Arabic Sign Language Recognition:
A Deep Learning Approach

تمييز لغة الإشارة العربية:
أسلوب التعلّم العميق

by

HAMDA GHALIB AWADH ALI ALMAHRI

Dissertation submitted in fulfilment
of the requirements for the degree of
MSc INFORMATICS
at
The British University in Dubai

May 2022

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that the content of this research is the direct result of my own work and that any use made in it of published or unpublished copyright material falls within the limits permitted by international copyright conventions.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

ABSTRACT

With more than 300 sign languages across the world, sign interprets are not always available to translate spoken words into sign language and vice versa. As people with hearing and speech impairments rely on Sign Language for communication, this would limit their communication with others. A solution for this would be utilizing Sign Language Recognition systems, which allow for communication between users of the sign language and those who do not without the need for interpreters.

As we consider the success of Deep Learning for Computer Vision tasks, we observe the advantage it can provide for Arabic Sign Language Recognition. For this research, we have two aims. First, we would like to review the current status of research in Arabic Sign Language Recognition using Deep Learning and find research gaps. Second, we aim to build a Sign Language Recognition system that bridges the gap.

We achieve this through a systematic review that identifies primary studies using deep learning models for Arabic Sign Language Recognition. Out of 414 identified studies, 67 were deemed of relevance to our topic. Out of those, 32 studies passed our full selection procedure. We were able to discover patterns in research and find that the biggest issue is data collection as current datasets don't offer enough variety and are not representative of real-life scenarios. Current methods are either too expensive, or easily affected by the surrounding environment.

Thus, for the second part, we offer a solution for data collection using MediaPipe, which allow us to collect data directly through the webcam. We are able to leverage this framework to build a recognition system for Emirati Sign Language that recognizes the signs for the seven Emirates. We used an LSTM model and achieve an accuracy of 100% in the testing dataset.

الملخص

مع وجود أكثر من 300 لغة إشارة حول العالم ، قد لا يتوفر دائمًا متخصصو ترجمة لغة الإشارة. نظرًا إلى أن الأشخاص الذين يعانون من الإعاقات السمعية والكلامية يعتمدون على لغة الإشارة للتواصل، فإن هذا من شأنه أن يحد من تواصلهم مع الآخرين. احدى الحلول لذلك هو استخدام أنظمة التعرف على لغة الإشارة، والتي تسمح بالتواصل بين مستخدمي لغة الإشارة وغير مستخدميها دون الاستعانة إلى مترجمين.

نظرًا إلى الاهتمام المتزايد بالتعلم العميق لمهام رؤية الكمبيوتر، فإننا نلاحظ الميزة التي يمكن أن توفرها هذه النظم لتمييز لغة الإشارة العربية. في هذا البحث، لدينا هدفان: أولاً، نود النظر في الوضع الحالي للأبحاث التي تغطي موضوع تمييز لغة الإشارة العربية باستخدام التعلم العميق والعثور على فجوات البحث. ثانيًا ، نهدف إلى بناء نظام تمييز لغة الإشارة العربية بما يسد الفجوة الموجودة.

نحقق ذلك من خلال مراجعة منهجية تحدد الدراسات لموضوعنا. من بين 414 دراسة تم تحديدها ، اعتبرت 67 دراسة ذات صلة بموضوعنا. من بين هؤلاء ، حددنا 32 دراسة لإجراء المراجعة الكاملة. تمكنا من اكتشاف الأنماط في البحث ووجدنا أن المشكلة الأكبر هي عملية جمع البيانات لأن مجموعة البيانات الحالية لا تمثل سيناريوهات الحياة الواقعية. الأساليب الحالية إما باهظة الثمن أو تتأثر بسهولة بالبيئة المحيطة.

بالتالي ، بالنسبة للجزء الثاني ، نقدم حلاً لجمع البيانات باستخدام MediaPipe ، والذي يسمح لنا بجمع البيانات مباشرةً من خلال كاميرا الويب. نستفيد من هذه التقنية في بناء نظام تمييز لغة الإشارة الإماراتية الذي يتعرف على إشارات الإمارات السبع. استخدمنا نموذج LSTM وحققنا دقة بنسبة 100٪ في مجموعة بيانات الاختبار.

DEDICATION

I would like to dedicate this dissertation to my parents and siblings, who have been a constant source of support and encouragement throughout my study. Thank you for inspiring me and helping me achieve my goals.

I hope that this work may contribute to research and benefit our community.

Contents

List of Illustrations

List of Tables

1 Introduction

Deep Learning has been applied with huge success to a variety of tasks including natural language processing, pattern recognition and computer vision. In this dissertation, we aim to research applying deep learning to the task of Arabic Sign Language Recognition. To provide a clear aim for this research, we set two goals.

1. Review the current research on Arabic Sign Language Recognition using deep learning. This will orient our research and allow us to focus on a real problem that we can solve.
2. Build a Deep Learning system that contributes to current research based on research gaps identified in the first question.

To achieve our first goal, we will be doing a systematic review that will provide a clearer perspective on the current position of research in this area and find current trends and/or research gaps. Based on our findings, we will then create an Arabic Sign Language Recognition System built to bridge gaps in research and solve any issues that may present themselves.

This dissertation is divided as following: Section 1 introduces the topic, Section 2 covers the systematic review and background, Section 3 will cover our methodology in bridging research gaps and how we will build the system, Section 4 relays the results, Section 5 discusses our findings along with prospects for future work, Section 6 will conclude the dissertation.

1.1 Arabic Sign Language (ArSL)

Sign Language utilizes a combination of hand gestures, facial expressions and body language as a method of communication. People with hearing and speech impairments rely on sign language to communicate with others. As such, sign interpreters would translate spoken words into sign language and vice versa. However, there remains challenges in employing their

services due to the lack of experts in this field and the large number of sign languages, as there are more than 300 sign languages across the world (Adeyanju, Bello, and Adegboye, 2021).

Sign language is structured differently from spoken and written language, with its own grammar, phonology, syntax, etc. The signs are made using specific hand configurations with the following parameters affecting the gesture meaning: hand shape, location, orientation, movement, and facial expressions. There is no universal sign language, and it will differ from one geographical location to the next. In fact, countries that share the same language may use different sign languages if the deaf community in these countries were not in contact when developing the sign language. This is actually the case for Arabic Sign Language (Adeyanju, Bello, and Adegboye, 2021).

In 2001, the Arabic Federation of the Deaf formally introduced the Arabic Sign Language as the official sign language of the speech and hearing impaired in Arab nations. However, while the Arabic Language is the fourth largest spoken language in the world, the Arabic Sign Language (ArSL) in considered to still be in its developmental stages. To understand the issues pertaining to the sign language, we must first identify the role of the Arabic Language as a spoken and written language and how it affects the Arabic Sign Language (Mustafa, 2021).

Arabic is the official language of nearly 380 million people in the Middle East, covering regions from North Africa to the Arabian Gulf. However, there are existing varieties in Arabic script between the Classic Arabic, standardized Modern Standard Arabic (MSA) and non-standardized regional dialects. The Classic Arabic is the language of the Holy Quran, it is used in poems, literary writing, and was spoken for more than 14 centuries by Arabs. The Modern Standard Arabic is directly descended from the Classic Arabic and is used today in formal context, both spoken and written, such as in radio broadcasts, news, and non-entertainment

content. As both Classic Arabic and MSA use the same syntax and morphology, little distinction is made between the two.

