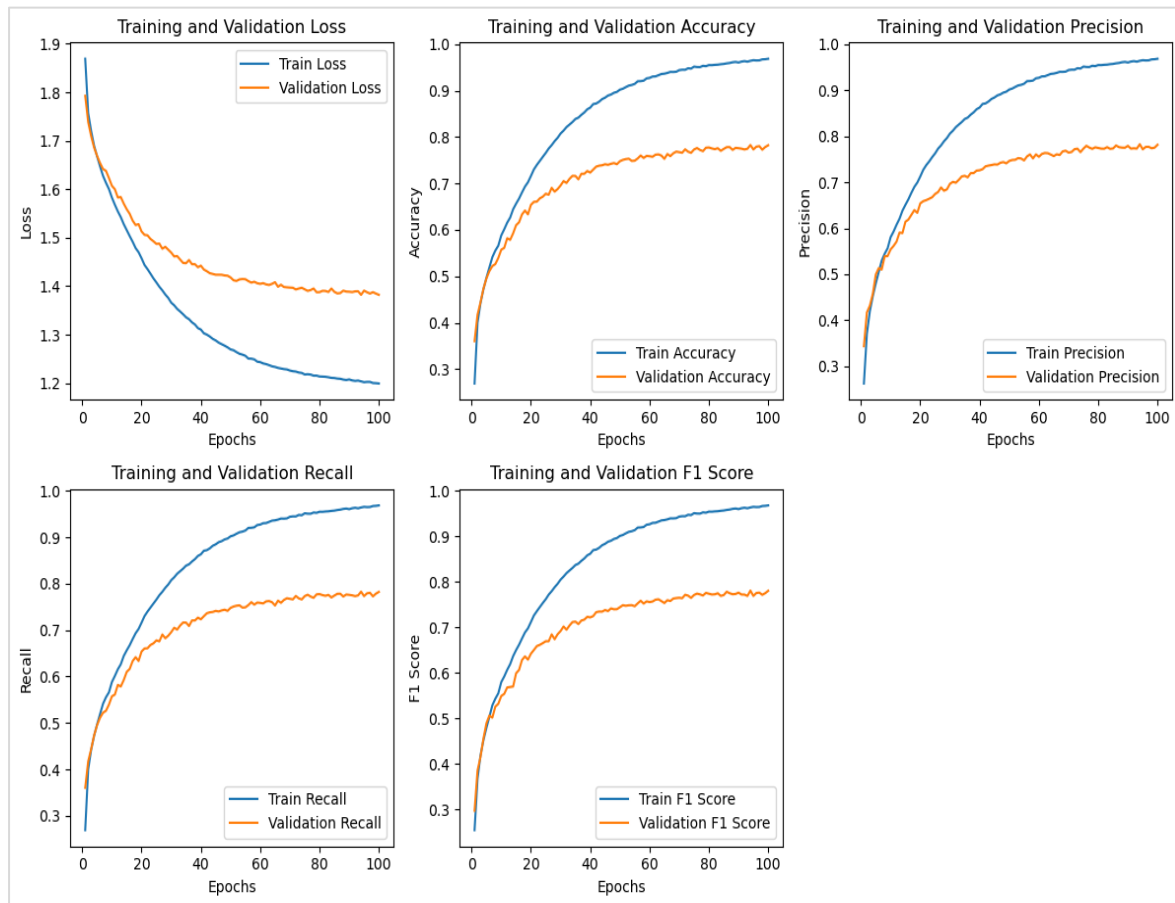


Table 3*Standard CNN Training Summary for 128×128 Images***Accuracy**

training (min: 0.269, max: 0.968, cur: 0.968)

validation (min: 0.361, max: 0.783, cur: 0.782)

F1

training (min: 0.254, max: 0.968, cur: 0.968)

validation (min: 0.297, max: 0.781, cur: 0.781)

Precision

training (min: 0.262, max: 0.968, cur: 0.968)

validation (min: 0.344, max: 0.783, cur: 0.781)

Recall

training (min: 0.269, max: 0.968, cur: 0.968)

validation (min: 0.361, max: 0.783, cur: 0.782)

Loss

training (min: 1.199, max: 1.869, cur: 1.199)

validation (min: 1.382, max: 1.792, cur: 1.382)

4.1.4 Standard CNN 224×224 Input Image Size (80 Epochs)

Figure 31

Standard CNN Training Results for 224×224 Images (100 Epochs)

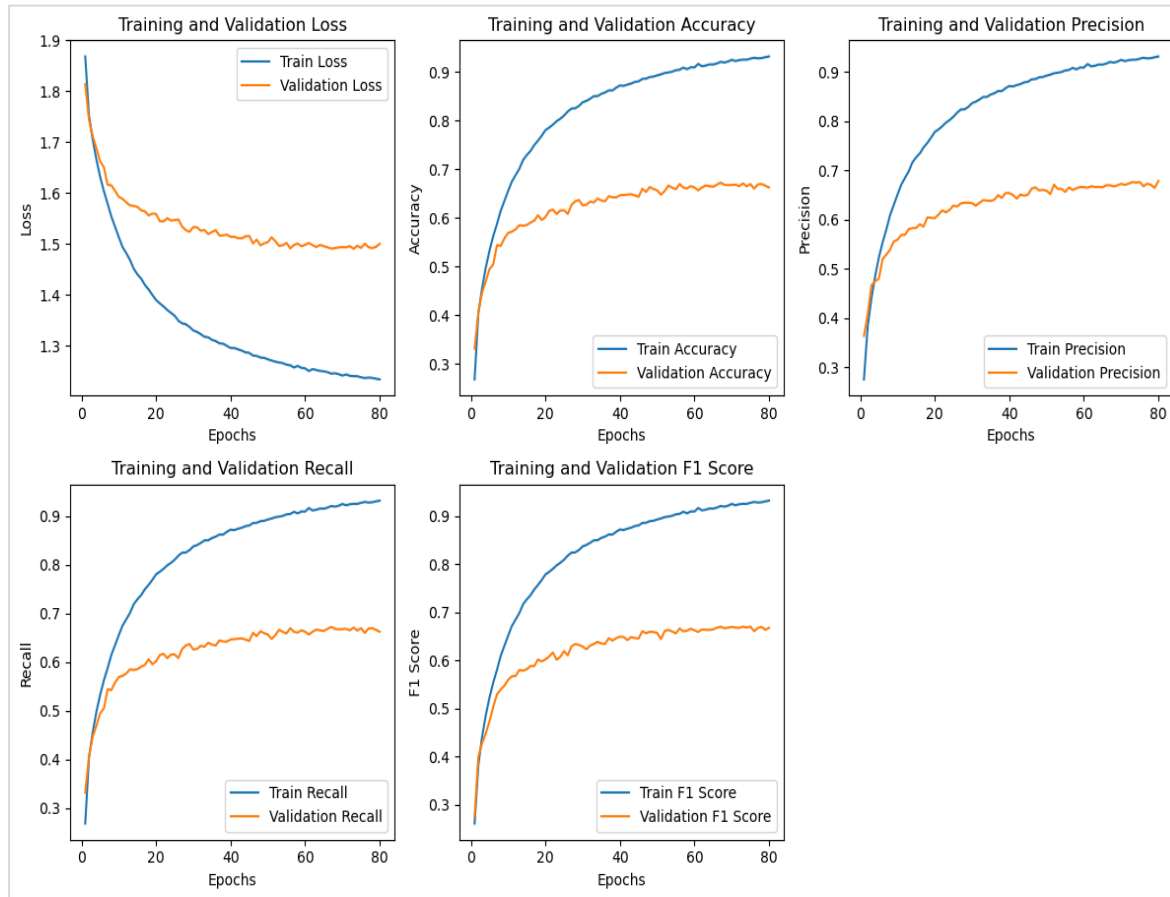


Table 4*Standard CNN Training Summary for 224×224 Images***Accuracy**

training (min: 0.268, max: 0.932, cur: 0.932)

validation (min: 0.332, max: 0.672, cur: 0.663)

F1

training (min: 0.261, max: 0.932, cur: 0.932)

validation (min: 0.275, max: 0.670, cur: 0.667)

Precision

training (min: 0.275, max: 0.932, cur: 0.932)

validation (min: 0.363, max: 0.679, cur: 0.679)

Recall

training (min: 0.268, max: 0.932, cur: 0.932)

validation (min: 0.332, max: 0.672, cur: 0.663)

Loss

training (min: 1.234, max: 1.869, cur: 1.234)

validation (min: 1.491, max: 1.814, cur: 1.501)

4.1.5 Expanded CNN 128×128 Input Image Size (80 Epochs)

Figure 32

Expanded CNN Training Results for 128×128 Images (80 Epochs)

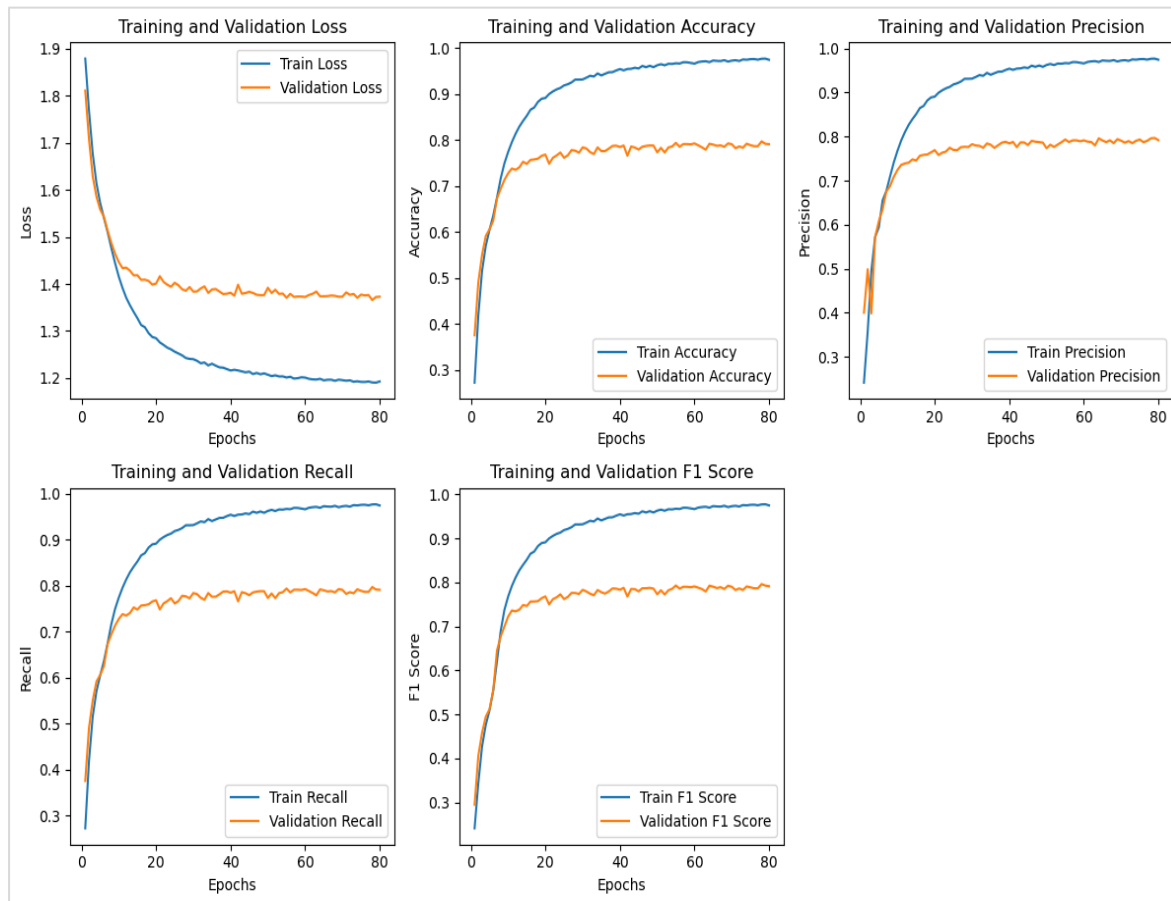


Table 5*Expanded CNN Training Summary for 128×128 Images***Accuracy**

training	(min: 0.272, max: 0.977, cur: 0.975)
validation	(min: 0.375, max: 0.797, cur: 0.791)

F1

training	(min: 0.242, max: 0.977, cur: 0.975)
validation	(min: 0.295, max: 0.796, cur: 0.791)

Precision

training	(min: 0.241, max: 0.977, cur: 0.975)
validation	(min: 0.398, max: 0.797, cur: 0.792)

Recall

training	(min: 0.272, max: 0.977, cur: 0.975)
validation	(min: 0.375, max: 0.797, cur: 0.791)

Loss

training	(min: 1.190, max: 1.879, cur: 1.192)
validation	(min: 1.366, max: 1.811, cur: 1.373)

4.1.6 Multi-Path CNN 64×64 Input Image Size (30 Epochs)

Figure 33

Multi-Path CNN Training Results for 64×64 Images (30 Epochs)

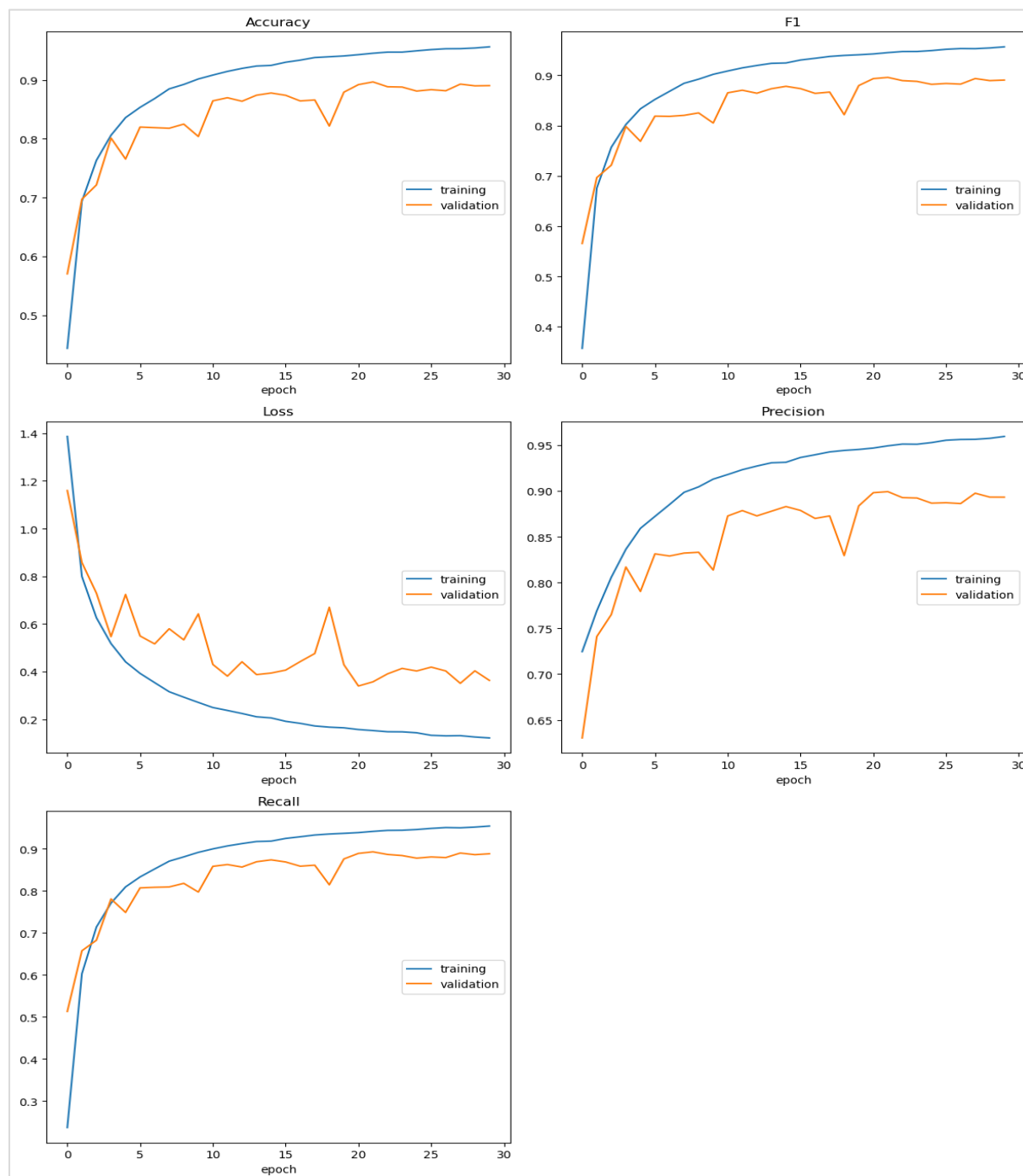


Table 6*Multi-Path CNN Training Summary for 64×64 Images***Accuracy**

training	(min: 0.444, max: 0.956, cur: 0.956)
validation	(min: 0.570, max: 0.896, cur: 0.890)

F1

training	(min: 0.357, max: 0.957, cur: 0.957)
validation	(min: 0.566, max: 0.896, cur: 0.891)

Precision

training	(min: 0.725, max: 0.959, cur: 0.959)
validation	(min: 0.630, max: 0.899, cur: 0.893)

Recall

training	(min: 0.237, max: 0.954, cur: 0.954)
validation	(min: 0.513, max: 0.893, cur: 0.888)

Loss

training	(min: 0.122, max: 1.387, cur: 0.122)
validation	(min: 0.340, max: 1.160, cur: 0.363)

4.1.7 VGG16 224×224 Input Image Size (30 Epochs)

Figure 34

VGG16 Training Results for 224×224 Images (30 Epochs)