On the other hand, regional dialects vary greatly from one region to another and are normally used in daily speech. For this reason, different dialects resulted in different Arabic Sign Languages. What this means is that while there is a “diglossia” case in which people of the Arab world can understand both MSA/Classic Arabic in addition to their own dialect and communicate effectively using both, the same cannot be said for sign language as each region sign language is created from the local dialect. While some interpreters advocate teaching and using the Unified Arabic Sign Language, there is pushback from local communities, citing the difficulties faced by the deaf in learning two separate sign languages. That being said, similar signs are still used across the different dialects for the Arabic Alphabet and as a foundation for signs (Al-Shamayleh et al., 2020).

1.2 ArSL Recognition

Sign Language Recognition is the task of recognizing actions from sign language. It involves multiple disciplines including Computer Vision and Natural Language Processing. The task involves the understanding of hand motions, positions, orientations and gets further complicated once we consider arm movement, facial expressions, and body language. Automatic Sign Language Recognition would allow communication between users of the sign language and those who do not without the need for interpreters, which gives more independence for the hearing and speech impaired community to socialize with their families, friends, peers, and integrate within society. This becomes even more important when considering the fact that there is no universal sign language across the world.

Generally speaking, the task of Sign Language Recognition would involve: (1) Data Acquisition, (2) Pre-processing, (3) Feature Extraction, (4) Classification (Al-Shamayleh et al., 2020).

1. **Data Acquisition**

The importance of this step lies in the quality of the data. The better the data, the better the system performs. There are multiple approaches to acquiring the data including vision-based methods, sensor-based methods, or using an existing dataset. Based on the input data, the researcher may opt for any of these approaches and use an appropriate collection device. For example, a researcher may use a camera to collect images/frames in a vision-based approach, while another may collect electrical signals through sensors in a sensor-based approach. We will be going into more details in section 2.2.

2. **Pre-processing**

In this phase, the raw data collected will need to be prepared in a way that makes it suited for use and training the model. This step might vary based on whether the data is an image, video, signals, skeletal data, etc. It might include filtering, noise reduction, resizing, image enhancement, and segmentation. In the case of images and videos, the segmentation step is important as it allows the system to identify the Region Of Interest (ROI).

3. **Feature Extraction**

In this step, the most relevant features are derived from the data to be fed into the classification model. The aim is to reduce the amount of data needed to train the model, which in turn reduces the computational cost and training time. It enhances the learning accuracy as it allows for better classification by identifying the features that can distinguish the different classes. Different features can be extracted such as geometric and statistical features.

4. Classification

In this stage, the model will classify our input data into different classes. In the case of sign language recognition, this means that the model will be trained on different signs and gestures and learn from the extracted features. The learned model will then classify test data, different from the training data, and output the class of the sign. The model performance can be measure through the recognition rate. There are multiple approaches to the classification stage: Rule-based, and Machine Learning-based.

The rule-based approach uses manually established rules set between the features to match the input against a sign or gesture while the machine learning-based approach will use models that creates the mappings on its own. While rule-based approaches rely on explicitly stated and static rules which lower the recognition rate, it requires less data than machine learning-based approach. Meanwhile, machine learning-based approach can learn its own rules from the data output and improves the more data is fed into training the model.

The challenges in developing a sign language recognition system include signing speed, which varies per signer. Segmentation of region of interest from entire images are also problematic due to the different environments in which the images are taken, illuminations, computational time, gesture tracking, nature of parameters used for communication such as hand shape, facial expression, orientation and so on (Adeyanju, Bello and Adegboye, 2021).

1.3 Deep Learning

Currently, there is a rising interest in applying deep learning in the task of Arabic Sign Language Recognition in order to manage the data. Just as humans learn through repeating tasks, machine learning algorithms are trained to learn and improve, and where learning can be supervised or

unsupervised. In supervised machine learning, the model takes in both labelled input and output training data to infer a function. While in unsupervised machine learning, only the input data is provided, and the model draws inferences (Adeyanju, Bello and Adegboye, 2021).

Deep learning is a group of machine learning methods that are based on neural networks; an approach that is inspired by biological neural networks in a brain and aims to mimic human reasoning (Mustafa, 2021). Simply put, the network is made of multiple layers including an input layer, several hidden layers, and an output layer. The more the layers, the deeper the network. The layers are connected through nodes, and each layer's output will be used as the input for the next layer. Lower layers will typically extract and learn simple features from the input data, and the next layers derive more complex features, with the complexity increasing as we move to higher layers. The "learning" happens through adjusting its node connections, which are called weights, and change the model interior parameters. This allows deep learning to find unpredictable structure in large datasets.

In the past, neural networks were harder to apply due to the large computation power required. However, recent advances led to a leap forward in deep learning research, including advances in hardware and computational power, availability of large labelled datasets, development of pre-trained models by specialists in the field. These systems have achieved huge success in the domains of Computer Vision and Natural Language Processing. Researchers were able to build state-of-the-art architectures based on deep learning and apply them to facial detection, speech recognition, machine translation, text classification, etc.

2 Systematic Review

Due to the lack of systematic reviews covering studies applying a deep learning approach to Arabic Sign Language Recognition, this systematic review aims to provide a clearer perspective on the current state of research in this area and will serve as the basis for the next part of this dissertation, in which we apply our findings to the problem of Arabic Sign Language. This section is divided as following: Subsection 1 explains the methodology for the systematic review, Subsection 2 relays the results, Subsection 3 discusses our findings along with prospects for future work that will serve as basis for us to bridge possible gaps.

2.1 Methodology

2.1.1 Search Strategy

We use a systematic review approach (Kitchenham, 2004) for our research. In this method, we first identify our research questions, keywords, and data sources. Then, we choose an include and exclude criteria for our study selection process. In the next step, a quality assessment is performed on the final papers to confirm their relevance to our research questions. Lastly, we extract information from the papers and discuss the results.

2.1.2 Identification of Research Questions

We first identify our research questions as follows:

RQ1: What are the existing studies on Arabic Sign Language Recognition using a deep learning approach?

RQ2: How was the task of Arabic Sign Language Recognition approached?

RQ3: What data was used?

RQ4: Which deep learning methods were used?

RQ5: How did the models perform?

We will use the PICO model (Schardt et al. 2007) as a framework for our research questions in the context of academic research.

Population: Arabic Sign Language

Intervention: Recognition using deep learning models

Comparison: Other models

Outcome: Arabic Sign Language Recognition (Accuracy)

2.1.3 Identification of Keywords and Data Sources

We collect our articles from reputable electronic bases as follows: ACM Digital Library, IEEE Xplore, ScienceDirect and Google Scholar. Table 1 shows the number of articles by source.

We search for the keywords ("Arabic Sign Language") AND ("Recognition") using the available function in each database to find relevant search results.

Data Source	Number of articles
ACM Library	27
IEEE Xplore	72
Science Direct	63
Google Scholar	252
Total	414

Table 1 Articles by source

2.1.4 Study Selection Criteria

In this phase, we apply an inclusion and exclusion criteria to ensure the information is relevant to our research questions.

Inclusion Criteria:

- Must be a primary study focused on ArSL Recognition
- Must include Arabic Sign Language
- Must use Deep Learning methods in the classification stage
- Must be published within the last 8 years.

Exclusion Criteria:

- Not in the domain of ArSL
- Focused on tasks other than ArSL recognition
- Does not include Deep learning methods

2.1.5 Study Selection Procedure

Using Rayyan (Ouzzani et al. 2016), a free web application for systematic reviews, we start screening our papers through multiple stages. Figure 1 follows the PRISMA flowchart (Moher et al. 2009), which demonstrates our selection procedure and how the articles were selected or excluded.

Phase 1: Identification: Keyword based filtering – We use the keywords we chose to search in databases. This search yields 414 articles and will be the base for our selection process.

Phase 2: Remove duplicates – We remove any duplicate articles. 84 articles were excluded and 330 included for the next phase.

Phase 3: Screening: Title and Abstract based filtering – We examine each article's title and abstract for the set criteria, removing any article deemed unrelated to our research. Articles were excluded if they were outside the domain of Sign Language Recognition, focused on other tasks, or were not conducted on Arabic Sign Language. Results that were surveys, book

chapters, secondary studies, or published before 2014 were removed as well. Articles deemed of any relevance were included. In total, 263 articles were excluded and 67 included.

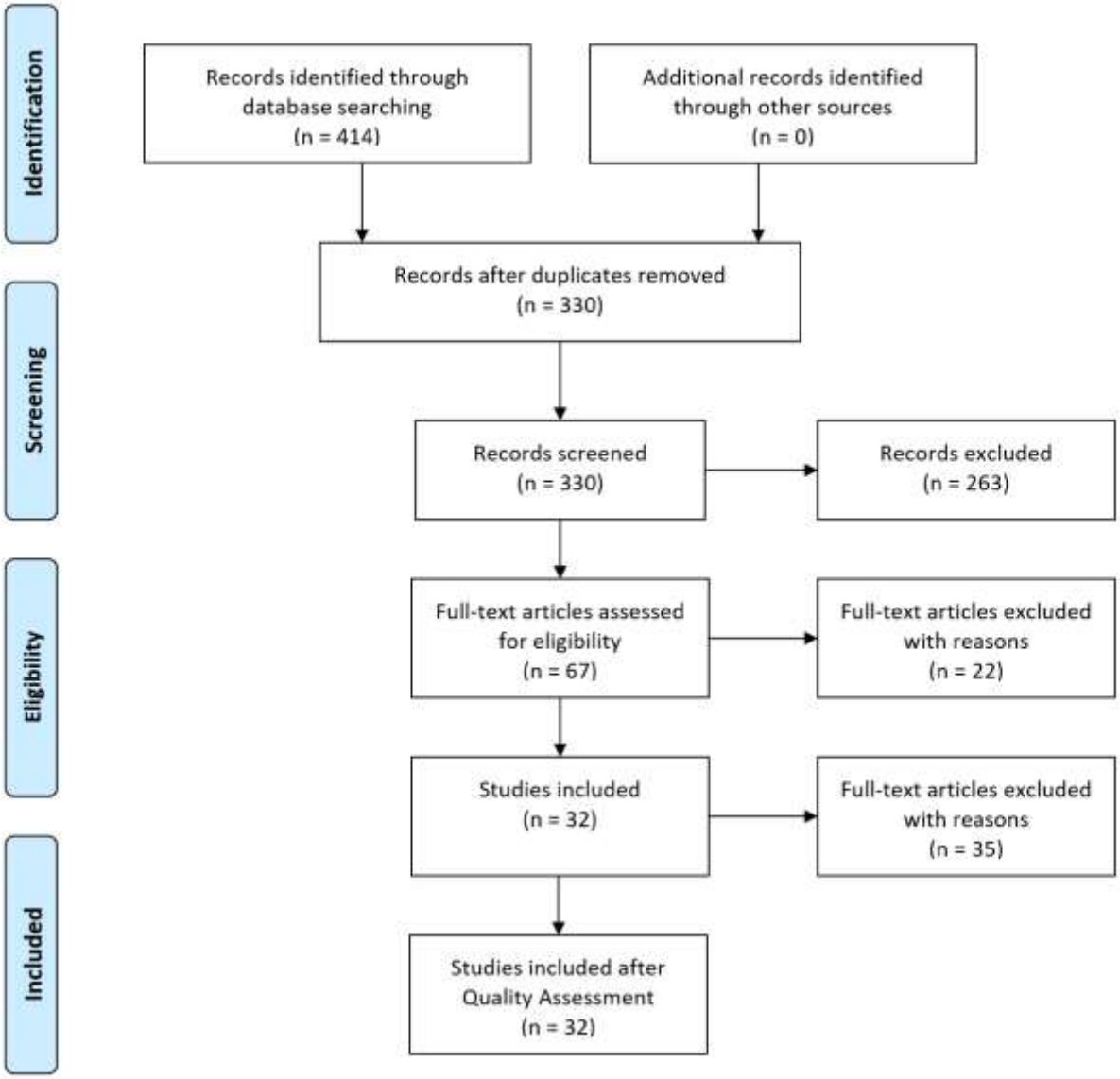


Figure 1 PRISMA flowchart

Phase 4: Eligibility: Full-text screening – In this final phase, we examine the articles closely. We check which data was used and if deep learning models were used or not. In the end, 35 articles were excluded, and 32 final articles fulfilled our criteria.

2.1.6 Quality Assessment (Quality Criteria)

We next define a quality criterion to ensure that the papers follow the aim of this systematic review and answer our research questions. The final articles included must answer the following questions:

1. Arabic Sign Language
    a. Acquiring method of the data is clearly defined?
    b. ArSL dataset is clearly described, including if the signs are alphabets, isolated, continuous and static/dynamic?
2. Recognition System
    a. Does the paper clearly identify goals/results relating to ArSL recognition and how their paper contributes to current research?
    b. Is the recognition system clearly described, including the different steps?
3. Deep Learning
    a. Is a deep learning model used at the classification level and described clearly?
    b. Evaluation methods are used, and recognition rate is stated.

We assign 1 point for each of these six questions as they hold the same weight in our review. We assign 1 point if the question is covered and 0 if not. If the question is answered partially, we assign 0.5 points. From there we sum the score of the six questions for a total of six and calculate the percentage.

Table 2 shows the result based on this assessment method. Articles that scored below 75% were discarded, as they do not fulfill the aim of the study. We can see that all 32 studies fulfill the criteria.

Study	Q1		Q2		Q3		Total	Percentage
	a	b	a	b	a	b		
(Elons, 2014)	0	0.5	1	1	1	1	4.5	75%
(Elons et al., 2014)	1	0.5	1	1	1	1	5.5	91.6%
(Mohandes, Aliyu and Deriche, 2014)	1	1	1	1	1	1	6	100%
(ALI, et al., 2014)	0.5	0.5	0.5	1	1	1	4.5	75%
(Samir, Aboulela and Tolba, 2014)	0.5	1	1	1	1	1	5.5	91.6%
(Al Mashagba, Al Mashagba and Nassar, 2014)	1	1	1	1	1	1	6	100%
(ElBadawy et al., 2015)	1	1	1	1	1	1	6	100%
(Sadeddine, Chelali and Djeradi, 2015)	1	1	0.5	1	1	1	5.5	91.6%
(ElBadawy et al., 2017)	1	0.5	0.5	1	1	1	5	83.33%
(Hisham and Hamouda, 2017)	1	1	1	1	1	1	6	100%
(Sadeddine et al., 2018)	1	1	1	1	1	1	6	100%
(Ahmed et al., 2018)	1	1	0.5	1	1	0.5	5	83.33%
(Hayani et al., 2019)	1	1	0.5	1	1	1	5.5	91.6%
(Kamruzzaman, 2020)	1	1	1	1	1	1	6	100%
(Latif et al., 2020)	1	1	1	1	1	1	6	100%
(Aly and Aly, 2020)	1	1	1	1	1	1	6	100%
(Alnahhas et al., 2020)	1	1	0.5	1	1	1	5.5	91.6%
(Suliman et al., 2021)	1	1	0.5	1	1	1	5.5	91.6%
(Bencherif et al., 2021)	1	1	1	1	1	1	6	100%
(Hamou and Chelali, 2021)	1	1	0.5	1	1	1	5.5	91.6%
(Tharwat, Ahmed and Bouallegue, 2021)	1	1	1	1	0.5	1	5.5	91.6%
(Boukdir et al., 2021)	0	1	1	1	1	1	5	83.33%
(Alani and Cosma, 2021)	1	1	1	1	1	1	6	100%
(Ismail, Dawwd and Ali, 2021)	0.5	1	1	1	1	1	5.5	91.6%
(Yousuf, Alwarfalli and Ighneiwa, 2021)	1	1	1	1	1	0.5	5.5	91.6%
(Ritonga et al., 2021)	1	1	0.5	1	1	1	5.5	91.6%
(Alawwad, Bchir and Maher, 2021)	1	1	1	1	1	1	6	100%