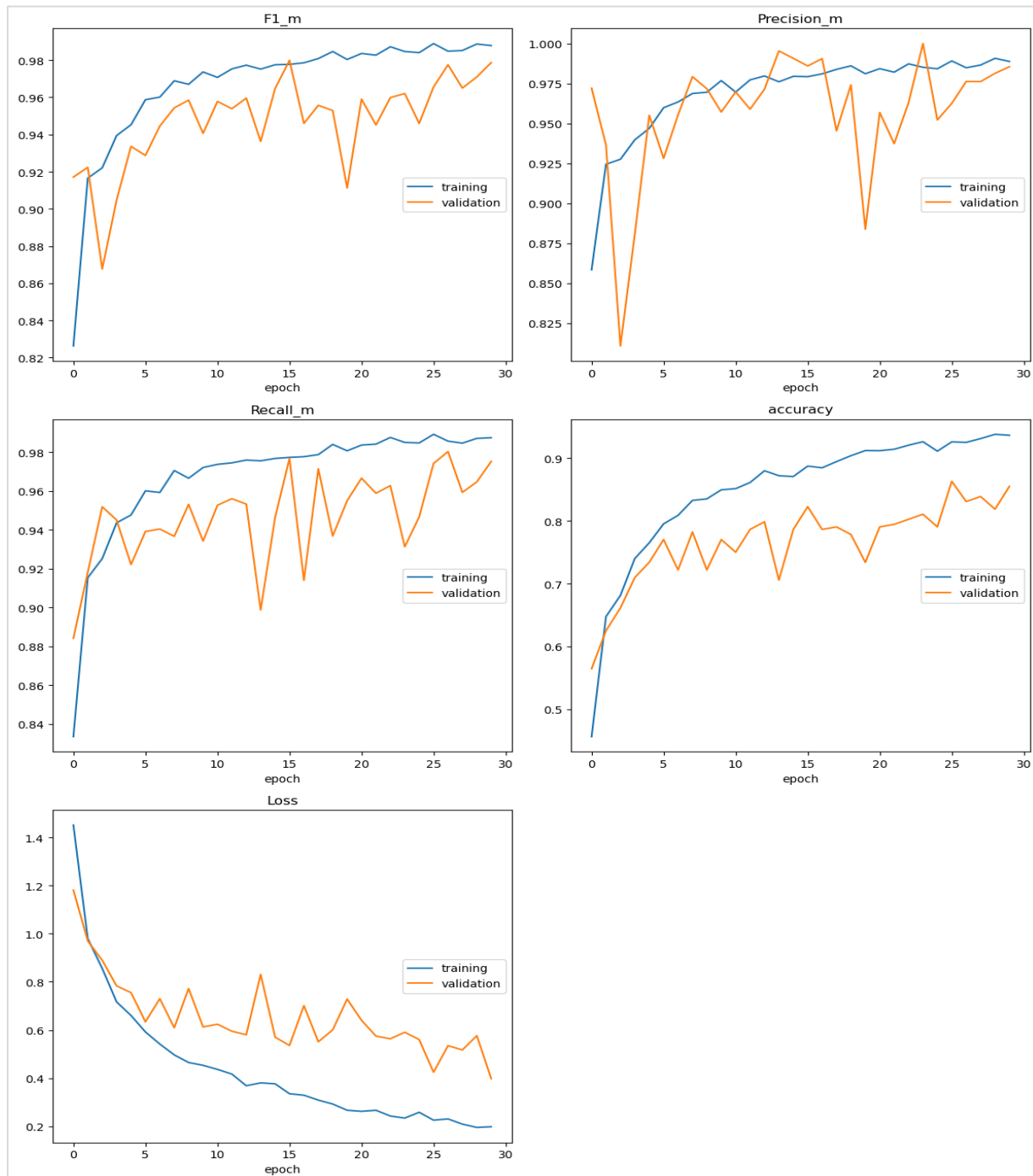


Table 7*Multi-Path CNN Training Summary for 224×224 Images***Accuracy**

training	(min: 0.456, max: 0.938, cur: 0.936)
validation	(min: 0.565, max: 0.863, cur: 0.855)

F1

training	(min: 0.826, max: 0.989, cur: 0.988)
validation	(min: 0.868, max: 0.980, cur: 0.979)

Precision

training	(min: 0.858, max: 0.991, cur: 0.989)
validation	(min: 0.811, max: 1.000, cur: 0.985)

Recall

training	(min: 0.833, max: 0.989, cur: 0.987)
validation	(min: 0.884, max: 0.980, cur: 0.975)

Loss

training	(min: 0.196, max: 1.451, cur: 0.199)
validation	(min: 0.398, max: 1.181, cur: 0.398)

4.1.8 RNN 128×128 Input Image Size (120 Epochs)

Figure 35

RNN Training Results for 128×128 Images (120 Epochs)

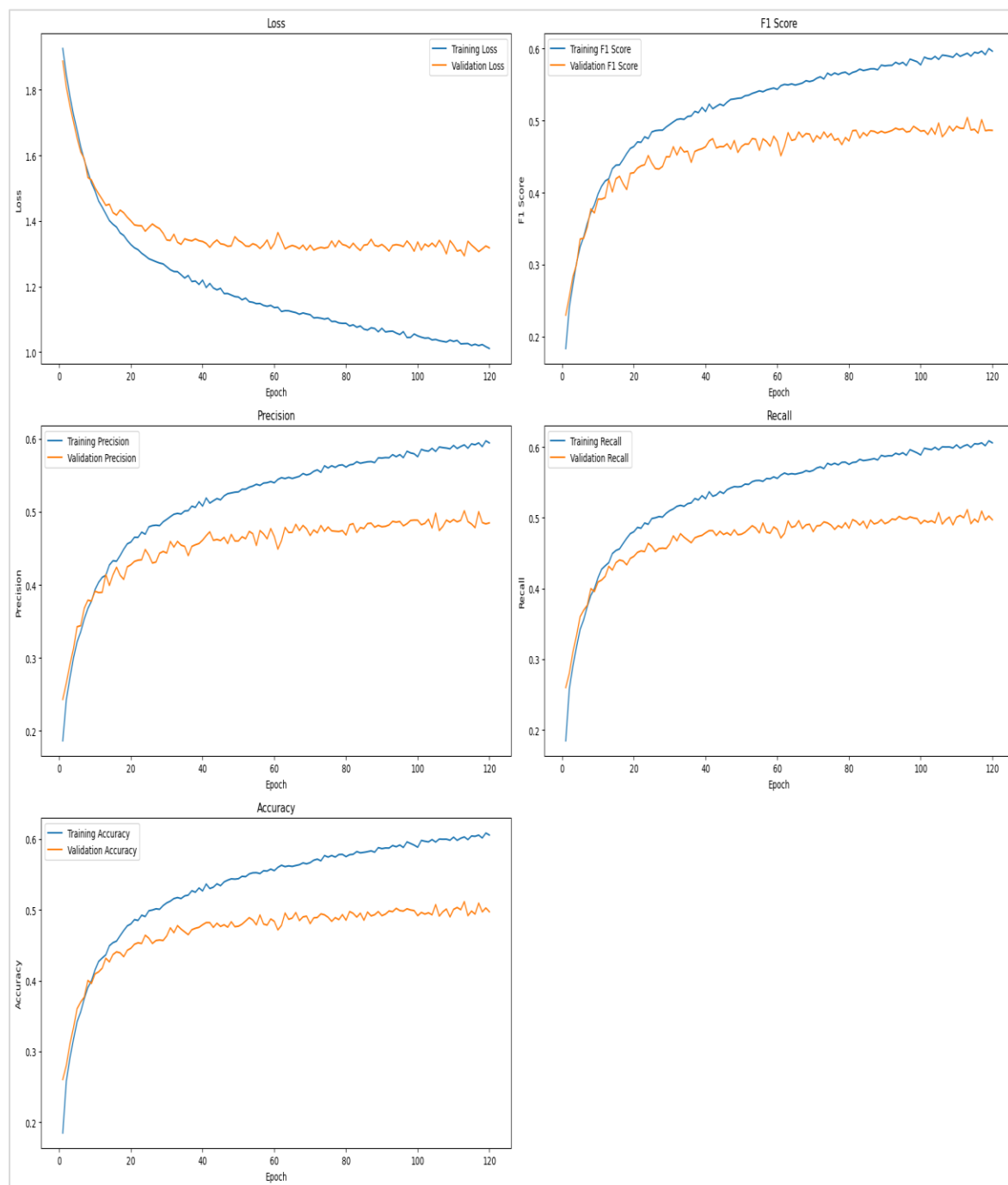


Table 8*RNN Training Summary for 128×128 Images***Accuracy**

training	(min: 0.185, max: 0.609, cur: 0.606)
validation	(min: 0.260, max: 0.512, cur: 0.497)

F1

training	(min: 0.183, max: 0.600, cur: 0.596)
validation	(min: 0.230, max: 0.504, cur: 0.486)

Precision

training	(min: 0.186, max: 0.598, cur: 0.595)
validation	(min: 0.243, max: 0.502, cur: 0.485)

Recall

training	(min: 0.185, max: 0.609, cur: 0.606)
validation	(min: 0.260, max: 0.511, cur: 0.497)

Loss

training	(min: 1.011, max: 1.926, cur: 1.011)
validation	(min: 1.294, max: 1.888, cur: 1.318)

4.1.9 LSTM 128×128 Input Image Size (120 Epochs)

Figure 36

LSTM Training Results for 128×128 Images (120 Epochs)

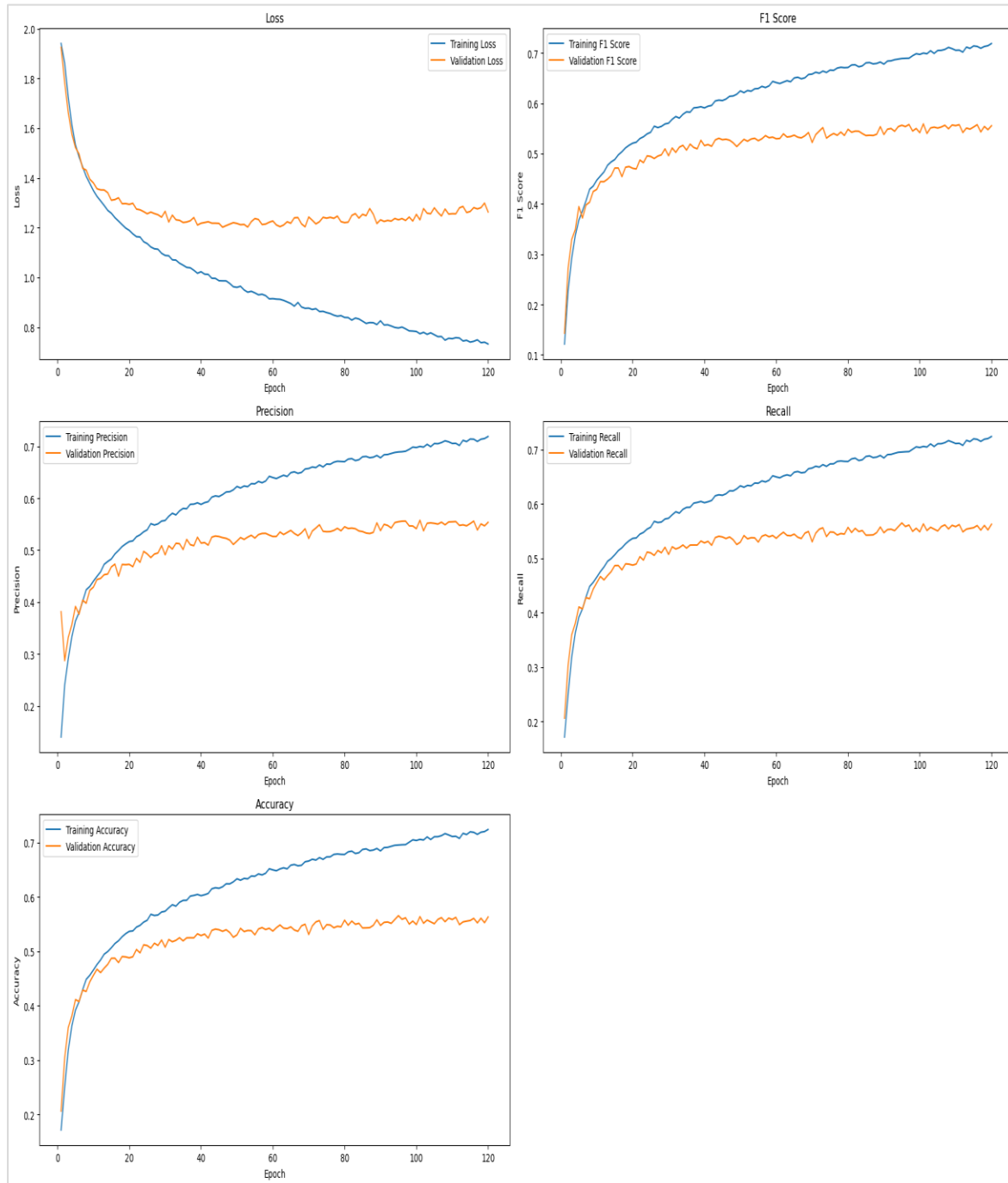


Table 9*LSTM Training Summary for 128×128 Images***Accuracy**

training (min: 0.172, max: 0.724, cur: 0.724)

validation (min: 0.207, max: 0.566, cur: 0.563)

F1

training (min: 0.122, max: 0.719, cur: 0.719)

validation (min: 0.143, max: 0.559, cur: 0.556)

Precision

training (min: 0.141, max: 0.719, cur: 0.719)

validation (min: 0.287, max: 0.558, cur: 0.554)

Recall

training (min: 0.172, max: 0.724, cur: 0.724)

validation (min: 0.207, max: 0.565, cur: 0.563)

Loss

training (min: 0.732, max: 1.940, cur: 0.732)

validation (min: 1.202, max: 1.924, cur: 1.263)

4.1.10 GRU 128×128 Input Image Size (120 Epochs)

Figure 37

GRU Training Results for 128×128 Images (120 Epochs)

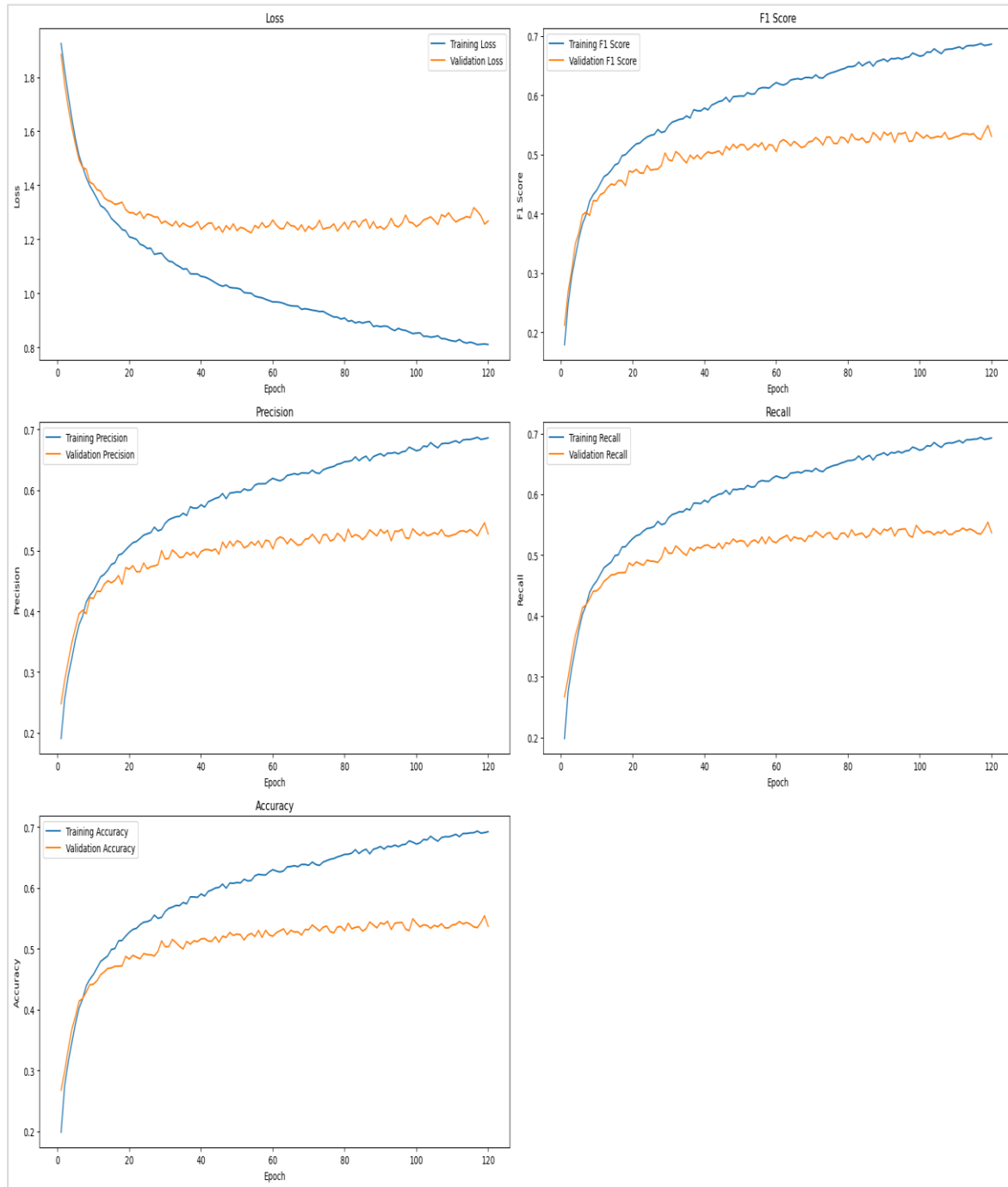


Table 10*GRU Training Summary for 128×128 Images***Accuracy**

training (min: 0.198, max: 0.694, cur: 0.693)

validation (min: 0.267, max: 0.554, cur: 0.537)

F1

training (min: 0.179, max: 0.687, cur: 0.686)

validation (min: 0.212, max: 0.549, cur: 0.531)

Precision

training (min: 0.190, max: 0.687, cur: 0.686)

validation (min: 0.248, max: 0.546, cur: 0.528)

Recall

training (min: 0.198, max: 0.694, cur: 0.693)

validation (min: 0.267, max: 0.554, cur: 0.537)

Loss

training (min: 0.810, max: 1.925, cur: 0.810)

validation (min: 1.224, max: 1.884, cur: 1.267)

4.1.11 ViT 128×128 Input Image Size (120 Epochs)

Figure 38

ViT Training Results for 128×128 Images (120 Epochs)