(Alshomrani, Aljoudi and Arif, 2021)	1	1	0.5	1	1	1	5.5	91.6%
(Luqman, El-Alfy and BinMakhashen, 2020)	1	0.5	1	1	1	1	5.5	91.6%
(Dabwan and Jadhav, 2021)	0.5	0.5	1	1	1	1	5	83.33%
(Abdul et al., 2021)	1	1	1	1	1	1	6	100%
(Sadeddine et al., 2021)	1	1	1	1	1	1	6	100%

Table 2 Quality Assessment

2.1.7 Data Extraction

We condense our articles in a table by extracting the following information: Author, Source, and Year, Classification Category, single/double handed, Static/Dynamic, Preprocessing, Feature Extraction, Classifier, Signer Dependent, Recognition Rate, Real time recognition. Results are summarized in table 3.

#	Author	Source	Year			
1	(Elons, 2014)	Conference Proceedings	2014	**Classification Category**	**single/double handed**	**Static/Dynamic**
				-	Both	Both
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)		200 gestures that are composed of 270 postures (189 postures containing two hands and 81 postures containing one hand). Each gesture is performed 10 times by 2 different people. 60 % for training purposes 40% testing
				Preprocessing	**Feature Extraction**	**Classifier**
					PCNN	Multi-level Multiplicative Neural Network (MNN)
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	83%	yes
2	(Elons et al., 2014)	Conference Proceedings	2014	**Classification Category**	**single/double handed**	**Static/Dynamic**
				Isolated words	Double handed	Dynamic
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	Leap motion sensor	50 different dynamic signs 4 different persons, two signs set are used as training set and the other two are used as test set
				Preprocessing	**Feature Extraction**	**Classifier**
					The fingers position and the distances between the fingers in each frame	Multi-layer perceptron Neural Network (MLP)
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	88%	no
3	(Mohandes, Aliyu and Deriche, 2014)	Conference Proceedings	2014	**Classification Category**	**single/double handed**	**Static/Dynamic**
				alphabet	single handed	Dynamic
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	Leap motion sensor	28 alphabets with total of 2800 frames for training testing (10 samples with each sample having 10 frames for each letter)
				Preprocessing	**Feature Extraction**	**Classifier**
					12 features including: finger length, finger width, average tip position with respect to x, y, and z-axis, hand sphere radius, palm position with respect to x, y, and z-axis, hand pitch, roll and yaw.	Multi-layer perceptron Neural Network (MLP) Nave Bayes Classifier (NBC)
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	99%	no
4	(ALI, et al., 2014)	Journal	2014	**Classification Category**	**single/double handed**	**Static/Dynamic**
				Isolated words	Double handed	static
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	Webcam	5 two-handed signs (200 samples each) were collected from one signer. 150 for training, 50 for testing
				Preprocessing	**Feature Extraction**	**Classifier**
				Binary skin classifiers: YCbCr and HSV segmentation: quantization technique	Hu moments	Multi-layer perceptron Neural Network (MLP)
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	87.60%	no
5	(Samir, Aboulel	Journal	2014	**Classification Category**	**single/double handed**	**Static/Dynamic**

	a and Tolba, 2014)			Isolated words	Both	static
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	Camera	100 Arabic signs
				Preprocessing	**Feature Extraction**	**Classifier**
				Segmentation: Clustering-based	PCNN	Multi-Stage Multiplicative Neural Network (MMNN) Multi-layer perceptron networks (MLP)
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	94%	yes
6	(Al Mashaqba, Al Mashaqba and Nassar, 2014)	Journal	2014	**Classification Category**	**single/double handed**	**Static/Dynamic**
				Isolated words	Double handed	Dynamic
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	Camera, colored gloved	40 signs (80-220 frames) by 10 signers half for training and half for testing
				Preprocessing	**Feature Extraction**	**Classifier**
				Hue Saturation Lightness Model (HIS) Segmentation: Gesture Mixture Model (GMM)	the centroid position for each hand horizontal and vertical velocity change of both hands across the frames the area change for each hand across the frames	Time Delay Neural Network (TDNN)
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	70%	no
7	(ElBadawy et al., 2015)	Conference Proceedings	2015	**Classification Category**	**single/double handed**	**Static/Dynamic**
				Isolated words	Double handed	Dynamic
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	1 Leap motion sensor (hands and fingers movements) 2 Cameras (facial expressions and body movement)	20 dynamic signs by two signers (right and left handed) including facial features and body movments half for testing half for training
				Preprocessing	**Feature Extraction**	**Classifier**
					PCNN	Multi-layer perceptron Neural Network (MLP)
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	95%	no
8	(Sadeddine, Chelali and Djeradi, 2015)	Conference Proceedings	2015	**Classification Category**	**single/double handed**	**Static/Dynamic**
				alphabet	single handed	static
				Data Acquisition Approach	**Collection Tools**	**Data**
				Public		Halawani Arabic Sign Language(ArSL)
				Preprocessing	**Feature Extraction**	**Classifier**
					Spatial reduction (DCT, DWT and PCA)	probabilistic neural network (PNN)
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	80.20%	no
9	(ElBadawy et al., 2017)	Conference Proceedings	2017	**Classification Category**	**single/double handed**	**Static/Dynamic**
				Isolated words	Both	Dynamic
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	Camera	200 samples from Unified Arab Sign Language dictionary (25 signs each done by two signers 4 times) 125 for training / 75 for testing
				Preprocessing	**Feature Extraction**	**Classifier**
				Segmentation: Canny Edge Detection	3D CNN	3D CNN (softmax layer)

				Signer Dependent	**Recognition Rate**	**Real time**
				yes	85%	no
10	(Hisham and Hamouda, 2017)	Journal	2017	**Classification Category**	**single/double handed**	**Static/Dynamic**
				multiple (alphabet/numbers/isolated)	Both	Both
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	Leap motion sensor	38 static gestures (28 letters, numbers (1:10) and 16 static words) 20 dynamic gestures
				Preprocessing	**Feature Extraction**	**Classifier**
				3D coordinates of hand motion and position of hands and fingers Segmentation: palm speed during the motion as transition during continous sentence (this is an additional proposal in the paper)	palm features set bone features	ANN with Multilayer Perceptron SVM KNN DTW
				Signer Dependent	**Recognition Rate**	**Real time**
				no (tested by frames from another user)	96.9 at highest and 81.80 at lowest (as complexity increases)	no
11	(Sadeddine et al., 2018)	Conference Proceedings	2018	**Classification Category**	**single/double handed**	**Static/Dynamic**
				alphabet	single handed	static
				Data Acquisition Approach	**Collection Tools**	**Data**
				Public		Halawani datase 30 signs with 65 images for each alphabet, total 1950 images
				Preprocessing	**Feature Extraction**	**Classifier**
				Segmentation: Otsu threshold	Hu Moments Local Binary Pattern Descriptor (LBPD) Zernike moments Generic Fourier	Multi-Layer Neural network (MLP) Probabilistic Neural Network (PNN)
				Signer Dependent	**Recognition Rate**	**Real time**
				no	90.41%	no
12	(Ahmed et al., 2018)	Conference Proceedings	2018	**Classification Category**	**single/double handed**	**Static/Dynamic**
				alphabet	single handed	both
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	Camera	28 characters from three different people training: five videos containing 37 frames (images) and 308 frames (images)
				Preprocessing	**Feature Extraction**	**Classifier**
				Segmentation: Sobel operator	Euclidean distance slop	k-mean k-mediod ANN
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	92.95%	no
13	(Hayani et al., 2019)	Conference Proceedings	2019	**Classification Category**	**single/double handed**	**Static/Dynamic**
				multiple (alphabet/numbers)	single handed	static
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	Camera	2030 images of numbers (from 0 to 10) and 5839 images of 28 letters of Arab sign language,
				Preprocessing	**Feature Extraction**	**Classifier**

					CNN	CNN KNN SVM
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	90.02%	no
14	(Kamruzzaman, 2020)	Journal	2020	**Classification Category**	**single/double handed**	**Static/Dynamic**
				alphabet	single handed	static
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	Camera	31 letters of Arabic sign language 100 images (training) and 25 (test) for each hand sign
				Preprocessing	**Feature Extraction**	**Classifier**
				augmentation	CNN	CNN
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	90%	no
15	(Latif et al., 2020)	Journal	2020	**Classification Category**	**single/double handed**	**Static/Dynamic**
				alphabet	single handed	static
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	Camera	54,000 images of the ArSL alphabets (40 people for 32 standard Arabic signs) 20% testing 20% validation 60% training
				Preprocessing	**Feature Extraction**	**Classifier**
					CNN	CNN
				Signer Dependent	**Recognition Rate**	**Real time**
				no	97.60%	yes
16	(Aly and Aly, 2020)	Journal	2020	**Classification Category**	**single/double handed**	**Static/Dynamic**
				Isolated words	Double handed	Dynamic
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	Camera	23 isolated Arabic word signs performed by three different users (150 sequence for each word)
				Preprocessing	**Feature Extraction**	**Classifier**
				Segmentation: DeepLabv3+	SOM model	BiLSTM
				Signer Dependent	**Recognition Rate**	**Real time**
				no	89.50%	no
17	(Alnahhas et al., 2020)	Journal	2020	**Classification Category**	**single/double handed**	**Static/Dynamic**
				Isolated words	Both	Dynamic
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	Leap motion sensor	44 signs (29 one handed and 15 two handed) performed by 5 signers 80% training and 20% testing
				Preprocessing	**Feature Extraction**	**Classifier**
					Euclidean coordinates	LSTM
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	89% for one handed 96% for two handed	no
18	(Sulaiman et al., 2021)	Conference Proceedings	2021	**Classification Category**	**single/double handed**	**Static/Dynamic**
				multiple (alphabet/numbers/isolated)	both	dynamic
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	Microsoft Kinect	150 signs (3 signers performed each 50 times)
				Preprocessing	**Feature Extraction**	**Classifier**
				Gaussian mixture model in the HSV	CNN (AlexNet)	LSTM

				Signer Dependent	**Recognition Rate**	**Real time**
				Both	95.9% (signer-dependent) 43.62% (signer-independent)	no
19	(Bencherif et al., 2021)	Journal	2021	**Classification Category**	**single/double handed**	**Static/Dynamic**
				multiple (alphabet/numbers/isolated)	both	dynamic
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	2 Microsoft Kinect devices, 1 camera	80 static and dynamic signs by 40 signers (repeated 5 times each)
				Preprocessing	**Feature Extraction**	**Classifier**
					2d skeletal data	Skeletal CNN
				Signer Dependent	**Recognition Rate**	**Real time**
				Both	Dependent mode: 98.39% (dynamic) 88.89% (static) Independent mode: 96.69% (dynamic) 86.34% (static)	yes
20	(Hamou and Chelali, 2021)	Conference Proceedings	2021	**Classification Category**	**single/double handed**	**Static/Dynamic**
				alphabet	single handed	static
				Data Acquisition Approach	**Collection Tools**	**Data**
				Public		Halawani Arabic Sign Language(ArSL) 50% training
				Preprocessing	**Feature Extraction**	**Classifier**
					LPQ MRELBP	SVM Radial Basis Function Neural Network (RBF-NN)
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	RBF-NN with MRELBP = 81.33% RBF-NN with LPQ = 75.67%	no
21	(Tharwat, Ahmed and Boualleg ue, 2021)	Journal	2021	**Classification Category**	**single/double handed**	**Static/Dynamic**
				alphabet	single handed	static
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	camera	4 datasets of the alphabet taken under different conditions (in total 9240 images of Arabic alphabet from 10 locations in different age groups)
				Preprocessing	**Feature Extraction**	**Classifier**
						C4.5 Naive-Bayesian KNN multilayer perceptron (MLP)
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	94.5357	no
22	(Boukdir et al., 2021)	Journal	2021	**Classification Category**	**single/double handed**	**Static/Dynamic**
				Isolated words	Double handed	Dynamic
				Data Acquisition Approach	**Collection Tools**	**Data**
				-	-	Moroccan SignLanguage (MoSL) dataset (224 videos from 56 signs)
				Preprocessing	**Feature Extraction**	**Classifier**
					2DCRNN 3DCNN	2DCRNN 3DCNN
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	92% for 2DCRNN 99% for 3DCNN	no
23	(Alani and	Journal	2021	**Classification Category**	**single/double handed**	**Static/Dynamic**
				alphabet	single handed	static

	Cosma, 2021)			**Data Acquisition Approach**	**Collection Tools**	**Data**
				Public		ArSL2018 (54,049 images by more than 40 signers for 32 classes) 60% training, 20% validation, 20% testing
				Preprocessing	**Feature Extraction**	**Classifier**
				SMOTE		CNN
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	97.29%	No
24	(Ismail, Dawwd and Ali, 2021)	Journal	2021	**Classification Category**	**single/double handed**	**Static/Dynamic**
				multiple (alphabet/numbers)	single handed	static
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	camera	220,000 images for 44 categories: 32 letters, 11 numbers (0:10), and 1 for none. 80% training, 10% testing, 10% validation
				Preprocessing	**Feature Extraction**	**Classifier**
				augmentation	CNN	CNN pre-trained models single model & multi-model
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	100%	yes
25	(Yousuf, Alwarfali and Ighneiwa, 2021)	Conference Proceedings	2021	**Classification Category**	**single/double handed**	**Static/Dynamic**
				Isolated words	single handed	Dynamic
				Data Acquisition Approach	**Collection Tools**	**Data**
				Sensor (Self-acquired)	EMG	5 words
				Preprocessing	**Feature Extraction**	**Classifier**
						ANN
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	90.66%	no
26	(Ritonga et al., 2021)	Journal	2021	**Classification Category**	**single/double handed**	**Static/Dynamic**
				alphabet	single handed	static
				Data Acquisition Approach	**Collection Tools**	**Data**
				Sensor (Self-acquired)	DG5-V Hand data gloves	30 alphabets (125 images each) 80% training, 20% testing
				Preprocessing	**Feature Extraction**	**Classifier**
				augmentation	CNN	CNN
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	90%	no
27	(Alawwad, Bchir and Maher, 2021)	Journal	2021	**Classification Category**	**single/double handed**	**Static/Dynamic**
				alphabet	single handed	static
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	camera	15,360 images of the alphabet 60% training, 20% validation, 20% testing
				Preprocessing	**Feature Extraction**	**Classifier**
				Segmentation: region-based	R-CNN	Faster Region-based Convolutional Neural Network (RCNN) VGG-16 and ResNet based
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	93%	no
28	(Alshomrani, Aljoudi and Arif, 2021)	Journal	2021	**Classification Category**	**single/double handed**	**Static/Dynamic**
				alphabet	single handed	static
				Data Acquisition Approach	**Collection Tools**	**Data**
				Public		ArSL2018 41280 are used, equal distrubution for 32 alphabets