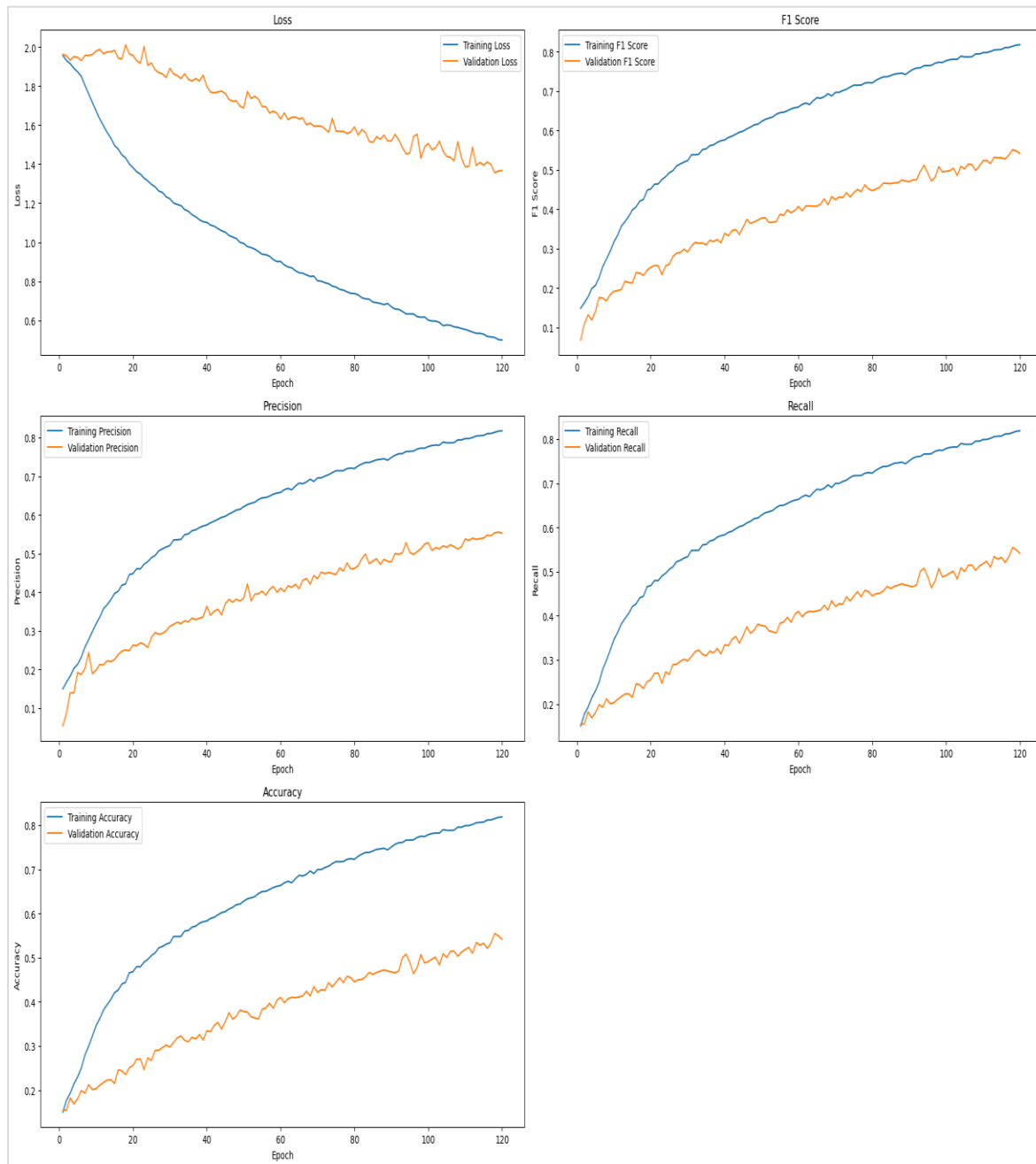


Table 11*ViT Training Summary for 128×128 Images***Accuracy**

training (min: 0.150, max: 0.819, cur: 0.819)

validation (min: 0.154, max: 0.555, cur: 0.542)

F1

training (min: 0.149, max: 0.818, cur: 0.818)

validation (min: 0.069, max: 0.552, cur: 0.542)

Precision

training (min: 0.149, max: 0.818, cur: 0.818)

validation (min: 0.053, max: 0.555, cur: 0.553)

Recall

training (min: 0.150, max: 0.819, cur: 0.819)

validation (min: 0.154, max: 0.555, cur: 0.541)

Loss

training (min: 0.499, max: 1.957, cur: 0.499)

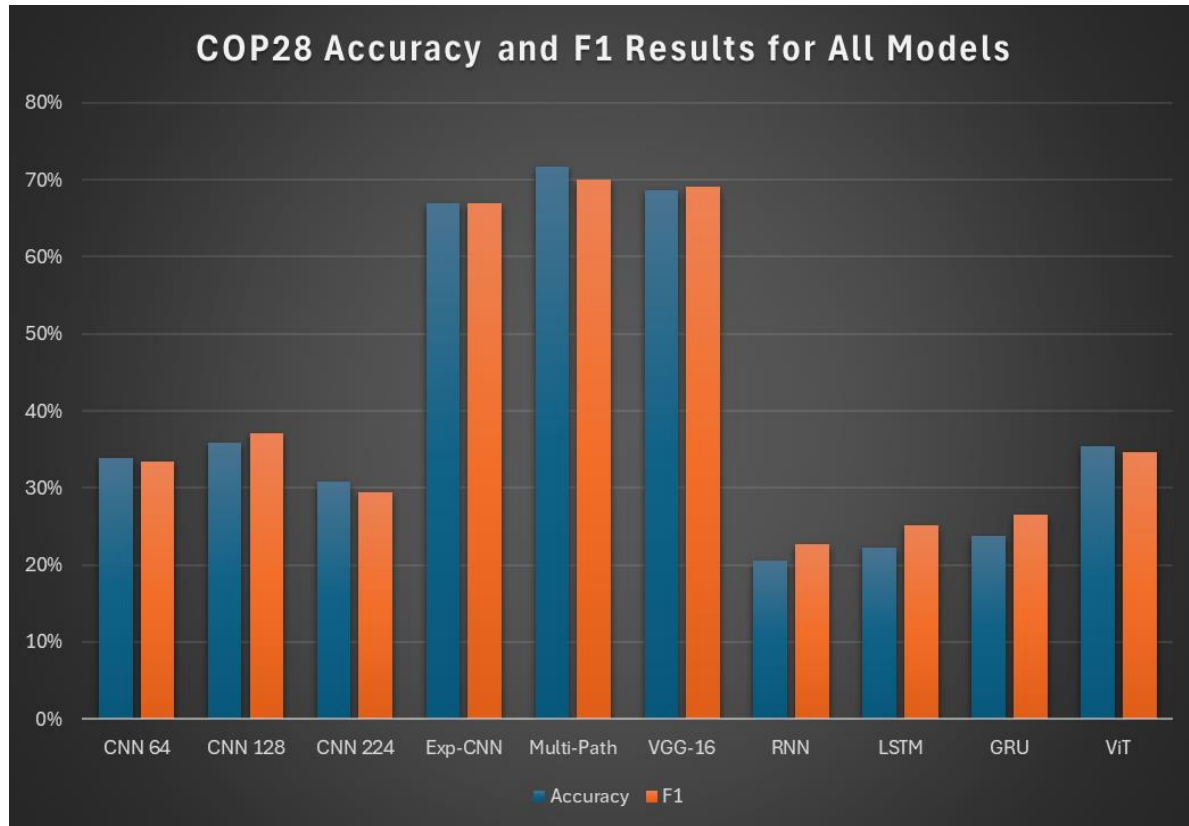
validation (min: 1.356, max: 2.012, cur: 1.368)

4.2 COP28 FER Results

4.2.1 Consolidated COP28 FER Prediction Results (Accuracy and F1)

Figure 39

The Results for COP28 Unseen Test Images for All Models



The application of various deep learning models on unseen test images of COP28 for FER yielded a range of results in terms of accuracy and F1 scores (see Figure 39 and Table 12 *COP28 FER Models Results*). The performance of these models demonstrates the differences in their ability to generalize and accurately classify the facial expressions present in the test data.

The CNN models, evaluated at different image resolutions, showed varied performance levels. The standard CNN model applied to 64x64 pixel images achieved an accuracy of

33.87% and an F1 score of 33.45%. The performance slightly improved when run with 128x128 pixel images, which achieved an accuracy of 35.91% and an F1 score of 37.11%. However, with 224x224 pixel images the model performed lower than expected, with an accuracy of 30.88% and an F1 score of 29.43%.

Table 12

COP28 FER Models Results

Model	Accuracy	F1
CNN 64x64	33.87%	33.45%
CNN 128x128	35.91%	37.11%
CNN 224x224	30.88%	29.43%
expanded CNN	67.02%	66.93%
Multi-Path CNN	71.70%	70.04%
VGG16	68.68%	69.08%
RNN	20.50%	22.68%
LSTM	22.20%	25.14%
GRU	23.70%	26.57%
ViT	35.39%	34.71%

The expanded CNN model exhibited a significantly higher performance compared to the standard CNN models. This model achieved an accuracy of 67.02% and an F1 score of 66.93%, indicating its better capability in recognizing facial expressions in the test images. Similarly, the multi-path CNN model demonstrated a strong performance, achieving the highest accuracy of 71.70% and an F1 score of 70.04%.

The VGG16 model, known for its deep architecture, also performed well, achieving an accuracy of 68.68% and an F1 score of 69.08%. This result suggests that the VGG16 model is effective in capturing the complex features required for accurate facial expression recognition.

The RNN-based models, including RNN, LSTM, and GRU, showed relatively lower performances in comparison to the CNN-based models. The RNN model achieved an accuracy of 20.50% and an F1 score of 22.68%. The LSTM model, which is designed to handle long-term dependencies better, achieved an accuracy of 22.20% and an F1 score of 25.14%. The GRU model, which is a variant of LSTM, performed slightly better than RNN and LSTM, with an accuracy of 23.70% and an F1 score of 26.57%.

Lastly, the vision transformer (ViT) model achieved an accuracy of 35.39% and an F1 score of 34.71%. This model showed comparable performance to the CNN models applied at lower resolutions but did not outperform the best-performing CNN and VGG16 models.

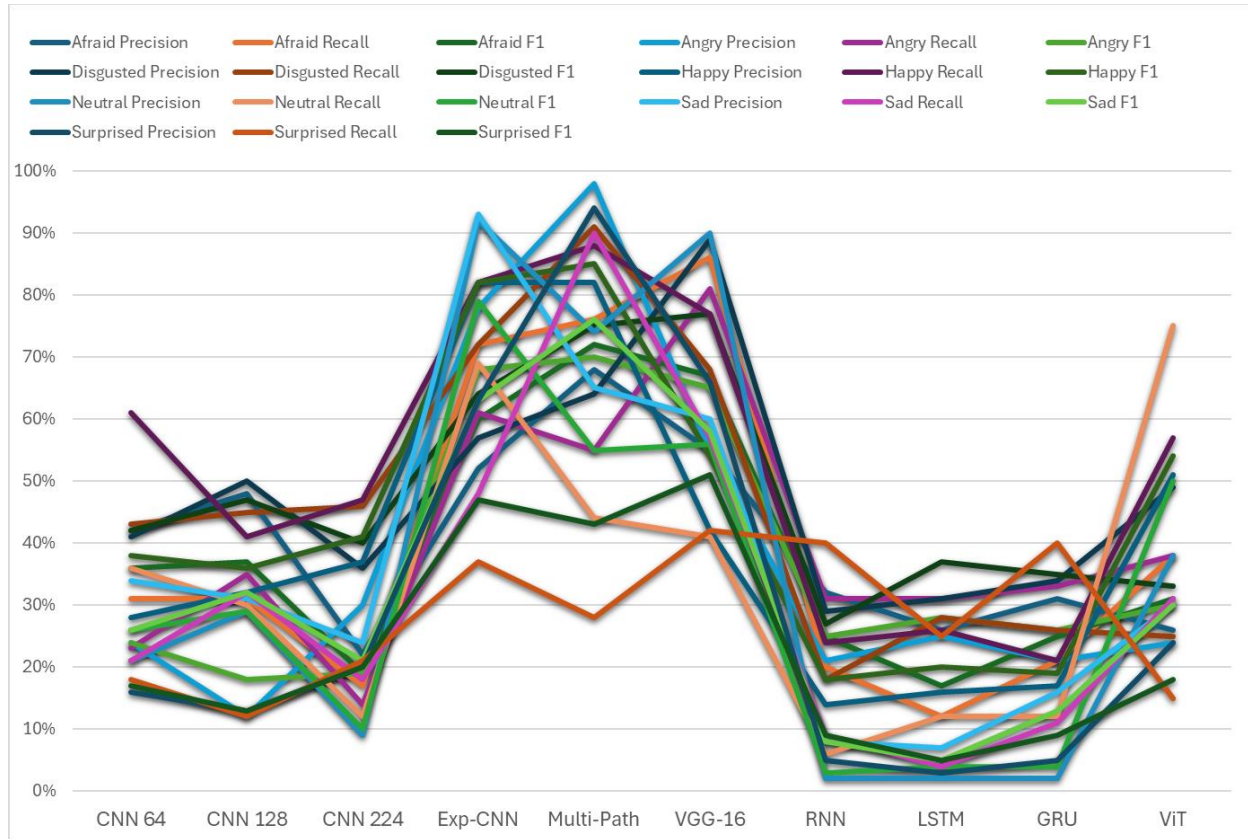
These results highlight the varied performance of different deep learning architectures in the task of facial expression recognition on the COP28 dataset, reflecting their respective strengths and limitations in handling the complexities of the test images.

4.2.2 Detailed Facial Expression Results for COP28 Test Images

If we more closely analyze the performance metrics of various deep learning models applied to the unseen test images from COP28 for FER, notable patterns and significant results can be observed across different expressions and models in terms of precision, recall, and F1 scores (see Figure 40, Figure 41, and Table 13).

Figure 40

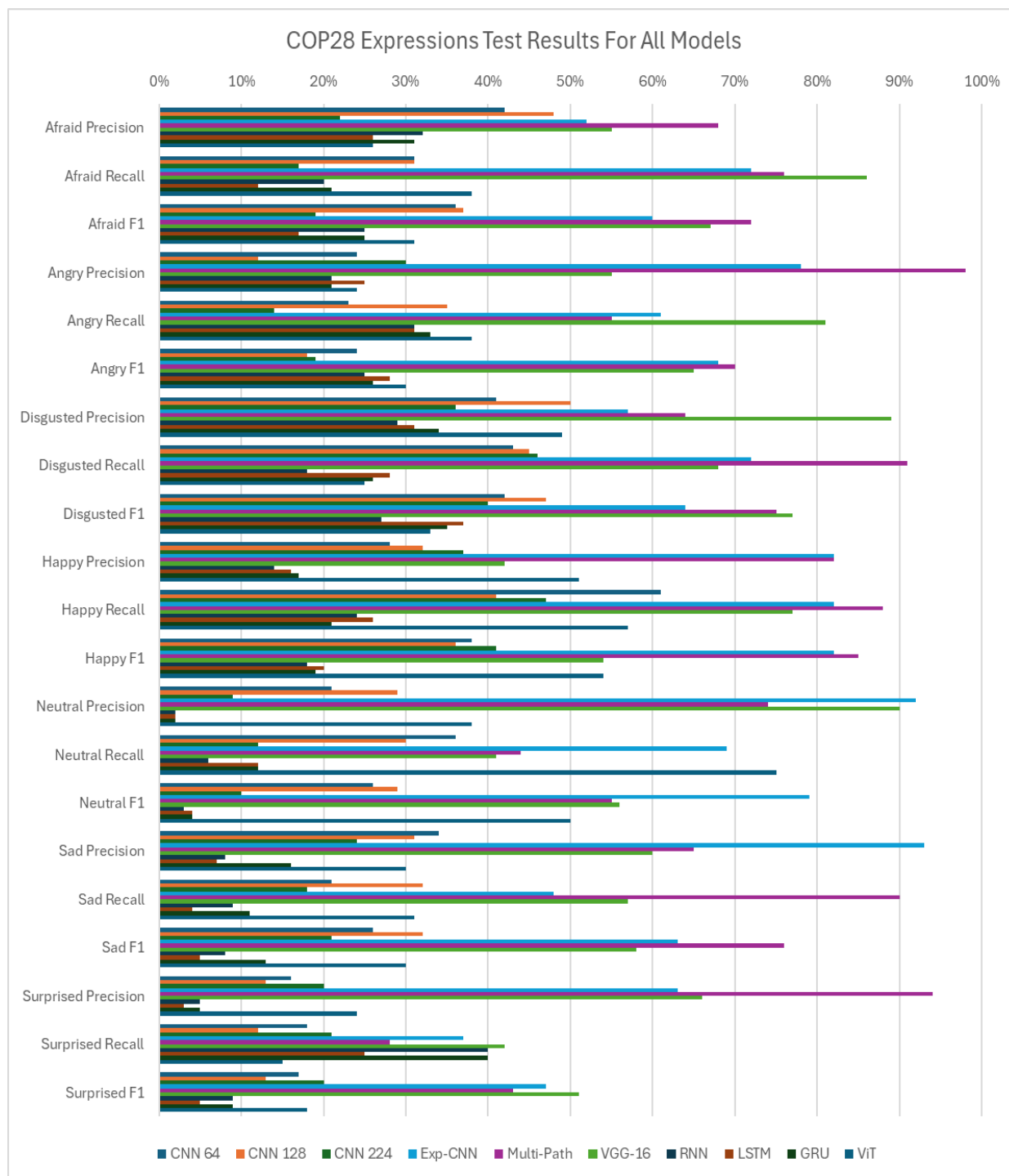
Facial Expression Results for COP28 Test Images (Models-Based)



For the expression ‘afraid’, the expanded CNN (Exp-CNN) and multi-path CNN models demonstrated significant enhancements over the standard CNN models. The Exp-CNN achieved 52% precision, 72% recall, and a 60% F1 score, while the multi-path CNN excelled with 68% precision, 76% recall, and a 72% F1 score. In contrast, the CNN models applied to different resolutions (64, 128, and 224) showed lower performance, with precision ranging from 22% to 48% and F1 scores from 19% to 37%. The VGG16 model also performed well with 55% precision and 86% recall, achieving a 67% F1 score.

Figure 41

Results for COP28 Test Images (Facial Expressions-Based)



Models like the RNN, LSTM, and GRU exhibited lower metrics, with notable performance improvements only in the ViT model, which achieved 38% recall.

For the ‘angry’ expression, the multi-path CNN model stood out with a precision of 98%, while the VGG16 model demonstrated high recall at 81%. The Exp-CNN also performed robustly with precision at 78% and a 68% F1 score. In contrast, the CNN 128 model exhibited low precision at 12%. The performance of the RNN, LSTM, and GRU models was generally lower, highlighting the superior performance of CNN-based models and VGG16 in capturing this expression.

Table 13

COP28 Facial Expression per Metric per Model

Expression	Metric	CNN 64	CNN 128	CNN 224	Exp-CNN	Multi-Path	VGG16	RNN	LSTM	GRU	ViT
Afraid	Precision	42%	48%	22%	52%	68%	55%	32%	26%	31%	26%
	Recall	31%	31%	17%	72%	76%	86%	20%	12%	21%	38%
	F1	36%	37%	19%	60%	72%	67%	25%	17%	25%	31%
Angry	Precision	24%	12%	30%	78%	98%	55%	21%	25%	21%	24%
	Recall	23%	35%	14%	61%	55%	81%	31%	31%	33%	38%
	F1	24%	18%	19%	68%	70%	65%	25%	28%	26%	30%
Disgusted	Precision	41%	50%	36%	57%	64%	89%	29%	31%	34%	49%
	Recall	43%	45%	46%	72%	91%	68%	18%	28%	26%	25%
	F1	42%	47%	40%	64%	75%	77%	27%	37%	35%	33%
Happy	Precision	28%	32%	37%	82%	82%	42%	14%	16%	17%	51%
	Recall	61%	41%	47%	82%	88%	77%	24%	26%	21%	57%
	F1	38%	36%	41%	82%	85%	54%	18%	20%	19%	54%
Neutral	Precision	21%	29%	9%	92%	74%	90%	02%	02%	02%	38%
	Recall	36%	30%	12%	69%	44%	41%	06%	12%	12%	75%
	F1	26%	29%	10%	79%	55%	56%	03%	04%	04%	50%
Sad	Precision	34%	31%	24%	93%	65%	60%	08%	07%	16%	30%
	Recall	21%	32%	18%	48%	90%	57%	09%	04%	11%	31%
	F1	26%	32%	21%	63%	76%	58%	08%	05%	13%	30%
Surprised	Precision	16%	13%	20%	63%	94%	66%	05%	03%	05%	24%

Recall	18%	12%	21%	37%	28%	42%	40%	25%	40%	15%
F1	17%	13%	20%	47%	43%	51%	09%	05%	09%	18%

In the case of the ‘disgusted’ expression, the VGG16 model exhibited the highest precision at 89%, while the multi-path CNN model achieved the highest recall at 91% and a notable F1 score of 75%. The Exp-CNN demonstrated a balanced performance with 57% precision and 72% recall. The performances of the CNN models were moderate, with precision ranging from 36% to 50%. Models like the RNN and LSTM ones had lower metrics, with the GRU model performing slightly better.