						50% training, 20% validation, 30% testing
				Preprocessing	**Feature Extraction**	**Classifier**
				augmentation	CNN	CNN
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	96.40%	no
29	(Luqman, El-Alfy and BinMakhashen, 2020)	Journal	2021	**Classification Category**	**single/double handed**	**Static/Dynamic**
				alphabet	single handed	static
				Data Acquisition Approach	**Collection Tools**	**Data**
				Public		ArSL2018 50% training, 20% validation, 30% testing
				Preprocessing	**Feature Extraction**	**Classifier**
				augmentation	CNN	CNN
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	99%	no
30	(Dabwan and Jadhav, 2021)	Conference Proceedings	2021	**Classification Category**	**single/double handed**	**Static/Dynamic**
				multiple (alphabet/isolated)	-	static
				Data Acquisition Approach	**Collection Tools**	**Data**
				Public		Yemeni Sign Language (but actually user ArSL2018 along with 4 words) 80% training, 20% testing
				Preprocessing	**Feature Extraction**	**Classifier**
				augmentation	CNN	CNN
				Signer Dependent	**Recognition Rate**	**Real time**
				yes	94%	no
31	(Abdul et al., 2021)	Journal	2021	**Classification Category**	**single/double handed**	**Static/Dynamic**
				multiple (alphabet/numbers/isolated)	both	both
				Data Acquisition Approach	**Collection Tools**	**Data**
				Vision (Self-acquired)	camera/Microsoft Kinect	KSU-ArSL 40 dynamic and 39 static words (5 times each from 40 signers)
				Preprocessing	**Feature Extraction**	**Classifier**
						Inception-BiLSTM
				Signer Dependent	**Recognition Rate**	**Real time**
				Both	78.90%	yes
32	(Sadeddine et al., 2021)	Journal	2021	**Classification Category**	**single/double handed**	**Static/Dynamic**
				alphabet	single handed	static
				Data Acquisition Approach	**Collection Tools**	**Data**
				Public		halwani (30 hand gestures)
				Preprocessing	**Feature Extraction**	**Classifier**
					Gradient Local Auto-Correlation (GLAC) Gabor Wavelet Transform (GWT) Fast Discrete Curve Transform (FDCT)	Individual: KNN, MLP, PNN Combined: Majority Voting (MV), Weighted Sum (WS)
				Signer Dependent	**Recognition Rate**	**Real time**
				both	95.83 (KNN)	no

Table 3 Extracted Article Data

2.2 Results

2.2.1 Synthesis of Data

In this section, we analyze our papers in detail using our research questions. This will help us to identify gaps in the research. We first take a quick look at paper distribution according to publication type and years in figures 2 and 3. We can see that papers are mostly journal papers and were there was a substantial increase in research papers in 2021. This emphasizes that there is an increased interest in deep learning approaches to Arabic Sign Language.

Figure 2 Publication Type

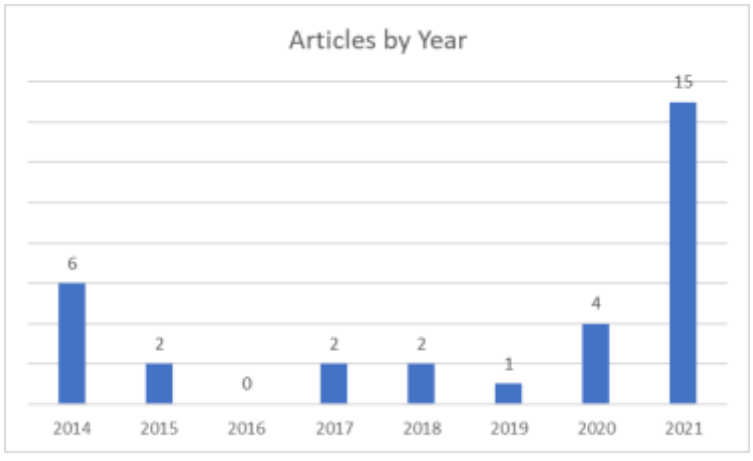


Figure 3 Articles by Year

Figure 4 reviews the distribution of the articles by year and publication type. We see that until 2020, research was similar between journals and conference papers. In 2021, journal papers had the largest number across years and article types.

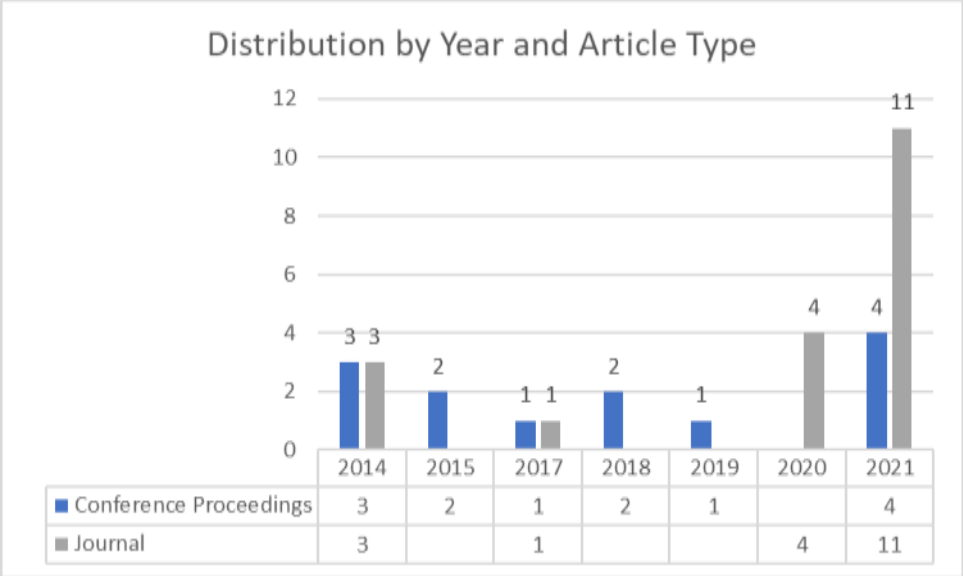


Figure 4 Distribution by Year and Article Type

2.2.2 Research Questions

RQ1: What are the existing studies on Arabic Sign Language Recognition using a deep learning approach?