For the ‘happy’ expression, both Exp-CNN and multi-path CNN models achieved high precision and recall at 82%, with multi-path CNN reaching an F1 score of 85%. VGG16 also showed a strong performance with a 77% recall and a 54% F1 score. The CNN models demonstrated moderate improvements in precision and recall with increasing image resolution, whereas the RNN, LSTM, GRU, and ViT models exhibited significantly lower performance.

The ‘neutral’ expression saw the Exp-CNN model achieving remarkable results with 92% precision and a 79% F1 score. The multi-path CNN and VGG16 models also performed well, with precision results of 74% and 90%, respectively. However, recall was notably lower for VGG16 at 41%. The CNN models showed improvement with higher resolutions, but the overall performance remained moderate. The RNN, LSTM, and GRU models demonstrated minimal effectiveness for this expression, with very low precision and recall metrics.

For the ‘sad’ expression, the Exp-CNN demonstrated an outstanding performance with 93% precision, while the multi-path CNN model excelled with recall at 90% and a 76% F1

score. VGG16 also performed well with a balanced precision and recall around 60%. The CNN models showed moderate precision, particularly the CNN 64 and CNN 128 models. The RNN, LSTM, and GRU models had a much lower performance, with precision and recall below 20% in many cases.

The ‘surprised’ expression revealed that the multi-path CNN model achieved exceptional precision at 94%, though its recall was lower at 28%. The Exp-CNN model had a more balanced performance with 63% precision and a 47% F1 score. VGG16 demonstrated a good overall performance with 66% precision and a 51% F1 score. The CNN models showed variability in their performances with moderate precision and recall. The RNN, LSTM, GRU, and ViT models struggled with this expression, with notably low metrics across all categories.

These detailed observations highlight the varied performance of different deep learning architectures in the task of facial expression recognition on the COP28 dataset. The superior performance of models like the multi-path CNN, Exp-CNN, and VGG16 underscores their effectiveness in accurately recognizing complex facial expressions. Conversely, the relatively lower performance of the RNN, LSTM, and GRU models suggests that these architectures may require further refinement or different approaches to improve their effectiveness in this specific context.

4.2.3 Examples of COP28 FER Results for All Models

Figure 42

Sample of COP28 Prediction Results for Standard CNN 128×128

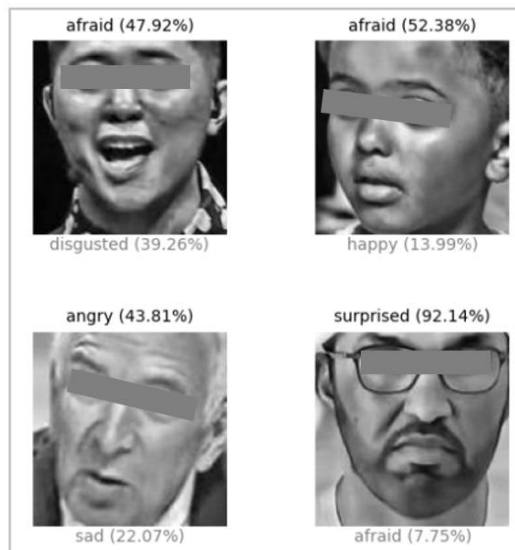


Figure 43

Sample of COP28 Prediction Results for Expanded CNN Model

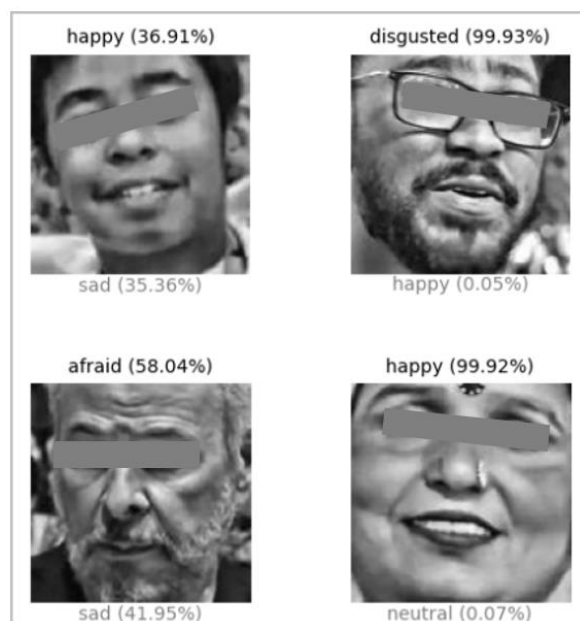
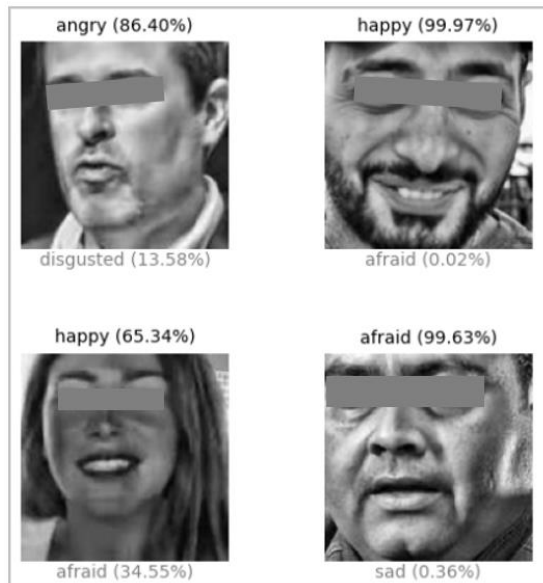


Figure 44

Sample of COP28 Prediction Results for Multi-Path CNN Model

**Figure 45**

Sample of COP28 Prediction Results for VGG16 Model

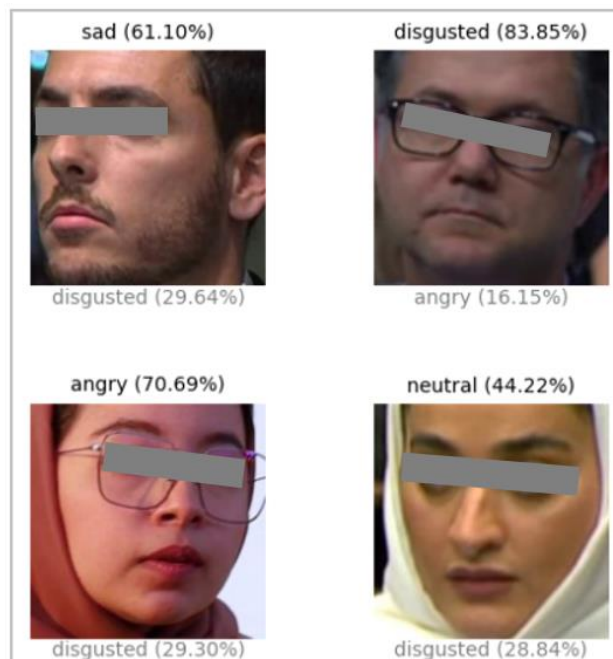
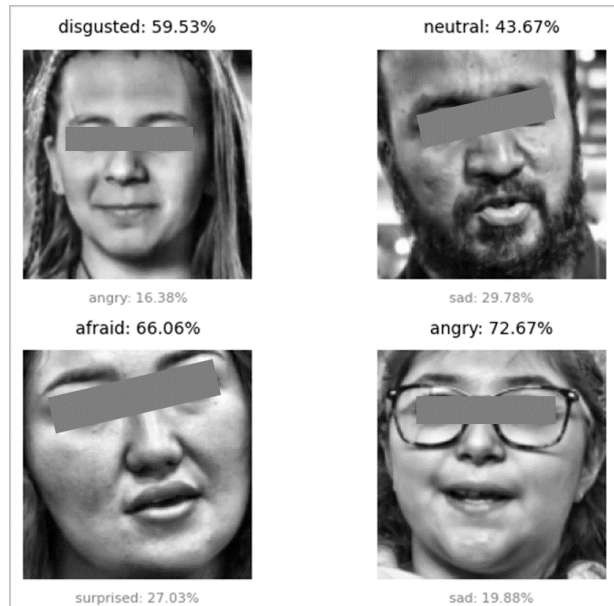


Figure 46

Sample of COP28 Prediction Results for RNN Model

**Figure 47**

Sample of COP28 Prediction Results for LSTM Model

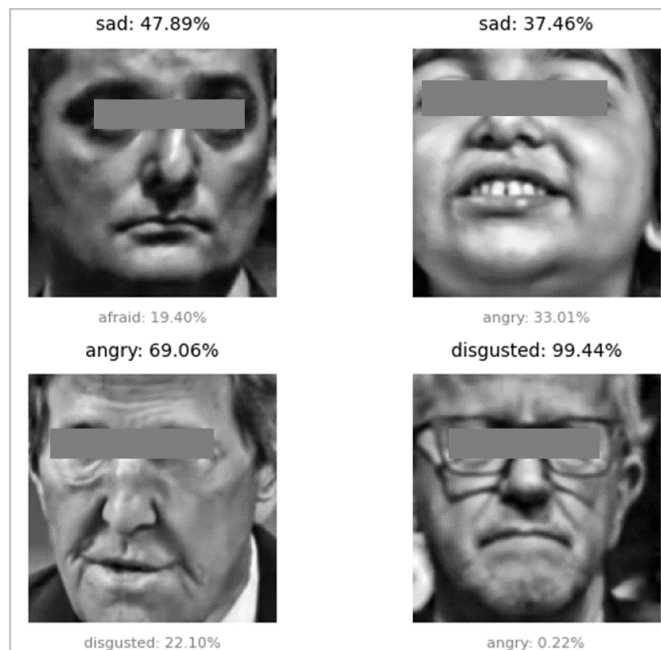
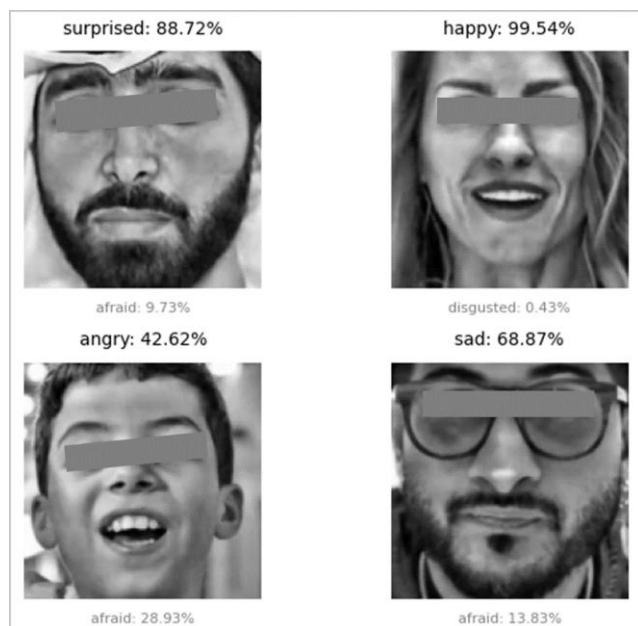
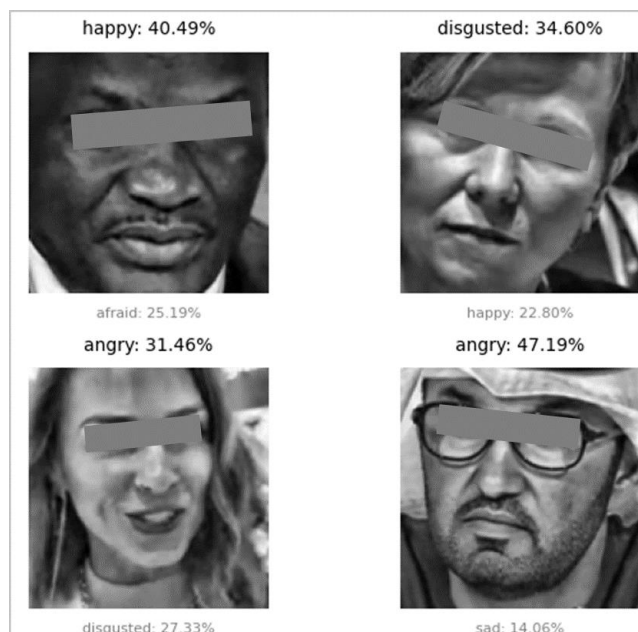


Figure 48

Sample of COP28 Prediction Results for GRU Model

**Figure 49**

Sample of COP28 Prediction Results for ViT Model



Chapter 5: Discussion and Evaluation

5.1 Consolidation Analysis of KDEF Training and Validation Results

The analysis of training and validation results for the KDEF dataset reveals distinct performance characteristics among the CNN-based models and non-CNN-based models. The data in Error! Not a valid bookmark self-reference. provides a comprehensive overview of the training and validation performance of models tested on the KDEF dataset, specifically focusing on accuracy and F1 scores. The results indicate significant variability in performance across the models, with certain models demonstrating superior accuracy and F1 scores in both the training and validation phases.

Table 1 clearly shows that CNN-based models generally outperform non-CNN-based models in terms of both training and validation accuracy and F1 scores.

CNN-based models, such as the standard CNN, expanded CNN, multi-path CNN, and VGG16, exhibit strong training performance. The expanded CNN model achieved the highest training accuracy and F1 score, with both at 97.7%, indicating its exceptional ability to learn from the training data. This model also maintained a high validation accuracy of 79.7% and an F1 score of 79.6%, reflecting its robustness and ability to generalize well. The multi-path CNN also showed impressive results, with a training accuracy of 95.6% and an F1 score of 95.7%, and an even higher validation accuracy and F1 score of 89.6%. This suggests that the multi-path CNN is highly effective in capturing complex features in the data and translating that learning to new, unseen data.

The VGG16 model, although slightly lower in training accuracy at 93.8%, achieved an

extraordinarily high training F1 score of 98.9%. This indicates a strong ability to classify expressions correctly during training. Its validation performance, with an accuracy of 86.3% and an F1 score of 98.0%, underscores its strong generalization capabilities. In contrast, the standard CNN model with smaller and larger input sizes (64×64 and 224×224) showed a robust training performance, particularly for the 128×128 input size, which achieved a training accuracy of 96.8% and an F1 score of 96.8%. However, their validation performance was lower compared to the multi-path CNN and VGG16, indicating potential overfitting or the need for better optimization strategies.

Non-CNN-based models, including RNN, LSTM, GRU, and ViT, generally displayed lower performance metrics compared to their CNN counterparts. The RNN model, for instance, had the lowest training accuracy at 60.9% and an F1 score of 60.0%, with even poorer validation results (51.2% accuracy and a 50.4% F1 score). This suggests that the RNN struggled to capture and retain the necessary features for accurate expression recognition, both during training and validation. The LSTM and GRU models showed moderate improvements over the RNN, with training accuracies of 72.4% and 69.4%, and F1 scores of 71.9% and 68.7%, respectively. However, their validation performance remained suboptimal, with accuracies of 56.6% and 55.4%, and F1 scores of 55.9% and 54.9%, respectively, indicating their limited generalization capability.

The ViT model, which utilizes transformer architecture, performed better than other non-CNN models, achieving a training accuracy of 81.9% and an F1 score of 81.8%. However, its validation performance, with an accuracy of 55.5% and an F1 score of 55.2%, still fell short compared to the leading CNN-based models. This suggests that while the ViT model is

capable of learning from the training data, it struggles to maintain that performance when applied to new data, likely due to the model's high complexity and potential overfitting.

5.2 Consolidation Analysis of the COP28 Prediction Results

The performance comparison of CNN-based models and non-CNN models on the COP28 dataset offers valuable insights into their strengths and weaknesses in facial expression recognition tasks. The data from Table 12 reveal that CNN-based models generally outperform non-CNN models, demonstrating higher overall accuracy and F1 scores. Among the CNN-based models, the multi-path CNN stands out with the highest accuracy of 71.70% and an F1 score of 70.04%, indicating its superior capability in capturing and interpreting facial expressions. The expanded CNN and VGG16 also perform admirably, achieving accuracies of 67.02% and 68.68% respectively, with corresponding F1 scores of 66.93% and 69.08%. These results suggest that the sophisticated architectures of these models, which incorporate multiple layers and advanced feature extraction techniques, significantly enhance their performance.

In contrast, the standard CNNs with different input sizes (64x64, 128x128, and 224x224) show varying levels of effectiveness. The CNN 128x128 model outperforms the other standard CNN models with an accuracy of 35.91% and an F1 score of 37.11%, indicating that a balanced input size can optimize performance. The CNN 64x64 and CNN 224x224 models, however, demonstrate lower effectiveness, with the latter achieving the lowest accuracy among CNN-based models at 30.88% and an F1 score of 29.43%. This indicates that simply increasing the input size does not necessarily lead to better results and may, in fact, introduce complexity that the model struggles to handle effectively.

Non-CNN models, including the RNN, LSTM, GRU, and ViT ones, generally exhibit lower performance metrics compared to CNN-based models. The ViT model emerges as the most effective among the non-CNN models, with an accuracy of 35.39% and an F1 score of 34.71%, yet it still lags behind the top-performing CNN models. The GRU and LSTM models follow with accuracies of 23.70% and 22.20%, and F1 scores of 26.57% and 25.14%, respectively. The RNN model shows the weakest performance with an accuracy of 20.50% and an F1 score of 22.68%. These results highlight the inherent challenges non-CNN models face in generalizing from the training data to the more diverse and complex COP28 dataset.

When analyzing the detailed expression metrics from Table 13, we see that CNN-based models consistently reach higher F1 scores across most expressions compared to non-CNN models. For instance, the multi-path CNN excels in recognizing ‘afraid’ (an F1 score of 72%), ‘angry’ (70%), ‘disgusted’ (75%), ‘happy’ (85%), and ‘sad’ (76%) expressions. Similarly, the expanded CNN and VGG16 show strong performances across these expressions, demonstrating their robustness and reliability in handling a wide range of facial expressions. In particular, the expanded CNN achieves a notable F1 score of 79% for ‘neutral’ expressions, indicating its efficacy in identifying subtler facial features.

In contrast, non-CNN models display significant variability in their performance across different expressions. The ViT model, despite leading among non-CNN models, shows its highest F1 scores in ‘happy’ (54%) and ‘neutral’ (50%) expressions but struggles considerably with ‘surprised’ (18%) expressions. The GRU and LSTM models also demonstrate moderate performances, with their highest F1 scores in ‘disgusted’ (35% for GRU and 37% for LSTM) and ‘angry’ (26% for both), yet they exhibit poor recognition of the ‘neutral’ and ‘surprised’

expressions. This inconsistency across expressions highlights the limitations of non-CNN models in maintaining reliable performance for facial expression recognition.

Overall, the comparison underscores the superior performance of CNN-based models in facial expression recognition tasks on the COP28 dataset. Their higher accuracy and F1 scores indicate a more robust ability to capture and interpret the spatial features essential for distinguishing facial expressions. Non-CNN models, while capable of handling sequential data, struggle to achieve the same level of performance, particularly in terms of consistency across different expressions. This emphasizes the importance of leveraging advanced convolutional architectures for tasks involving complex image recognition and highlights the significant advantages these models offer in practical applications.

In the subsequent sections, a detailed analysis of each individual model will be conducted.

5.3 Analysis of the Standard CNN Model Results (Training and COP28)

The results obtained from training the standard CNN model with three different input sizes (64×64 , 128×128 , and 224×224) on the KDEP dataset and testing on the COP28 unseen images provide valuable insights into the impact of input image size on the model's performance. Analyzing these results helps to understand the strengths and limitations of the standard CNN architecture in the context of FER.

For the 64×64 input size, the model achieved a training accuracy of 89.9% and a validation accuracy of 79.4%. These results indicate that the model is capable of learning and

generalizing well to the validation set. However, the performance on the COP28 unseen images was notably lower, with an accuracy of 33.87% and an F1 score of 33.45%. This discrepancy suggests that while the model performs adequately on the KDEF dataset, it struggles to generalize to new data from the COP28 event. The precision and recall metrics further reveal that certain expressions, such as ‘happy’, were recognized more effectively than others like ‘angry’ and ‘surprised’, highlighting the model’s varying performance across different emotional expressions.

Increasing the input size to 128×128 pixels resulted in improved training accuracy, reaching 96.8%, and a validation accuracy of 78.2%. The precision, recall, and F1 scores for both training and validation were similarly high, indicating a robust model for the KDEF dataset. The COP28 test results showed a slight improvement over the 64×64 model, with an accuracy of 35.91% and an F1 score of 37.11%. The precision and recall for individual expressions were better balanced, though some expressions, such as ‘angry’ and ‘surprised’, still exhibited lower scores. This suggests that the model benefits from the higher resolution but still faces challenges with generalizing to new datasets.

When the input size was further increased to 224×224 pixels, the training accuracy slightly decreased to 93.2%, with a corresponding validation accuracy of 66.3%. This drop in validation performance could indicate overfitting, where the model learns the training data too well but fails to generalize effectively. This is reflected in the COP28 test results, where the accuracy was 30.88% and the F1 score was 29.43%. The precision and recall for most expressions were lower than those observed with the 128×128 input size, particularly for ‘neutral’ and ‘sad’, suggesting that the increased input size did not provide the expected benefits in this

context.

The observed performance across different input sizes can be critically evaluated in light of the model structure. The standard CNN model, with its three convolutional layers, max-pooling layers, and dropout layer, is relatively simple compared to more advanced architectures. This simplicity might limit its ability to capture complex facial features, especially in higher resolution images. Additionally, the final fully connected layers with only 128 neurons might be insufficient to fully leverage the increased information from larger input sizes.

The standard CNN model exhibits several strengths that make it a viable option for FER. One of the primary strengths is its relatively high accuracy on the KDEF dataset across different input sizes. This indicates that the model can effectively learn and recognize patterns in the training data. Additionally, the model's simplicity and straightforward architecture, with three convolutional layers followed by pooling and dropout layers, make it computationally efficient and easier to train compared to more complex models. The model also demonstrates good performance metrics, such as precision and recall, for certain expressions like 'disgusted' and 'happy', suggesting that it is capable of capturing and distinguishing some facial expressions with reasonable accuracy.

However, the model also has several weaknesses that limit its effectiveness in real-world applications. The most notable weakness is its poor generalization to the COP28 unseen images, where the accuracy and F1 scores significantly drop across all input sizes. This indicates that the model struggles to adapt to new and diverse datasets, which is critical for practical deployment. Additionally, the model's performance varies significantly across different

emotional expressions, with expressions like ‘angry’ and ‘surprised’ consistently showing lower precision, recall, and F1 scores. This inconsistency suggests that the model may have biases or limitations in its feature extraction capabilities. The observed overfitting, especially at higher resolutions, further highlights the need for a more robust training approach to prevent the model from learning the training data too well at the expense of generalizability.

To address these weaknesses, several enhancements can be suggested. First, incorporating more advanced architectures, such as transfer learning models like VGG16 or ResNet, could improve the model’s ability to generalize to unseen data by leveraging pre-trained weights on large, diverse datasets. Additionally, increasing the complexity of the model by adding more convolutional layers, increasing the number of neurons in the fully connected layers, or employing more several data augmentation techniques could help improve performance. Regularization techniques such as L2 regularization or dropout could also be adjusted to better manage overfitting. Moreover, fine-tuning the model on a more diverse dataset that includes a wider range of expressions and environmental conditions could help reduce biases and improve the recognition of less well-performing expressions. Lastly, experimenting with different input sizes and preprocessing methods could further optimize the model’s performance for specific applications like the COP28 event.

5.4 Analysis of the Expanded CNN Model Results (Training and COP28)

For the expanded CNN model, the training phase showed a robust performance with accuracy reaching a peak of 97.7% and an equally high F1 score, precision, and recall, all peaking at 97.7%. These metrics indicate that the model learned well from the training data,

achieving a low training loss of 1.192 from an initial value of 1.879. This high training performance suggests that the model is capable of capturing complex patterns in the facial expression images provided by the KDEF dataset.

In contrast, the validation results, while still respectable, highlight a notable performance drop compared to the training metrics. The validation accuracy peaked at 79.7%, with a corresponding F1 score, precision, and recall maxing out around the same value. This performance gap between training and validation indicates potential overfitting, where the model performs exceedingly well on the training data but struggles to generalize to unseen data. The validation loss of 1.373, while improved from its maximum of 1.811, further supports this observation, suggesting the need for techniques like regularization, data augmentation, or possibly simplifying the model architecture to improve generalization.

Evaluating the model on the COP28 test images provides additional insights into its real-world applicability. The model achieved an overall accuracy of 67.02% and an F1 score of 66.93%, indicating a moderate performance in a practical setting. A detailed breakdown of the precision, recall, and F1 scores for individual expressions reveals varying performance across different emotions. For instance, the model excelled in recognizing ‘happy’ expressions with an F1 score of 82% and ‘neutral’ expressions with a precision of 92%, yet struggled significantly with ‘sad’ and ‘surprised’ expressions, achieving F1 scores of 63% and 47% respectively. These discrepancies suggest that while the model can reliably identify certain expressions, it has difficulty with others, possibly due to insufficient representation of these expressions in the training data or inherent complexities in distinguishing them.

The expanded CNN model structure utilized here is fairly intricate, featuring multiple convolutional layers with batch normalization and max pooling, followed by fully connected layers with dropout for regularization. This complexity likely contributes to the model's strong performance on the training data but also to its overfitting issues. The model includes eight convolutional layers, which enable deep feature extraction but also increase the risk of overfitting if not adequately regularized or if trained on a relatively small dataset. The use of batch normalization helps mitigate this to some extent by normalizing the inputs to each layer, thereby stabilizing the learning process.

The expanded CNN model's strengths are evident in its high training performance and the sophisticated architecture that allows for detailed feature extraction. The multiple convolutional layers with batch normalization and max pooling stages ensure that the model captures intricate patterns in the facial expression images. This depth is crucial for tasks like FER, where subtle nuances in facial features must be detected and categorized accurately. The high training accuracy and F1 score demonstrate the model's capacity to learn from the data effectively. Moreover, the use of dropout layers in the fully connected stages helps prevent overfitting to some extent by introducing regularization, which is critical for maintaining model robustness.

However, the model also exhibits notable weaknesses that are primarily reflected in its validation and test performance. The significant drop in accuracy and other metrics from training to validation and test sets indicates overfitting, suggesting that while the model learns the training data well, it struggles to generalize to new, unseen data. The disparity in performance across different expressions, particularly for the 'sad' and 'surprised' expressions, points to

potential biases or imbalances in the training dataset. The high complexity of the model, with its numerous layers and parameters, may also contribute to this overfitting, making it prone to capturing noise and irrelevant patterns in the training data rather than generalizable features.

To enhance the performance and generalization capabilities of the expanded CNN model, several strategies can be employed. First, increasing the size and diversity of the training dataset can provide the model with more representative samples, particularly for underperforming expressions like ‘sad’ and ‘surprised’. Data augmentation techniques, such as salting and adding noise, can artificially expand the training dataset and help the model learn more robust features. Simplifying the model architecture by reducing the number of layers or parameters may also help strike a better balance between complexity and generalization. Finally, experimenting with different learning rates, optimization algorithms, and early stopping criteria can fine-tune the training process, ensuring the model achieves optimal performance across both the training and validation datasets.

5.5 Analysis of Multi-Path CNN Model Results (Training and COP28)

During the training phase, the multi-path CNN achieved a peak accuracy of 95.6%, with corresponding high values in F1 score 95.7%, precision 95.9%, and recall 95.4%. These metrics suggest that the model is highly effective at learning and distinguishing between different facial expressions within the training data. The low training loss of 0.122 further supports this conclusion, indicating that the model’s predictions align closely with the actual labels in the training set.

In the validation phase, the model maintained a strong performance with a peak accuracy of 89.6% and similarly high F1 score 89.1%, precision 89.3%, and recall 88.8%. The relatively small drop from training to validation performance indicates that the model generalizes well to unseen data, suggesting that overfitting is minimal. The validation loss of 0.363, while higher than the training loss, remains low, underscoring the model's robustness. These results imply that the multi-path CNN is not only capable of learning complex patterns in the training data but also effectively applies this learning to new data, which is crucial for real-world applications.

When evaluated on the COP28 test images, the model's overall accuracy was 71.70%, with an F1 score of 70.04%. This performance, while lower than the training and validation results, is still commendable given the likely differences in data distribution and environmental factors between the KDEF dataset and the COP28 images. A detailed examination of the model's performance across different expressions reveals significant variability. For instance, the model achieved high precision and recall for the 'happy' expressions at 82% and 88%, respectively, resulting in a strong F1 score of 85%. This indicates that the model is particularly adept at recognizing 'happy' expressions, likely because of the clear and consistent features associated with this emotion.

In contrast, the model struggled with 'surprised' expressions, achieving a precision of 94% but a much lower recall of 28%, leading to an F1 score of 43%. This discrepancy suggests that while the model is confident when it identifies 'surprised' expressions, it often fails to detect them, possibly due to the subtler or less consistent features. Similarly, the perfor-

mance regarding ‘neutral’ expressions was also less robust, with an F1 score of 55%, highlighting potential difficulties in distinguishing neutral expressions from others. These results indicate that while the multi-path CNN is effective in many scenarios, it may require further refinement to improve its sensitivity and accuracy across all expression types.

The multi-path CNN model’s architecture, which includes multiple paths for feature extraction and complex layers like two-path convolutions and MassAtt modules, plays a significant role in its performance. This design allows the model to capture a wide range of features at different scales and depths, contributing to its high training and validation performance. However, the complexity of the model also means that it may be more prone to overfitting and require extensive training data to achieve optimal generalization. The integration of point-wise feature selection (PWFS) layers and advanced attention mechanisms like MassAtt modules enhance the model’s ability to focus on relevant features while minimizing noise, which is crucial for accurate expression recognition.

The strengths of the multi-path CNN model are evident in its innovative architecture and the high performance achieved during training and validation on the KDEF dataset. The use of multiple paths for feature extraction allows the model to capture a diverse set of features, enhancing its ability to recognize complex patterns in facial expressions. The incorporation of advanced layers such as two-path convolutions and MassAtt modules improves the model’s attention to critical features, while the PWFS layers help in balancing the feature representation. These architectural choices contribute to the model’s high training accuracy and robust validation performance, demonstrating its potential for effective facial expression recognition.

However, the model also has notable weaknesses, particularly when applied to the COP28 test images. The drop in the accuracy and F1 score compared to the training and validation results suggests that the model struggles to generalize to new, unseen data. This issue is further highlighted by the significant variability in performance across different expressions. The model's difficulty in accurately recognizing 'surprised' and 'neutral' expressions indicates potential weaknesses in capturing the full variability of these emotions. Additionally, the complexity of the multi-path CNN, while a strength in terms of feature extraction, also poses challenges in terms of overfitting and the need for extensive computational resources and training data.

To enhance the performance of the multi-path CNN model, several strategies can be implemented. Augmenting the dataset with more diverse and representative samples, especially for underperforming expressions, can help the model learn more balanced features. Techniques such as synthetic data generation or oversampling can address class imbalances and improve the model's ability to generalize. Simplifying the model architecture by reducing the number of layers or parameters may also help mitigate overfitting while maintaining performance. Finally, experimenting with different optimization algorithms and learning rate schedules can optimize the training process, leading to better overall performance and stability across different datasets.

5.6 Transfer Learning VGG16 Model Results (Training and COP28)

The performance of the pre-trained transfer learning VGG16 model on the KDEF dataset demonstrates strong learning capabilities, especially when considering the complexity and subtlety involved in facial expression recognition. During the training phase, the model

achieved a peak accuracy of 93.8% and an F1 score of 98.9%, indicating a high level of precision and recall in distinguishing between different facial expressions. The training loss decreased significantly to 0.199, showing that the model effectively minimized errors and overfitting during the training process. This high training performance suggests that the VGG16 model, with its pre-trained layers, was able to leverage previously learned features effectively, resulting in robust learning outcomes.

In the validation phase, the VGG16 model continued to exhibit strong performance with a peak accuracy of 86.3% and an F1 score of 98.0%. The precision and recall metrics during validation were also impressive, with precision reaching a perfect score of 1.000 at its peak and recall closely following at 98.0%. The validation loss of 0.398, while higher than the training loss, indicates that the model maintains a reasonable generalization capability to unseen data. This high level of performance in both the training and validation phases underscores the effectiveness of transfer learning in enhancing the model's ability to generalize, as the pre-trained layers of VGG16 likely captured significant features from a large and diverse dataset prior to fine-tuning on the KDEF dataset.

When evaluated on the COP28 test images, the VGG16 model achieved an accuracy of 68.68% and an F1 score of 69.08%, reflecting a moderate drop in performance compared to the training and validation results. This discrepancy highlights some challenges in applying the model to real-world data, which might present different distribution and noise levels than the training dataset. A closer look at the performance across different expressions reveals considerable variability. For example, the model performed well on 'disgusted' expressions with

an F1 score of 77% and ‘afraid’ expressions with an F1 score of 67%, yet struggled with ‘neutral’ and ‘happy’ expressions, achieving lower F1 scores of 56% and 54%, respectively. The lower precision for ‘happy’ expressions at 42% and the significant recall gap for ‘neutral’ expressions at 41% suggest that the model finds it challenging to consistently identify these particular emotions, possibly due to the test set containing more subtle or varied features compared to the training set.

The VGG16 model’s architecture, with its depth and the pre-trained layers, provides a solid foundation for FER tasks. The pre-trained layers help in capturing essential features from the images, which are then fine-tuned on the specific dataset. However, the moderate performance on the COP28 test set indicates that while transfer learning significantly boosts performance, it also requires careful tuning and possibly additional data augmentation or regularization techniques to handle the variability and complexity of real-world data. The model’s ability to maintain high precision and recall in the training and validation phases while struggling with certain expressions in the test phase suggests a need for further enhancement in handling diverse data distributions and potentially addressing class imbalances or variations more effectively.

The strengths of the pre-trained transfer learning VGG16 model lie in its ability to leverage pre-trained layers to achieve a high performance on the KDEF dataset. The model’s architecture, known for its depth and robust feature extraction capabilities, allows it to capture intricate details in facial expressions. The high training and validation accuracies, along with the impressive F1 scores, precision, and recall, highlight the model’s effectiveness in learning

and distinguishing between various facial expressions. The use of transfer learning significantly reduces the training time and computational resources required, as the pre-trained layers provide a strong foundation for the model to build upon, leading to faster convergence and better performance with the target dataset.

However, the model exhibits weaknesses, particularly when applied to the COP28 test images. The drop in accuracy and the F1 score from the validation to the test phase indicates that the model struggles to generalize to new, unseen data with different characteristics. This issue is further exacerbated by the variability in performance across different expressions. The model's difficulty in accurately identifying 'neutral' and 'happy' expressions suggests potential limitations in capturing the full range of variability and subtlety in these emotions. Additionally, the lower recall for these expressions suggests that the model has a tendency to miss many true positives, which could be due to class imbalances or insufficient representation of these expressions in the training dataset.

To enhance the performance of the VGG16 model, several strategies can be employed. Increasing the size and diversity of the training dataset, particularly for underrepresented expressions, can aid the model in learning more balanced features and improve its generalization capabilities. Data augmentation techniques, such as random cropping, rotation, and color jittering, can artificially expand the dataset and help the model become more robust to variations in the test data.

5.7 RNN Model Results (Training and COP28)

The performance of the RNN model on the KDEF dataset and the COP28 test images

reveals significant insights into its capabilities and limitations in FER tasks. During the training phase, the RNN model achieved a peak accuracy of 60.9%, with an F1 score of 60% and precision at 59.8%. These metrics indicate a moderate level of learning from the training data. The training loss decreased from an initial 1.926 to a final value of 1.011, showing that the model is successfully minimizing the error over time. However, the relatively modest peak accuracy and F1 score suggest that the model may be struggling to capture the full complexity of facial expressions.

The validation results provide a more nuanced view of the model's performance. With a maximum accuracy of 51.2% and an F1 score of 50.4%, there is a noticeable drop from the training performance, indicating potential overfitting. The validation loss, which dropped from 1.888 to 1.318, although higher than the training loss, suggests that the model is generalizing to some extent but not optimally. The validation metrics indicate that while the RNN model is learning to recognize patterns in the training data, its ability to generalize to new, unseen data remains limited.