As there is a total of 32 studies included in our review, we group research articles by themes and notable findings in order to streamline our observations and present the information in a manner that is easy to follow.

Vision-Based Recognition

(ALI, et al., 2014) attempts applying Multi-layer perceptron Neural Network (MLP) to the task of ArSL recognition. They use a dataset containing 5 two-handed signs with 200 samples each, with 150 samples for training, and 50 for testing. They achieve a recognition rate of 87.6%.

(Samir, Aboulela and Tolba, 2014) Compares two neural networks, Multi-Stage Multiplicative Neural Network (MMNN) and MLP on 100 signs, and finds that MMNN performed better than traditional MLP in both training time and accuracy, achieving an accuracy of 94%. Another interesting finding is the use of graph matching as a technique to recollect static postures to form dynamic gestures.

(Al Mashagba, Al Mashagba and Nassar, 2014) introduces the first paper attempting a vision-based recognition system using Time Delay Neural Network (TDNN) with a recognition rate of 70% on 40 isolated word signs collected using coloured gloves and a camera.

(Elons, 2014) presented a Graphics Processing Unit (GPU) based approach to Arabic sign language real-time recognition. They use a PCNN to extract static postures features and utilize a Multi-level Multiplicative Neural Network as a classifier, which achieved a better result when compared with traditional Multi-layer perceptron Neural Network in both training time and accuracy. The system contains two layers: the first determines if the signer is using one or two hands while the second determines the final classification. The system was applied on a dataset of 200 signs achieved an 83% recognition accuracy during testing.

(Elons et al., 2014) uses "Leap Motion Controller", which was considered a new digital sensor at the time of publishing the paper. The aim is to use this vision-based sensor to eliminate the major issues in vision-based systems such as skin colour, lighting, etc. The sensor tracks hands and finger positions and movements in frames and captures them as 3D information. It uses two features: finger positions and finger positions distances. Then, the temporal and spatial features are used as input for a Multi-layer perceptron Neural Network (MLP) to output sign meaning. The system was tested on 50 different dynamic signs from two different people and the recognition accuracy reached 88%. The system could benefit of using another device to track different features such as orientation and non-manual features.

(Mohandes, Aliyu and Deriche, 2014) Similarly uses Leap Motion to create a dataset of 28 alphabets from a single user, where each letter has 10 samples, and each of these samples has 10 frames, for a total of 2800 frames. Upon analysing the data in MATLAB, 23 values were returned, of which 12 features were selected. Two models are compared for classification: Naive Bayes Classifier (NBC) and a Multi-layer Perceptron (MLP) trained by the back-propagation algorithm. The MLP model achieved a higher accuracy of 99.1% compared to NBC's accuracy of 96.4%. Upon analysing the misclassified letters, it was found that the issue was not due to similarity between the signs, but rather the LMC field of view caused some of the fingers to be obscured by other fingers or the hand palm. This can be solved by using another device simultaneously from another angle, but further study is needed on how to combine the features from the two devices. We explore the use of multiple devices further below.

Multiple Collection Devices

Both (Bencherif et al., 2021) and (ElBadawy et al., 2015) utilize extra devices for to solve this issue. (Bencherif et al., 2021) uses 2 Microsoft Kinect devices along with a camera to collect their data. They take a different approach by feeding their data into an OpenPose library to extract skeletal data and feed the 2D information into a skeletal CNN for recognition. Simplifying the CNN network to take 2D points as input instead of the original 3D points lead to cutting a third of the computational power required for the recognition task. Normally, there's a pay-off between recognition accuracy and time, but this approach contributes to solving this issue.

(ElBadawy et al., 2015) use one Leap motion sensor to capture hands and fingers movements, and two cameras to capture facial expressions and body movement each. They note that Kinect is a promising sensor but doesn't offer the same flexibility as Leap motion sensor in capturing

single fingers movements. Adding the non-manual features raise the recognition rate from 90% to 95% for 20 dynamic word signs.

Feature Extraction

(Sadeddine, Chelali and Djeradi, 2015) takes a different approach focusing the effect of feature extraction techniques on the recognition rate by combining Discrete Wavelet Transform (DWT) and Principal Components Analysis PCA using a Probabilistic Neural Network (PNN). They find that this achieves better results than using one feature alone upon applying the system to ArSL and ASL datasets.

(Sadeddine et al., 2018) compares the performance of different descriptors for feature extraction such as: Hu's invariant moments, Local Binary Pattern (LBP), Zernike moments, and Generic Fourier (GFD) and applies them to a MLP and PNN for classification in different combinations. They find that the fusion of the descriptors improves performance when compared to other related works and allows better flexibility and processing time. (Sadeddine et al., 2021) performs a similar study comparing Gradient Local Auto-Correlation (GLAC), Gabor Wavelet Transform (GWT), and Fast Discrete Curve Transform (FDCT). They use KNN, MLP, and PNN for classification in addition to combines classifiers. They note that running the descriptors in parallel improves systems performance as well.

(Hamou and Chelali, 2021) similarly tests the performance of Local Phase Quantization (LPQ) and Median Robust Extended Local Binary Pattern (MRELBP) when applied to SVM and Radial Basis Function Neural Network (RBF-NN). The authors note that the SVM model outperforms the RBF-NN in both cases, with the MRELBP performing better. We must note that the paper applied the models on multiple datasets other than the ArSL dataset, which is

static. Achieved results are similar but when applied to dynamic gestures, the RBF-NN performs better than SVM.

Deep Learning for both Feature Extraction and Classification

The authors of (ElBadawy et al., 2015) go on to focus on the feature extraction aspect of a recognition system in (ElBadawy et al., 2017) where they explore the use of a 3D convolutional neural network (CNN) as a technique for feature extraction and classification. They use depth maps as input and are able to achieve a 98% accuracy for observed data and 85% for new data, which can be improved with more data from different users.

(Boukdir et al., 2021) is another paper that utilizes a deep learning model for both feature extraction and classification, in order to solve the problem of having to decide on which features to use in traditional methods. They use two models, a 2DCRNN and a 3DCNN model: in the 2DCRNN model, the features are extracted by detecting the relationship between video frames using a recurring network pattern while the 3DCNN model learns the spatio-temporal features out of small patches. The main aim is to compare how each of this model can be applied to the dataset. The 3DCNN outperforms the 2DCRNN, where they achieve an accuracy of 99% and 92%, respectively.

(Alnahhas et al., 2020) focuses on analysing the mathematical features extracted from the LMC sensor and enhancing the system with the use of LSTM network for multiclass ArSL classification, which is suitable for temporal data as the sign gestures are represented as a frame sequence. Their model is able to achieve promising results for dynamic gestures.

(Hayani et al., 2019) Similarly uses a CNN model inspired by LeNet-5, a 7-layer CNN architecture used in the banking sector that can process higher quality images with more layers. The model is used for both feature extraction and classification and is able to achieve a 90.02%

accuracy by increasing the training data to 80% of the overall dataset, proving that deep learning models perform better with data increase. It also performs better compared to KNN and SVM.

Pre-processing

Interestingly, (Kamruzzaman, 2020) who also use a CNN for feature extraction and classification was able to improve the accuracy to 90% by using data augmentation during the pre-processing stage, which creates artificial images constructed from the real images to increase and diversify the dataset.

To compare, (Latif et al., 2020) is another paper utilizing a CNN model who notes that increasing the training number of data does increase performance until a certain point, after which the size will have little effect on the accuracy using their model architecture that achieved 97.6% accuracy. However, we must note that (Latif et al., 2020) is training on a 50,000 dataset while (Kamruzzaman, 2020) is training on 100 images. They do add that in future research, they plan to use augmentation techniques to increase their data size, noting that the increase would require more computational power and processing time.