When applied to the COP28 test images, the RNN model's performance further declines, with an overall accuracy of 20.50% and an F1 score of 22.68%. The breakdown in its performance across different expressions highlights significant variability. For example, the model performed relatively better in recognizing 'afraid' and 'angry' expressions, with F1 scores of 25% for each, but struggled significantly with 'neutral' and 'surprised' expressions, achieving F1 scores of 3% and 9%, respectively. These results suggest that the RNN model's ability to generalize from the KDEP dataset to the more diverse and potentially more challenging COP28 images is inadequate.

The strengths of the RNN model lie in its architectural simplicity and its ability to process sequential data, which is crucial for tasks involving temporal dependencies. The use of two RNN layers allows the model to capture temporal patterns in the input data, which is beneficial for recognizing dynamic facial expressions. The model's moderate performance during training indicates that it can learn from the data, albeit with limited success. The inclusion of dropout layers helps to mitigate overfitting, although the validation and test results suggest that this is not entirely effective.

However, the model also exhibits significant weaknesses. The substantial drop in performance from the training phase to the validation and test ones indicates overfitting, where the model learns the training data well but fails to generalize. The variability in performance across different expressions suggests that the model is not robust in handling diverse facial expressions. The low performance on the COP28 dataset underscores the model's limited ability to adapt to new and potentially more complex data distributions. Additionally, the relatively modest peak accuracy and F1 scores during training suggest that the RNN model may not be sufficiently complex to capture the intricate features necessary for accurate facial expression recognition.

5.8 LSTM Model Results (Training and COP28)

The performance of the LSTM model on the KDEF dataset and the COP28 test images provides a detailed insight into its capabilities and limitations in the task of facial expression recognition. During the training phase, the LSTM model achieved a peak accuracy of 72.4%, with a corresponding F1 score of 71.9%. These results indicate that the model was able to effectively learn and distinguish between different facial expressions from the training data. The

training loss, which decreased from an initial value of 1.940 to 0.732, suggests a substantial improvement in the model's predictive accuracy over the training period. However, while these results demonstrate the model's capacity to learn, the relatively high training loss at the start also indicates that the initial learning process was challenging.

The validation phase results show a significant drop in performance compared to the training results. The validation accuracy peaked at 56.6%, with the F1 score reaching 55.9%. This decline from the training performance suggests that the model may have issues with overfitting, where it performs well on the training data but fails to generalize effectively to new, unseen data. The validation loss, while lower than its initial value of 1.924, remained relatively high at 1.263, further indicating that the model's generalization ability is not optimal. The consistency in the decrease of the loss and the increase in accuracy and F1 scores from training to validation suggests that while the model learns to recognize patterns, it does not do so uniformly across different datasets.

When applied to the COP28 test images, the LSTM model's performance was notably lower, with an overall accuracy of 22.20% and an F1 score of 25.14%. A detailed breakdown of the performance across different expressions highlights significant variability. For instance, the model performed relatively better in recognizing 'angry' expressions, achieving an F1 score of 28%, while it struggled significantly with 'surprised' and 'neutral' expressions, with F1 scores of 5% and 4%, respectively. These discrepancies indicate that the model's ability to generalize to new contexts, such as the COP28 dataset, is limited. The substantial drop in accuracy and F1 scores from the training and validation phases to the test phase underscores the

challenges the model faces in adapting to different data distributions and environmental factors.

The strengths of the LSTM model lie in its architecture, which is well suited for handling sequential data and capturing temporal dependencies. The use of two LSTM layers allows the model to learn and retain information over long sequences, which is beneficial for tasks involving dynamic and time-dependent patterns, such as facial expression recognition. The moderate performance during training indicates that the LSTM model can effectively capture and distinguish features within the training data. Additionally, the inclusion of dropout layers helps mitigate overfitting to some extent by providing regularization, which is crucial for maintaining model robustness.

However, the model exhibits several weaknesses, particularly in terms of generalization and performance variability across different expressions. The significant drop in accuracy and F1 scores from training to validation and further to the test phase indicates overfitting and a limited ability to generalize. The variability in performance across different expressions suggests that the model struggles to consistently recognize all types of facial expressions, potentially due to imbalances in the training data or the inherent complexity of certain expressions. The relatively modest peak accuracy and F1 scores during training also indicate that the LSTM model, despite its sequential learning capabilities, may not be sufficiently complex to capture all necessary features for accurate facial expression recognition.

5.9 GRU Model Results (Training and COP28)

During training, the GRU model achieved a peak accuracy of 69.4%, with a corresponding F1 score of 68.7%. This indicates that the model can learn and distinguish between different facial expressions to a considerable extent. The training loss decreased significantly from 1.925 to 0.810, reflecting the model's ability to reduce prediction error over the training period. However, despite these improvements, the performance metrics suggest that there is still room for further optimization.

In the validation phase, the GRU model's accuracy peaked at 55.4%, with an F1 score of 54.9%. This decline from the training performance suggests a degree of overfitting, where the model performs well on the training data but struggles to generalize to new data. The validation loss, which started at 1.884 and ended at 1.267, remained notably higher than the training loss, indicating that the model's predictive accuracy on unseen data is not as high as on the training set. These validation results highlight the model's limitations in maintaining a high performance across different datasets, which is crucial for practical applications.

When applied to the COP28 test images, the GRU model's performance further declined, achieving an overall accuracy of 23.7% and an F1 score of 26.57%. The breakdown in its performance across different expressions showed significant variability. For example, the model performed relatively well in recognizing 'disgusted' expressions, with an F1 score of 35%, but struggled with 'neutral' and 'surprised' expressions, achieving F1 scores of 4% and 9%, respectively. This inconsistency indicates that while the model can identify certain expressions reasonably well, it has difficulty with others, particularly in diverse and challenging real-world scenarios.

The strengths of the GRU model lie in its ability to handle sequential data and capture temporal dependencies, which are essential for dynamic and time-dependent tasks like facial expression recognition. The architecture, featuring two GRU layers, allows the model to learn long-range dependencies and patterns in the input data. The model's performance during training suggests that it can effectively capture and distinguish features within the training set, and this is supported by the significant reduction in training loss. Additionally, the inclusion of dropout layers provides regularization, which helps to mitigate overfitting to some extent.

However, the model exhibits several weaknesses, primarily in its generalization capability and variability in performance across different expressions. The significant drop in accuracy and F1 scores from the training phase to the validation and test ones indicates that the model struggles to generalize beyond the training data. The variability in performance across different expressions suggests that the model is not robust in handling diverse facial expressions, which could be due to imbalances in the training data or the inherent complexity of certain expressions. The relatively modest peak accuracy and F1 scores, particularly on the COP28 test set, further highlight the model's limitations in adapting to new and varied data distributions. These weaknesses underscore the challenges of achieving consistent and reliable performance in real-world facial expression recognition tasks using the GRU model.

5.10 ViT Model Results (Training and COP28)

The ViT model's performance on the KDEF dataset and the COP28 test images highlights its capabilities as well as areas where it falls short in facial expression recognition tasks. During training on the KDEF dataset, the ViT model achieved a peak accuracy of 81.9%, with a corresponding F1 score of 81.8%. This indicates a strong learning capability, as the model

effectively captured and distinguished between different facial expressions. The training loss reduced significantly from an initial 1.957 to 0.499, demonstrating the model's efficiency in minimizing prediction errors and learning the intricate patterns in the data. These high training metrics underscore the model's potential for handling complex image recognition tasks when sufficient training data is available.

The validation phase, however, reveals a stark contrast. The validation accuracy peaked at 55.5%, with an F1 score of 55.2%. This drop from the training performance suggests that the ViT model may be overfitting, that is, performing well on the training data but struggling to generalize to new, unseen data. The validation loss, which only slightly decreased to 1.368 from an initial 2.012, further indicates that the model's predictions using validation data are less accurate. These results highlight the challenges the ViT model faces in maintaining a high performance across different datasets, a crucial aspect for real-world applications.

When evaluated on the COP28 test images, the ViT model's performance further declined, achieving an overall accuracy of 35.39% and an F1 score of 34.71%. The breakdown in its performance across different expressions showed significant variability. For example, the model performed reasonably well in recognizing 'happy' expressions, with an F1 score of 54%, but struggled considerably with 'surprised' expressions, achieving an F1 score of only 18%. This variability suggests that while the ViT model can identify some expressions with moderate success, it has difficulty with others, particularly in diverse and challenging real-world scenarios. The model's lower performance on the COP28 dataset underscores its limited ability to adapt to new and varied data distributions.

The strengths of the ViT model lie in its architecture, which leverages the power of transformer networks to capture long-range dependencies and intricate patterns in the data. The model's high training performance indicates its potential to learn complex features from the input images. The use of patch embeddings and positional encoding allows the model to handle spatial information effectively, which is crucial for tasks like facial expression recognition. Additionally, the ViT model's design, which includes multiple transformer layers, enhances its ability to process and interpret detailed features from the data, contributing to its strong performance during training.

However, the model exhibits significant weaknesses, particularly in its generalization capability and performance variability across different expressions. The substantial drop in accuracy and F1 scores from the training phase to the validation and test ones indicates that the model struggles to generalize beyond the training data. The variability in performance across different expressions suggests that the model is not robust in handling diverse facial expressions, which could be due to imbalances in the training data or the inherent complexity of certain expressions. The relatively modest peak accuracy and F1 scores on the COP28 test set further highlight the model's limitations in adapting to new and varied data distributions. These weaknesses underscore the challenges of achieving consistent and reliable performance in real-world facial expression recognition tasks using the ViT model.

Chapter 6: Conclusion, Limitations, and Future Works

6.1 Summary of Findings

This study undertook a comprehensive evaluation of various deep learning models for facial expression recognition (FER) in the context of COP28, a large-scale public event. The models examined included standard CNN, expanded CNN, multi-path CNN, pre-trained transfer learning VGG16, RNN, LSTM, GRU, and vision transformer (ViT). The work utilized the KDEF dataset for training and validation and assessed model performance using unseen COP28 test images.

The standard CNN models demonstrated moderate performance across different image resolutions (64×64 , 128×128 , and 224×224 pixels). The model with 128×128 resolution achieved the highest training accuracy (96.8%) but struggled with generalization, showing a significant drop in validation and test performance. The expanded CNN model outperformed the standard CNN models, achieving a training accuracy of 97.7%, one of the highest on the KDEF dataset. However, the validation accuracy dropped to 79.7%, and the COP28 test accuracy was 67.02%. This model exhibited a strong ability to learn from the training data, but its generalization to new, unseen data was less robust.

The multi-path CNN model exhibited the highest performance among the tested models, with a validation accuracy of 89.6% and a COP28 test accuracy of 71.70%. Its sophisticated architecture, which includes multiple paths for feature extraction and attention mechanisms, contributed to its superior performance. The VGG16 model, leveraging transfer learning, achieved high training and validation performance with accuracies of 93.8% and 86.3%, respectively. However, its performance on COP28 test images was slightly lower (68.68% accuracy), indicating challenges in adapting to new data distributions.

Sequential models such as RNN, LSTM, and GRU exhibited lower performance compared to CNN-based models. The LSTM and GRU models performed slightly better than the RNN but still fell short in generalization, with COP28 test accuracies of 22.20% and 23.70%, respectively. The ViT model showed strong training performance (81.9% accuracy) but struggled with generalization, achieving a COP28 test accuracy of 35.39%. This suggests that while the ViT can capture detailed features, it may require more extensive training data or different regularization techniques to perform well in diverse real-world scenarios.

6.2 Study Limitations

Despite the comprehensive analysis, several limitations were identified. First, the study primarily utilized the KDEF dataset for training, which may not fully represent the diversity and complexity of facial expressions encountered in real-world scenarios like COP28. This limitation likely contributed to the models' varying performance on the COP28 test images. Second, most models exhibited a significant drop in performance when applied to the COP28 test images, highlighting challenges in generalization. This suggests that the models may have overfitted the training data and struggled to adapt to new and unseen data distributions.

Additionally, the models showed inconsistent performance across different expressions, with certain expressions like 'happy' and 'neutral' being recognized more accurately than others like 'surprised' and 'sad'. This indicates potential imbalances in the training data or inherent difficulties in distinguishing certain expressions. Furthermore, the study did not extensively explore hyperparameter tuning or the use of more computationally intensive models due to resource constraints. More extensive experimentation with different hyperparameters and model configurations could potentially improve performance.

6.3 Addressing the Research Questions

RQ1. What are the strengths and limitations of deep learning models trained on controlled datasets when applied to real-world images from events like COP28?

The study found that while models trained on controlled datasets like KDEF can achieve high accuracy and a robust performance during training and validation, their performance significantly drops when applied to real-world images from events like COP28. This discrepancy highlights the models' limited generalization capabilities. For instance, the multi-path CNN, which showed the best performance, still exhibited a notable decrease in accuracy when tested on COP28 images compared to its validation performance. This indicates that the controlled conditions of datasets like KDEF do not fully capture the variability and complexity of real-world environments, leading to challenges in adapting to diverse and dynamic contexts.

RQ2. How do different image resolutions (64x64, 128x128, and 224x224 pixels) impact the performance of CNN models in FER on controlled datasets like KDEF?

The study demonstrated that different image resolutions significantly impact the performance of CNN models in FER. The standard CNN model with 128×128 pixel images achieved the highest training accuracy, suggesting that this resolution provides a balanced level of detail that enhances model learning. However, increasing the resolution to 224×224 pixels did not yield better results and even led to decreased performance, likely due to the increased complexity and potential overfitting. Conversely, lower-resolution images (64×64 pixels) did not provide sufficient detail, resulting in lower performance. Therefore, 128×128 pixels emerged as the optimal resolution for training CNN models on the KDEF dataset.

RQ3. How does transfer learning with VGG16 compare to training from scratch in terms of accuracy and training time for FER?

The VGG16 model leveraging transfer learning demonstrated superior performance compared to models trained from scratch. The VGG16 achieved high training and validation accuracy, indicating effective feature extraction and generalization capabilities. Additionally, transfer learning significantly reduced the training time, as the model utilized pre-trained weights from a large, diverse dataset, requiring only fine-tuning for the specific FER task. This approach not only improved accuracy but also enhanced training efficiency, making transfer learning with VGG16 a valuable strategy for FER applications.

RQ4. What are the comparative performances of non-CNN models (RNN, LSTM, GRU, and ViT) versus CNN-based models in FER, particularly in the unique setting of events like COP28?

The study revealed that CNN-based models consistently outperformed non-CNN models in FER, particularly in the context of events like COP28. While RNN, LSTM, GRU, and ViT models exhibited moderate training performance, their generalization to the COP28 test images was limited, resulting in significantly lower accuracies. The multi-path CNN and VGG16 models, both CNN-based, showed the highest test accuracies, indicating their superior capability in capturing and interpreting facial expressions. The results suggest that CNN architectures are better suited for FER tasks due to their ability to effectively process spatial information in images.

6.4 Future Research Directions

To address the identified limitations and build upon the findings of this study, several future research directions are proposed. Subsequent works should incorporate more diverse and extensive datasets that capture a wider range of facial expressions and environmental conditions. This will help improve the models' ability to generalize to real-world scenarios. Implementing more sophisticated data augmentation techniques, such as generative adversarial networks (GANs) for synthetic data generation, can help create more balanced and representative training datasets.

Exploring ensemble techniques, where multiple models are combined to improve overall performance, could enhance the robustness and accuracy of FER systems. Further investigation into transfer learning, including fine-tuning pre-trained models on larger and more diverse datasets, can help improve model generalization and performance on new data. Conducting extensive hyperparameter optimization using techniques like grid search or Bayesian optimization can help identify the best model configurations and improve performance.

Future studies should focus on deploying FER models in real-world settings and continuously evaluating their performance. This will provide valuable insights into the practical challenges and effectiveness of these models in dynamic environments. Additionally, investigating newer and more advanced deep learning architectures, such as hybrid models combining CNNs and transformers or incorporating graph neural networks, could lead to significant improvements in FER capabilities.

It is also essential to explore the potential of multimodal approaches, combining visual

data with other modalities like audio or text, to enhance the recognition of facial expressions. Developing techniques to handle occlusions and varying lighting conditions more effectively can further improve the robustness of FER systems. Additionally, integrating explainability into FER models to understand their decision-making processes can build trust and transparency in their applications.

Furthermore, future research could investigate the impact of cultural and demographic factors on facial expression recognition. Understanding how different cultures express emotions and how demographic variations affect FER performance can lead to the development of more inclusive and accurate models. Finally, the ethical implications of FER technology should be considered, ensuring that the models are developed and deployed responsibly, respecting privacy and avoiding potential biases.

6.5 Final Remarks

This study provides a thorough comparative analysis of various deep learning models for facial expression recognition in the context of a large-scale public event like COP28. While the multi-path CNN and VGG16 models showed promising results, the challenges in generalization and expression recognition variability highlight the need for further research and development. By addressing the identified limitations and exploring the proposed future research directions, significant advancements in FER technology can be achieved, enhancing its applicability and reliability in real-world scenarios.

References

- Aguilera, A., Mellado, D., & Rojas, F. (2023). An Assessment of In-the-Wild Datasets for Multimodal Emotion Recognition. *Sensors*, 23(11), 5184.
<https://doi.org/10.3390/s23115184>
- Ahmed, F., & Hossain, E. (2013). Automated Facial Expression Recognition Using Gradient-Based Ternary Texture Patterns. *Chinese Journal of Engineering*, 2013, 1–8.
<https://doi.org/10.1155/2013/831747>
- Aifanti, N., Papachristou, C., & Delopoulos, A. (2010). The MUG Facial Expression Database. *11th International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, 10, 1–4.
- Akhand, M. A. H., Roy, S., Siddique, N., Kamal, M. A. S., & Shimamura, T. (2021). Facial Emotion Recognition Using Transfer Learning in the Deep CNN. *Electronics*, 10(9), 1036. <https://doi.org/10.3390/electronics10091036>
- Alharbi, S. A. H., Button, K., Zhang, L., O'Shea, K. J., Fasolt, V., Lee, A. J., DeBruine, L. M., & Jones, B. C. (2020). Are Affective Factors Related to Individual Differences in Facial Expression Recognition? *Royal Society Open Science*, 7(9), 190699.
<https://doi.org/10.1098/rsos.190699>
- Aneja, D., Colburn, A., Faigin, G., Shapiro, L., & Mones, B. (2017). Modeling Stylized Character Expressions via Deep Learning. *Computer Vision – ACCV 2016*, 136–153.
https://doi.org/10.1007/978-3-319-54184-6_9
- Ansel, J., Yang, E., He, H., Gimelshein, N., Jain, A., Voznesensky, M., Bao, B., Bell, P., Berard, D., Burovski, E., Chauhan, G., Chourdia, A., Constable, W., Desmaison, A.,

DeVito, Z., Ellison, E., Feng, W., Gong, J., Gschwind, M., Chintala, S. (2024).

PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation. *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, 2, 929–947. <https://doi.org/10.1145/3620665.3640366>

Appasaheb Borgalli, R., & Surve, S. (2022). Deep Learning Framework for Facial Emotion Recognition Using CNN Architectures. *2022 International Conference on Electronics and Renewable Systems (ICEARS)*, 1777–1784.

<https://doi.org/10.1109/ICEARS53579.2022.9751735>

Bazarevsky, V., Kartynnik, Y., Vakunov, A., Raveendran, K., & Grundmann, M. (2019).

BlazeFace: Sub-Millisecond Neural Face Detection on Mobile GPUs. *ArXiv*.

<https://arxiv.org/abs/1907.05047>

Belhumeur, P. N., Hespanha, J. P., & Kriegman, D. J. (1996). Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection. In B. Buxton & R. Cipolla (Eds.), *Computer Vision—ECCV '96* (pp. 43–58). Springer Berlin Heidelberg.

<https://doi.org/10.1007/BFb0015522>

Bhagat, D., Vakil, A., Gupta, R. K., & Kumar, A. (2024). Facial Emotion Recognition (FER) Using Convolutional Neural Network (CNN). *Procedia Computer Science*, 235,

2079–2089. <https://doi.org/10.1016/j.procs.2024.04.197>

Bui, K.-H. N., Oh, H., & Yi, H. (2020). Traffic Density Classification Using Sound Datasets:

An Empirical Study on Traffic Flow at Asymmetric Roads. *IEEE Access*, 8,

125671–125679. <https://doi.org/10.1109/ACCESS.2020.3007917>

- Canedo, D., & Neves, A. J. R. (2019). Facial Expression Recognition Using Computer Vision: A Systematic Review. *Applied Sciences*, 9(21), 4678.
<https://doi.org/10.3390/app9214678>
- Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation. *ArXiv*. <https://arxiv.org/abs/1406.1078>
- Chowdary, M. K., Nguyen, T. N., & Hemanth, D. J. (2023). Deep Learning-Based Facial Emotion Recognition for Human–Computer Interaction Applications. *Neural Computing and Applications*, 35(32), 23311–23328. <https://doi.org/10.1007/s00521-021-06012-8>
- Cortes, C., & Vapnik, V. (1995). Support-Vector Networks. *Machine Learning*, 20(3), 273–297. <https://doi.org/10.1007/BF00994018>
- Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1), 21–27. <https://doi.org/10.1109/TIT.1967.1053964>
- Dalal, N., & Triggs, B. (n.d.). Histograms of Oriented Gradients for Human Detection. 2005 *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, 886–893. <https://doi.org/10.1109/CVPR.2005.177>
- Das, S., Tariq, A., Santos, T., Kantareddy, S. S., & Banerjee, I. (2023). Recurrent Neural Networks (RNNs): Architectures, Training Tricks, and Introduction to Influential Research. In *Machine Learning for Brain Disorders* (pp. 117–138).
https://doi.org/10.1007/978-1-0716-3195-9_4
- Dhall, A., Goecke, R., Lucey, S., & Gedeon, T. (2011). Static Facial Expression Analysis in

Tough Conditions: Data, Evaluation Protocol and Benchmark. *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, 2106–2112.

<https://doi.org/10.1109/ICCVW.2011.6130508>

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., & Darrell, T. (2014).

DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition.

In E. P. Xing & T. Jebara (Eds.), *Proceedings of the 31st International Conference on Machine Learning*, 32(1), 647–655. [https://proceedings.mlr.press/v32/do-](https://proceedings.mlr.press/v32/donahue14.html)

[nahue14.html](https://proceedings.mlr.press/v32/donahue14.html)

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T.,

Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Hounsby, N.

(2020). An Image is Worth 16x16 Words: Transformers for Image Recognition at

Scale. *ArXiv*. <https://arxiv.org/abs/2010.11929>

Du, S., Tao, Y., & Martinez, A. M. (2014). Compound Facial Expressions of Emotion. *Proceedings of the National Academy of Sciences*, 111(15).

<https://doi.org/10.1073/pnas.1322355111>

Ezati, A., Dezyani, M., Rana, R., Rajabi, R., & Ayatollahi, A. (2024). A Lightweight Atten-

tion-Based Deep Network via Multi-Scale Feature Fusion for Multi-View Facial Ex-

pression Recognition. *ArXiv*. <https://arxiv.org/pdf/2403.14318>

Gautam, C., & Seeja, K. R. (2023). Facial Emotion Recognition Using Handcrafted features and CNN. *Procedia Computer Science*, 218, 1295–1303.

<https://doi.org/10.1016/j.procs.2023.01.108>

Goodfellow, I. J., Erhan, D., Carrier, P. L., Courville, A., Mirza, M., Hamner, B., Cukierski,

W., Tang, Y., Thaler, D., Lee, D.-H., Zhou, Y., Ramaiah, C., Feng, F., Li, R., Wang,

- X., Athanasakis, D., Shawe-Taylor, J., Milakov, M., Park, J., Bengio, Y. (2013). Challenges in Representation Learning: A Report on Three Machine Learning Contests. In M. Lee, A. Hirose, Z.-G. Hou, & R. M. Kil (Eds.), *Neural Information Processing* (pp. 117–124). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-42051-1_16
- Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative Adversarial Networks. *ArXiv*. <https://arxiv.org/abs/1406.2661>
- Graves, A., Jaitly, N., & Mohamed, A. (2013). Hybrid Speech Recognition with Deep Bidirectional LSTM. *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, 273–278. <https://doi.org/10.1109/ASRU.2013.6707742>
- Hashemi, M. (2019). Enlarging Smaller Images Before Inputting into Convolutional Neural Network: Zero-Padding vs. Interpolation. *Journal of Big Data*, 6(1), 98. <https://doi.org/10.1186/s40537-019-0263-7>
- He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. *ArXiv*. <https://arxiv.org/abs/1512.03385>
- Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, 18(7), 1527–1554. <https://doi.org/10.1162/neco.2006.18.7.1527>
- Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), 1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>
- Holder, R. P., & Tapamo, J.-R. (2018). Using Facial Expression Recognition for Crowd Monitoring. *Image and Video Technology*, 463–476. <https://doi.org/10.1007/978-3->

[319-75786-5](#) 37

- Howard, A., Zhmoginov, A., Chen, L.-C., Sandler, M., & Zhu, M. (2018). Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation. *CVPR*.
- Hsiao, T.-Y., Chang, Y.-C., Chou, H.-H., & Chiu, C.-T. (2019). Filter-Based Deep-Compression with Global Average Pooling for Convolutional Networks. *Journal of Systems Architecture*, 95, 9–18. <https://doi.org/10.1016/j.sysarc.2019.02.008>
- Hu, J., Shen, L., Albanie, S., Sun, G., & Wu, E. (2017). Squeeze-and-Excitation Networks. *ArXiv*. <https://arxiv.org/abs/1709.01507>
- Hu, Z. (2023). A CNN-Based Facial Expression Recognition System. *Highlights in Science, Engineering and Technology*, 39, 496–507. <https://doi.org/10.54097/hset.v39i.6576>
- Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2261–2269. <https://doi.org/10.1109/CVPR.2017.243>
- Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., & Darrell, T. (2014). Caffe: Convolutional Architecture for Fast Feature Embedding. *Proceedings of the 22nd ACM International Conference on Multimedia*, 675–678. <https://doi.org/10.1145/2647868.2654889>
- Kahou, S. E., Pal, C., Bouthillier, X., Froumenty, P., Gülçehre, Ç., Memisevic, R., Vincent, P., Courville, A., Bengio, Y., Ferrari, R. C., Mirza, M., Jean, S., Carrier, P.-L., Dauphin, Y., Boulanger-Lewandowski, N., Aggarwal, A., Zumer, J., Lamblin, P., Raymond, J.-P., Wu, Z. (2013). Combining Modality Specific Deep Neural Networks for

Emotion Recognition in Video. *Proceedings of the 15th ACM on International Conference on Multimodal Interaction*, 543–550.

<https://doi.org/10.1145/2522848.2531745>

Khaireddin, Y., & Chen, Z. (2021). Facial Emotion Recognition: State of the Art Performance on FER2013. *ArXiv*. <https://arxiv.org/abs/2105.03588>

Khanzada, A., Bai, C., & Celepcikay, F. T. (2020). Facial Expression Recognition with Deep Learning. *ArXiv*. <https://arxiv.org/abs/2004.11823>

Khorsheed, A. A. (2022). Intelligent Diagnosis of Covid-19 Based on CNN-PNN. *Journal La Multiapp*, 3(4), 174–197. <https://doi.org/10.37899/journallamultiapp.v3i4.691>

Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. *ArXiv*. <https://arxiv.org/abs/1412.6980>

Kopalidis, T., Solachidis, V., Vretos, N., & Daras, P. (2024). Advances in Facial Expression Recognition: A Survey of Methods, Benchmarks, Models, and Datasets. *Information*, 15(3), 135. <https://doi.org/10.3390/info15030135>

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet Classification with Deep Convolutional Neural Networks. *Communications of the ACM*, 60(6), 84–90. <https://doi.org/10.1145/3065386>

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1(4), 541–551. <https://doi.org/10.1162/neco.1989.1.4.541>

Li, S., & Deng, W. (2019). Reliable Crowdsourcing and Deep Locality—Preserving Learning for Unconstrained Facial Expression Recognition. *IEEE Transactions on Image Processing*, 28(1), 356–370.

Liu, K., Zhang, M., & Pan, Z. (2016). Facial Expression Recognition with CNN Ensemble.

2016 International Conference on Cyberworlds (CW), 163–166.

<https://doi.org/10.1109/CW.2016.34>

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016).

SSD: Single Shot MultiBox Detector. *Computer Vision—ECCV 2016*, 21–37.

https://doi.org/10.1007/978-3-319-46448-0_2

Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2), 91–110.

<https://doi.org/10.1023/B:VISI.0000029664.99615.94>

Lucey, P., Cohn, J. F., Kanade, T., Saragih, J., Ambadar, Z., & Matthews, I. (2010). The Ex-

tended Cohn-Kanade Dataset (CK+): A Complete Dataset for Action Unit and Emotion-Specified Expression. *2010 IEEE Computer Society Conference on Computer*

Vision and Pattern Recognition – Workshops, 94–101.

<https://doi.org/10.1109/CVPRW.2010.5543262>

Lundqvist, D. (1998). *About KDEF*. Department of Clinical Neuroscience.

<https://kdef.se/home/aboutKDEF>

Lundqvist, D., Flykt, A., & Öhman, A. (1998). *The Karolinska Directed Emotional Faces –*

KDEF. Department of Clinical Neuroscience, Psychology Section, Karolinska Insti-

tutet. <https://kdef.se/faq/using-and-publishing-kdef-and-akdef>

Lyons, M., Akamatsu, S., Kamachi, M., & Gyoba, J. (1998). Coding Facial Expressions with

Gabor Wavelets. *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, 200–205.

<https://doi.org/10.1109/AFGR.1998.670949>

- Majeed, S. H., & Isa, N. A. M. (2020). Iterated Adaptive Entropy-Clip Limit Histogram Equalization for Poor Contrast Images. *IEEE Access*, 8, 144218–144245.
<https://doi.org/10.1109/ACCESS.2020.3014453>
- Martinez, B., & Valstar, M. F. (2016). Advances, Challenges, and Opportunities in Automatic Facial Expression Recognition. In *Advances in Face Detection and Facial Image Analysis* (pp. 63–100). Springer International Publishing.
https://doi.org/10.1007/978-3-319-25958-1_4
- Mathew, S. S., AlKhatib, M., & El Barachi, M. (2021). A Deep Learning Approach for Real-Time Analysis of Attendees’ Engagement in Public Events. *Journal of Communications Software and Systems*, 17(2), 106–115. <https://doi.org/10.24138/jcomss-2021-0072>
- McDuff, D., El Kaliouby, R., Senechal, T., Amr, M., Cohn, J. F., & Picard, R. (2013). Affective-MIT Facial Expression Dataset (AM-FED): Naturalistic and Spontaneous Facial Expressions Collected “In-the-Wild.” *2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 881–888.
<https://doi.org/10.1109/CVPRW.2013.130>
- Mejia-Escobar, C., Cazorla, M., & Martinez-Martin, E. (2023). Improving Facial Expression Recognition Through Data Preparation and Merging. *IEEE Access*, 11, 71339–71360. <https://doi.org/10.1109/ACCESS.2023.3293728>
- Minaee, S., Minaei, M., & Abdolrashidi, A. (2021). Deep-Emotion: Facial Expression Recognition Using Attentional Convolutional Network. *Sensors*, 21(9), 3046.
<https://doi.org/10.3390/s21093046>

- Mohana, M., & Subashini, P. (2024). Facial Expression Recognition Using Machine Learning and Deep Learning Techniques: A Systematic Review. *SN Computer Science*, 5(4), 432. <https://doi.org/10.1007/s42979-024-02792-7>
- Mollahosseini, A., Hasani, B., & Mahoor, M. H. (2019). AffectNet: A Database for Facial Expression, Valence, and Arousal Computing in the Wild. *IEEE Transactions on Affective Computing*, 10(1), 18–31. <https://doi.org/10.1109/TAFFC.2017.2740923>
- Nan, F., Jing, W., Tian, F., Zhang, J., Chao, K.-M., Hong, Z., & Zheng, Q. (2022). Feature Super-Resolution Based Facial Expression Recognition for Multi-Scale Low-Resolution Images. *Knowledge-Based Systems*, 236, 107678. <https://doi.org/10.1016/j.knosys.2021.107678>
- Naser, O. A., Syed Ahmad, S. M., Samsudin, K., & Hanafi, M. (2023). Investigating the Impact of Yaw Pose Variation on Facial Recognition Performance. *Advances in Artificial Intelligence and Machine Learning*, 3(2), 1039–1055. <https://doi.org/10.54364/AAIML.2023.1162>
- Noh, S.-H. (2021). Analysis of Gradient Vanishing of RNNs and Performance Comparison. *Information*, 12(11), 442. <https://doi.org/10.3390/info12110442>
- Ogundokun, R. O., Maskeliunas, R., Misra, S., & Damaševičius, R. (2022). Improved CNN Based on Batch Normalization and Adam Optimizer. *Computational Science and Its Applications – ICCSA 2022 Workshops*, 593–604. https://doi.org/10.1007/978-3-031-10548-7_43
- Ojala, T., Pietikainen, M., & Maenpaa, T. (2002). Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7), 971–987.

<https://doi.org/10.1109/TPAMI.2002.1017623>

Olah, C. (2015). *Understanding LSTM Networks*. GitHub. [https://colah.github.io/posts/2015-](https://colah.github.io/posts/2015-08-Understanding-LSTMs/)

[08-Understanding-LSTMs/](https://colah.github.io/posts/2015-08-Understanding-LSTMs/)

Pal, R., & Saraswat, M. (2018). Enhanced Bag of Features Using AlexNet and Improved Biogeography-Based Optimization for Histopathological Image Analysis. *2018 Eleventh International Conference on Contemporary Computing (IC3)*, 1–6.

<https://doi.org/10.1109/IC3.2018.8530540>

Pizer, S. M., Amburn, E. P., Austin, J. D., Cromartie, R., Geselowitz, A., Greer, T., Ter Haar

Romeny, B., Zimmerman, J. B., & Zuiderveld, K. (1987). Adaptive Histogram

Equalization and Its Variations. *Computer Vision, Graphics, and Image Processing*,

39(3), 355–368. [https://doi.org/10.1016/S0734-189X\(87\)80186-X](https://doi.org/10.1016/S0734-189X(87)80186-X)

Puthanidam, R. V., & Moh, T.-S. (2018). A Hybrid Approach for Facial Expression Recognition. *Proceedings of the 12th International Conference on Ubiquitous Information*

Management and Communication, 1–8. <https://doi.org/10.1145/3164541.3164593>

Rachapudi, V., & Lavanya Devi, G. (2021). Improved Convolutional Neural Network Based Histopathological Image Classification. *Evolutionary Intelligence*, 14(3), 1337–

1343. <https://doi.org/10.1007/s12065-020-00367-y>

Rani, R., Arora, S., Verma, V., Mahajan, S., & Sharma, R. (2024). Enhancing Facial Expression Recognition Through Generative Adversarial Networks-Based Augmentation.

International Journal of System Assurance Engineering and Management, 15(3),

1037–1056. <https://doi.org/10.1007/s13198-023-02186-7>

Rescigno, M., Spezialetti, M., & Rossi, S. (2020). Personalized Models for Facial Emotion

Recognition Through Transfer Learning. *Multimedia Tools and Applications*, 79(47–

- 48), 35811–35828. <https://doi.org/10.1007/s11042-020-09405-4>
- Rguibi, Z., Hajami, A., Zitouni, D., Elqaraoui, A., & Bedraoui, A. (2022). CXAI: Explaining Convolutional Neural Networks for Medical Imaging Diagnostic. *Electronics*, 11(11), 1775. <https://doi.org/10.3390/electronics11111775>
- Ruan, B.-K., Shuai, H.-H., & Cheng, W.-H. (2022). Vision Transformers: State of the Art and Research Challenges. *ArXiv*. <https://arxiv.org/abs/2207.03041>
- Sak, H., Senior, A., & Beaufays, F. (2014). Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition. *ArXiv*. <https://arxiv.org/abs/1402.1128>
- Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2018). MobileNetV2: Inverted Residuals and Linear Bottlenecks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4510–4520. <https://doi.org/10.1109/CVPR.2018.00474>
- Santoso, B. E., & Kusuma, G. P. (2022). Facial Emotion Recognition on FER2013 Using VGGSPINALNET. *Journal of Theoretical and Applied Information Technology*, 100(7), 2008–2102.
- Shewalkar, A. (2018). *Comparison of RNN, LSTM and GRU on Speech Recognition Data*. North Dakota State University. <https://api.semanticscholar.org/CorpusID:88483286>
- Shewalkar, A., Nyavanandi, D., & Ludwig, S. A. (2019). Performance Evaluation of Deep Neural Networks Applied to Speech Recognition: RNN, LSTM and GRU. *Journal of Artificial Intelligence and Soft Computing Research*, 9(4), 235–245. <https://doi.org/10.2478/jaiscr-2019-0006>
- Siarni-Namini, S., Tavakoli, N., & Namin, A. S. (2019). The Performance of LSTM and

- BiLSTM in Forecasting Time Series. *2019 IEEE International Conference on Big Data (Big Data)*, 3285–3292. <https://doi.org/10.1109/BigData47090.2019.9005997>
- Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. *ArXiv*. <https://arxiv.org/abs/1409.1556>
- Sonali, Sahu, S., Singh, A. K., Ghrera, S. P., & Elhoseny, M. (2019). An Approach for De-Noising and Contrast Enhancement of Retinal Fundus Image Using CLAHE. *Optics & Laser Technology*, 110, 87–98. <https://doi.org/10.1016/j.optlastec.2018.06.061>
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Drop-out: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56), 1929–1958. <http://jmlr.org/papers/v15/srivastava14a.html>
- Stratou, G., Ghosh, A., Debevec, P., & Morency, L.-P. (2011). Effect of Illumination on Automatic Expression Recognition: A Novel 3D Relightable Facial Database. *Face and Gesture 2011*, 611–618. <https://doi.org/10.1109/FG.2011.5771467>
- Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. (2017). Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 31(1). <https://doi.org/10.1609/aaai.v31i1.11231>
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2014). Going Deeper with Convolutions. *ArXiv*. <https://arxiv.org/abs/1409.4842>
- Tan, M., & Le, Q. V. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *ArXiv*. <https://arxiv.org/abs/1905.11946>
- Tang, Y. (2013). Deep Learning using Linear Support Vector Machines. *ArXiv*.

<https://arxiv.org/abs/1306.0239>

Vaidya, S. (2020). *Natural Human Face Images for Emotion Recognition*. Kaggle.

<https://www.kaggle.com/datasets/sudarshanvaidya/random-images-for-face-emotion-recognition>

Vaswani, A., Brain, G., 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 (NIPS 2017)*, 5998–6008. <https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf>

Viola, P., & Jones, M. (2001). Rapid Object Detection Using a Boosted Cascade of Simple Features. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, 511–518.

<https://doi.org/10.1109/CVPR.2001.990517>

Wang, Z. J., Turko, R., Shaikh, O., Park, H., Das, N., Hohman, F., Kahng, M., & Polo Chau, D. H. (2021). CNN Explainer: Learning Convolutional Neural Networks with Interactive Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 27(2), 1396–1406. <https://doi.org/10.1109/TVCG.2020.3030418>

Wicaksana, I. P. R. E., Davinsi, G. R., Afriyanto, M. A., Wibowo, A., & Suri, P. A. (2024). *Systematic Literature Review: The Influence and Effectiveness of Deep Learning in Image Processing for Emotion Recognition*. <https://assets-eu.researchsquare.com/files/rs-3856084/v1/fe3e2728-6ba7-492d-9b3b-0ee0bfc0fa00.pdf?c=1705613865>

Williams, R. J., & Zipser, D. (1989). A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Computation*, 1(2), 270–280.

<https://doi.org/10.1162/neco.1989.1.2.270>

- Xu, T., White, J., Kalkan, S., & Gunes, H. (2020). Investigating Bias and Fairness in Facial Expression Recognition. *ArXiv*. <https://arxiv.org/abs/2007.10075>
- Yang, H., Zhang, Z., & Yin, L. (2018). Identity-Adaptive Facial Expression Recognition Through Expression Regeneration Using Conditional Generative Adversarial Networks. *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, 294–301. <https://doi.org/10.1109/FG.2018.00050>
- Yao, L., & Guan, Y. (2018). An Improved LSTM Structure for Natural Language Processing. *2018 IEEE International Conference of Safety Produce Informatization (IICSPI)*, 565–569. <https://doi.org/10.1109/IICSPI.2018.8690387>
- Ye, S. (2024). Applying Ensemble Learning to Multiple Stock Price Predictions: A Comparative Study. *Applied and Computational Engineering*, 50(1), 189–198. <https://doi.org/10.54254/2755-2721/50/20241501>
- Yu, Z., Liu, G., Liu, Q., & Deng, J. (2018). Spatio-Temporal Convolutional Features with Nested LSTM for Facial Expression Recognition. *Neurocomputing*, 317, 50–57. <https://doi.org/10.1016/j.neucom.2018.07.028>
- Yu, Z., & Zhang, C. (2015). Image Based Static Facial Expression Recognition with Multiple Deep Network Learning. *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*, 435–442. <https://doi.org/10.1145/2818346.2830595>
- Zeng, D., Veldhuis, R., & Spreeuwiers, L. (2021). A Survey of Face Recognition Techniques Under Occlusion. *IET Biometrics*, 10(6), 581–606. <https://doi.org/10.1049/bme2.12029>

Zhang, L., Verma, B., Tjondronegoro, D., & Chandran, V. (2019). Facial Expression Analysis under Partial Occlusion. *ACM Computing Surveys*, 51(2), 1–49.

<https://doi.org/10.1145/3158369>

Zhang, P., Dai, X., Yang, J., Xiao, B., Yuan, L., Zhang, L., & Gao, J. (2021). Multi-Scale Vision Longformer: A New Vision Transformer for High-Resolution Image Encoding.

ArXiv. <https://arxiv.org/abs/2103.15358>

Zhang, Z., Luo, P., Loy, C. C., & Tang, X. (2016). From Facial Expression Recognition to Interpersonal Relation Prediction. *ArXiv*. <https://arxiv.org/abs/1609.06426>

Zhao, L., & Zhang, Z. (2024). An Improved Pooling Method for Convolutional Neural Networks. *Scientific Reports*, 14(1), 1589. <https://doi.org/10.1038/s41598-024-51258-6>

Zheng, W., Zheng, K., Gao, L., Zhangzhong, L., Lan, R., Xu, L., & Yu, J. (2024). GRU–Transformer: A Novel Hybrid Model for Predicting Soil Moisture Content in Root Zones. *Agronomy*, 14(3), 432. <https://doi.org/10.3390/agronomy14030432>

Appendices

Appendix A: Multi-Path CNN Model Parameters.

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 64, 64, 1)	0	[]
conv2d (Conv2D)	(None, 64, 64, 32)	320	['input_1[0][0]']
batch_normalization (BatchNormalization)	(None, 64, 64, 32)	128	['conv2d[0][0]']
re_lu (ReLU)	(None, 64, 64, 32)	0	['batch_normalization[0][0]']
separable_conv2d (SepConv2D)	(None, 64, 64, 32)	1344	['re_lu[0][0]']
batch_normalization_1 (BatchNormalization)	(None, 64, 64, 32)	128	['separable_conv2d[0][0]']
re_lu_1 (ReLU)	(None, 64, 64, 32)	0	['batch_normalization_1[0][0]']
conv2d_1 (Conv2D)	(None, 64, 64, 32)	9248	['re_lu_1[0][0]']
batch_normalization_2 (BatchNormalization)	(None, 64, 64, 32)	128	['conv2d_1[0][0]']
max_pooling2d (MaxPooling2D)	(None, 32, 32, 32)	0	['batch_normalization_2[0][0]']
re_lu_2 (ReLU)	(None, 32, 32, 32)	0	['max_pooling2d[0][0]']
dropout (Dropout)	(None, 32, 32, 32)	0	['re_lu_2[0][0]']
tf.compat.v1.gather (TFOpLambda)	(None, 32, 32, 32)	0	['dropout[0][0]']
tf.split (TFOpLambda)	[(None, 32, 32, 16), (None, 32, 32, 16)]	0	['tf.compat.v1.gather[0][0]']
conv2d_2 (Conv2D)	(None, 32, 32, 18)	2610	['tf.split[0][0]']
conv2d_3 (Conv2D)	(None, 32, 32, 18)	2610	['tf.split[0][1]']
batch_normalization_3 (BatchNormalization)	(None, 32, 32, 18)	72	['conv2d_2[0][0]']
batch_normalization_4 (BatchNormalization)	(None, 32, 32, 18)	72	['conv2d_3[0][0]']
re_lu_3 (ReLU)	(None, 32, 32, 18)	0	['batch_normalization_3[0][0]']
re_lu_4 (ReLU)	(None, 32, 32, 18)	0	['batch_normalization_4[0][0]']
tf.concat (TFOpLambda)	(None, 32, 32, 36)	0	['re_lu_3[0][0]', 're_lu_4[0][0]']
separable_conv2d_1 (SepConv2D)	(None, 32, 32, 18)	990	['tf.concat[0][0]']
separable_conv2d_2 (SepConv2D)	(None, 32, 32, 18)	990	['tf.concat[0][0]']
batch_normalization_5 (BatchNormalization)	(None, 32, 32, 18)	72	['separable_conv2d_1[0][0]']
batch_normalization_6 (BatchNormalization)	(None, 32, 32, 18)	72	['separable_conv2d_2[0][0]']
re_lu_5 (ReLU)	(None, 32, 32, 18)	0	['batch_normalization_5[0][0]']
re_lu_6 (ReLU)	(None, 32, 32, 18)	0	['batch_normalization_6[0][0]']
tf.concat_1 (TFOpLambda)	(None, 32, 32, 36)	0	['re_lu_5[0][0]', 're_lu_6[0][0]']
conv2d_4 (Conv2D)	(None, 32, 32, 18)	5850	['tf.concat_1[0][0]']
conv2d_5 (Conv2D)	(None, 32, 32, 18)	5850	['tf.concat_1[0][0]']
batch_normalization_7 (BatchNormalization)	(None, 32, 32, 18)	72	['conv2d_4[0][0]']
batch_normalization_8 (BatchNormalization)	(None, 32, 32, 18)	72	['conv2d_5[0][0]']
re_lu_7 (ReLU)	(None, 32, 32, 18)	0	['batch_normalization_7[0][0]']
re_lu_8 (ReLU)	(None, 32, 32, 18)	0	['batch_normalization_8[0][0]']
tf.concat_2 (TFOpLambda)	(None, 32, 32, 36)	0	['re_lu_7[0][0]', 're_lu_8[0][0]']
lambda (Lambda)	(None, 32, 32, 1)	0	['tf.concat_2[0][0]']
conv2d_6 (Conv2D)	(None, 16, 16, 2)	20	['lambda[0][0]']
tf.math.reduce_mean (TFOpLambda)	(None, 1, 1, 36)	0	['tf.concat_2[0][0]']
conv2d_7 (Conv2D)	(None, 8, 8, 4)	76	['conv2d_6[0][0]']
dense (Dense)	(None, 1, 1, 9)	333	['tf.math.reduce_mean[0][0]']
conv2d_transpose (Conv2DTrans)	(None, 16, 16, 4)	148	['conv2d_7[0][0]']

dense_1 (Dense)	(None, 1, 1, 36)	360	['dense_1[0][0]']
conv2d_transpose_1 (Conv2DTrans)	(None, 32, 32, 1)	37	['conv2d_transpose_1[0][0]']
tf.math.multiply (TFOpLambda)	(None, 32, 32, 36)	0	['dense_1[0][0]', 'conv2d_transpose_1[0][0]']
tf.math.multiply_1 (TFOpLambda)	(None, 32, 32, 36)	0	['tf.math.multiply_1[0][0]', 'tf.concat_2[0][0]']
conv2d_8 (Conv2D)	(None, 32, 32, 36)	1332	['tf.math.multiply_1[0][0]']
batch_normalization_9 (BatchNormalization)	(None, 32, 32, 36)	144	['conv2d_8[0][0]']
max_pooling2d_1 (MaxPooling2D)	(None, 16, 16, 36)	0	['batch_normalization_9[0][0]']
re_lu_9 (ReLU)	(None, 16, 16, 36)	0	['max_pooling2d_1[0][0]']
dropout_1 (Dropout)	(None, 16, 16, 36)	0	['re_lu_9[0][0]']
conv2d_9 (Conv2D)	(None, 16, 16, 40)	13000	['dropout_1[0][0]']
batch_normalization_10 (BatchNormalization)	(None, 16, 16, 40)	160	['conv2d_9[0][0]']
re_lu_10 (ReLU)	(None, 16, 16, 40)	0	['batch_normalization_10[0][0]']
separable_conv2d_3 (SepConv2D)	(None, 16, 16, 40)	2000	['re_lu_10[0][0]']
batch_normalization_11 (BatchNormalization)	(None, 16, 16, 40)	160	['separable_conv2d_3[0][0]']
re_lu_11 (ReLU)	(None, 16, 16, 40)	0	['batch_normalization_11[0][0]']
conv2d_10 (Conv2D)	(None, 16, 16, 40)	14440	['re_lu_11[0][0]']
batch_normalization_12 (BatchNormalization)	(None, 16, 16, 40)	160	['conv2d_10[0][0]']
max_pooling2d_2 (MaxPooling2D)	(None, 8, 8, 40)	0	['batch_normalization_12[0][0]']
re_lu_12 (ReLU)	(None, 8, 8, 40)	0	['max_pooling2d_2[0][0]']
dropout_2 (Dropout)	(None, 8, 8, 40)	0	['re_lu_12[0][0]']
tf.compat.v1.gather_1 (TFOpLambda)	(None, 8, 8, 40)	0	['dropout_2[0][0]']
tf.split_1 (TFOpLambda)	[(None, 8, 8, 20), (None, 8, 8, 20)]	0	['tf.compat.v1.gather_1[0][0]']
conv2d_11 (Conv2D)	(None, 8, 8, 21)	3801	['tf.split_1[0][0]']
conv2d_12 (Conv2D)	(None, 8, 8, 21)	3801	['tf.split_1[0][1]']
batch_normalization_13 (BatchNormalization)	(None, 8, 8, 21)	84	['conv2d_11[0][0]']
batch_normalization_14 (BatchNormalization)	(None, 8, 8, 21)	84	['conv2d_12[0][0]']
re_lu_13 (ReLU)	(None, 8, 8, 21)	0	['batch_normalization_13[0][0]']
re_lu_14 (ReLU)	(None, 8, 8, 21)	0	['batch_normalization_14[0][0]']
tf.concat_3 (TFOpLambda)	(None, 8, 8, 42)	0	['re_lu_13[0][0]', 're_lu_14[0][0]']
separable_conv2d_4 (SepConv2D)	(None, 8, 8, 21)	1281	['tf.concat_3[0][0]']
separable_conv2d_5 (SepConv2D)	(None, 8, 8, 21)	1281	['tf.concat_3[0][0]']
batch_normalization_15 (BatchNormalization)	(None, 8, 8, 21)	84	['separable_conv2d_4[0][0]']
batch_normalization_16 (BatchNormalization)	(None, 8, 8, 21)	84	['separable_conv2d_5[0][0]']
re_lu_15 (ReLU)	(None, 8, 8, 21)	0	['batch_normalization_15[0][0]']
re_lu_16 (ReLU)	(None, 8, 8, 21)	0	['batch_normalization_16[0][0]']
tf.concat_4 (TFOpLambda)	(None, 8, 8, 42)	0	['re_lu_15[0][0]', 're_lu_16[0][0]']
conv2d_13 (Conv2D)	(None, 8, 8, 21)	7959	['tf.concat_4[0][0]']
conv2d_14 (Conv2D)	(None, 8, 8, 21)	7959	['tf.concat_4[0][0]']
batch_normalization_17 (BatchNormalization)	(None, 8, 8, 21)	84	['conv2d_13[0][0]']
batch_normalization_18 (BatchNormalization)	(None, 8, 8, 21)	84	['conv2d_14[0][0]']
re_lu_17 (ReLU)	(None, 8, 8, 21)	0	['batch_normalization_17[0][0]']
re_lu_18 (ReLU)	(None, 8, 8, 21)	0	['batch_normalization_18[0][0]']
tf.concat_5 (TFOpLambda)	(None, 8, 8, 42)	0	['re_lu_17[0][0]', 're_lu_18[0][0]']
lambda_1 (Lambda)	(None, 8, 8, 1)	0	['tf.concat_5[0][0]']
conv2d_15 (Conv2D)	(None, 4, 4, 2)	20	['lambda_1[0][0]']
tf.math.reduce_mean_1	(None, 1, 1, 42)	0	['tf.concat_5[0][0]']

(TFOpLambda)			
conv2d_16 (Conv2D)	(None, 2, 2, 4)	76	['conv2d_15[0][0]']
dense_2 (Dense)	(None, 1, 1, 10)	430	['tf.math.reduce_mean_1[0][0]']
conv2d_transpose_2 (Conv2DTrans)	(None, 4, 4, 4)	148	['conv2d_16[0][0]']
dense_3 (Dense)	(None, 1, 1, 42)	462	['dense_2[0][0]']
conv2d_transpose_3 (Conv2DTrans)	(None, 8, 8, 1)	37	['conv2d_transpose_2[0][0]']
tf.math.multiply_2 (TFOpLambda)	(None, 8, 8, 42)	0	['dense_3[0][0]', 'conv2d_transpose_3[0][0]']
tf.math.multiply_3 (TFOpLambda)	(None, 8, 8, 42)	0	['tf.math.multiply_2[0][0]', 'tf.concat_5[0][0]']
conv2d_17 (Conv2D)	(None, 8, 8, 42)	1806	['tf.math.multiply_3[0][0]']
batch_normalization_19 (BatchNormal- ization)	(None, 8, 8, 42)	168	['conv2d_17[0][0]']
max_pooling2d_3 (MaxPooling2D)	(None, 4, 4, 42)	0	['batch_normalization_19[0][0]']
re_lu_19 (ReLU)	(None, 4, 4, 42)	0	['max_pooling2d_3[0][0]']
pwfs (PWFS)	(None, 32, 32, 16)	0	['dropout[0][0]']
pwfs_1 (PWFS)	(None, 16, 16, 18)	0	['dropout_1[0][0]']
pwfs_2 (PWFS)	(None, 8, 8, 20)	0	['dropout_2[0][0]']
dropout_3 (Dropout)	(None, 4, 4, 42)	0	['re_lu_19[0][0]']
global_average_pooling2d (GlobalAveragePooling2D)	(None, 16)	0	['pwfs[0][0]']
global_average_pooling2d_1 (GlobalAveragePooling2D)	(None, 18)	0	['pwfs_1[0][0]']
global_average_pooling2d_2 (GlobalAveragePooling2D)	(None, 20)	0	['pwfs_2[0][0]']
global_average_pooling2d_3 (GlobalAveragePooling2D)	(None, 42)	0	['dropout_3[0][0]']
tf.concat_6 (TFOpLambda)	(None, 96)	0	['global_average_pooling2d[0][0]', 'global_average_pooling2d_1[0][0]', 'global_average_pooling2d_2[0][0]', 'global_average_pooling2d_3[0][0]']
dense_4 (Dense)	(None, 7)	679	['tf.concat_6[0][0]']

Total params: 93410 (364.88 KB)

Trainable params: 92354 (360.76 KB)

Non-trainable params: 1056 (4.12 KB)

Appendix B: Transfer Learning VGG16 Parameters.

Layer (type)	Output Shape	Param#
input_1 (InputLayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1,792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36,928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73,856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147,584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295,168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590,080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590,080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1,180,160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2,359,808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2,359,808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2,359,808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2,359,808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2,359,808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0

Total params: 14,714,688 (56.13 MB)

Trainable params: 14,714,688 (56.13 MB)

Non-trainable params: 0 (0.00 Byte)

Fune-Tuned Output Layers

Layer (type)	Output Shape	Param#
vgg16 (Functional)	(None, 7, 7, 512)	14,714,688
flatten (Flatten)	(None, 25,088)	0
dense (Dense)	(None, 512)	12,845,568
dense_1 (Dense)	(None, 7)	3,591

Total params: 27,563,847 (105.15 MB)

Trainable params: 27,563,847 (105.15 MB)

Non-trainable params: 0 (0.00 Byte)