(Alshomrani, Aljoudi and Arif, 2021) does apply augmentation on the dataset they used to test their CNN model, which is publicly available. Their approach ensures that all classes have the same amount of images for features to be extracted from, and are able to achieve an accuracy of 96.4%. (Luqman, El-Alfy and BinMakhashen, 2020) applies a similar approach with augmenting their dataset and add a Gabor filter for enhancing the feature extraction to their CNN model, achieving an accuracy of 99%. (Dabwan and Jadhav, 2021) follows the same approach. They claim to be applying their model to further research on Yemeni Sign Language (YSL), however, they mostly train the model on the ArSL alphabet dataset in addition to 4 other signs from the YSL dataset. They achieve an accuracy of 94%.

On that note, (Alani and Cosma, 2021) discussed the effect of imbalanced training samples for each class on the model overall accuracy. They use the same dataset as (Alshomrani, Aljoudi and Arif, 2021) and (Luqman, El-Alfy and BinMakhashen, 2020), and note that the sample number, or images, available for each letter varies considerably. Due to the nature of deep learning models, the more available training data, the better the model can perform. Thus, this imbalance would need to be resolved to achieve better results. Thus, resampling methods including oversampling and undersampling have been applied. The paper finds that applying the synthetic minority oversampling technique (SMOTE) achieved the best results by increasing the performance from 96.59% to 97.29%. They also prove that this result is statistically significant.

Deep Learning vs Statistical Models

(Hisham and Hamouda, 2017) introduces the most ambitious paper in terms of the classification category. They test their model on alphabets, numbers, and isolated words, and single handed and two handed signs. 38 static gestures: 28 letters, numbers (1:10) and 16 static words, 20 dynamic gestures. They collect the data through LMC and use ANN with Multilayer Perceptron, SVM, KNN, and DTW. We notice that ANN was not the best performer and generally achieved good results for static signs and performs lower with more complex signs. Another contribution of the paper is a segmentation approach that can be applied to continuous sentences in real time for ArSL recognition with promising results.

Similarly, (Ahmed et al., 2018) compares ANN with K-means and K-centroid classifiers on an ArSL alphabet dataset and notes that ANN had the lowest performance: 93.3673% (k-mean), 93.1354% (k-medoid) and 92.9499% (ANN). Although we note that this paper does not mention using a separate training and testing dataset and only mention their training set, which does not accurately represent the model performance.

Another comparison between NNs and statistical models in (Tharwat, Ahmed and Bouallegue, 2021), where an MLP is compared against statistical machine learning models C4.5, Naive-Bayesian, and KNN. The KNN model performed the best, with 99.5% accuracy. However, we must note that increasing the neurons used in the MLP model increases the accuracy, while increasing the time it takes to build the model.

Pre-trained Models

Another ambitious paper is (Ismail, Dawwd and Ali, 2021) where they compare the performance of multiple pre-trained deep learning models on a dataset of 220,000 images. This is achieved using data augmentation to reduce overfitting and increasing the dataset size by 48 times. Altogether, they compare the performance of 8 single models and a multi-model made from combining two single models (DenseNet121 model and VGG16 model) and training them in parallel at the extraction stage and then combining them when classifying. Both the single DenseNet121 model and multi-model achieve 100% accuracy with single VGG16 model following close behind at 99.97%. This is the highest accuracy achieved in all our researched papers, which we can attribute to the huge amount of images in the dataset and the training subset (80%) compared to the validation and testing subsets (10% each).

(Alawwad, Bchir and Maher, 2021) applies a Faster Region-based Convolutional Neural Network (RCNN) model based on deep VGG-16 and ResNet architectures as well. The basis of this model is to be able to recognize the hand sign as the region of interest even with different lighting, background, and skin tones. The model is then able to learn the hand position, extract and map the features, which in turn helps with the issue of having to choose relevant features. Both models performed similarly. Achieving 93% accuracy but the authors note that ResNet achieved its results with considerably less training time.

Signer Dependent vs Independent Testing

(Aly and Aly, 2020) addresses the problem of signer-independent sign language recognition through a combination of DeepLabv3+ semantic segmentation, SOM model for feature extraction, and a deep BiLSTM model for recognition. They note that the use of their segmentation and feature extraction models are able to extract hand information from video frames efficiently under different conditions and from different users. They even note that removing the segmentation model decreases the performance of their framework from 89.5% to 69.0%, emphasizing its importance to the success of their model.

(Suliman et al., 2021) utilizes an LSTM model for the recognition of a variety of signs including numbers, alphabets, and isolated words. The data was collected using Microsoft Kinect which follows a similar approach to the LMC device. They test their model using two approaches: singer-dependent and signer-independent. They find that their model was able to achieve an accuracy of 95.9% for the first and 43.62% for the latter. According to the authors, this is to be expected considering the significate increase in complexity when the model is applied to unseen data and the similarity between some signs.

(Abdul et al., 2021) are able to achieve a 78.9% accuracy for the singer-independent inception-BiLSTM model, which was trained on a KSU-ArSL dataset made of both static and dynamic signs. Their inception model is able to extract the most informative spatial features from the images with the BiLSTM extracting the temporal features. They note the bidirectional model is able to perform better than a normal LSTM model with more efficiency and within less time.

Sensor-Based Recognition

(Yousuf, Alwarfalli and Ighneiwa, 2021) is the first and only paper to use Electromyography (EMG) and apply ANN model for classification. EMG is a sensor-based method of acquiring

data using finger motions by taking the measurement of the electrical pulse in human muscle. Intelligent EMG is the application of an intelligent method to the EMG to improve classification. They apply their model on 5 isolated words, however, they are unable to test it in real time due to the difficulties faced in collecting big data.

(Ritonga et al., 2021) describes another sensor-based method of acquiring data using DG5-V Hand data gloves, which capture features like finger bending, hand orientation, position and rotation. They utilize a CNN model for both feature extraction and classification and achieve an accuracy of 90%. The paper notes that this model is in its early stages and could benefit from adding other devices link LMC or XBOX connect.

RQ2: How was the task of Arabic Sign Language Recognition approached?

Researchers have applied many frameworks, methods, and approaches in this task. Based on our research, there are two main approaches; one is based on the data's acquisition method, and the other on the classification category. We will elaborate on both below.

1. **Acquisition Method** relates to how the data, or the signs, were collected. There are two main approaches:
    a. Vision-Based Recognition: the data consists of static and dynamic images (as a video). The devices used for collection range from simple cameras to optical tracking devices such as a Leap Motion Controller or Kinect, which can track movement.
    b. Sensor-Based Recognition: data like the movement, velocity, and location of the hand is obtained from sensor devices. Devices include Inertial Measurement Unit (IMU), Electromyography (EMG), Wi-Fi and Radar.

The main difference between the two approaches is how the device collects the data. In the vision-based approach the device collects visual data while the sensor-based approach collects signals through its sensors.

c. Public Datasets: A third approach is to simply use a dataset that is publicly available, such as Halawani Arabic Sign Language (ArSL) (Al-Jarrah and Halawani, 2001) or ArSL2018 (Latif et al., 2019) datasets.

Figure 5 displays the percentage of chosen data acquisition methods utilized by researchers. We see that vision-based is the most chosen approach followed by datasets while sensor-based approaches are the lowest at 6%.

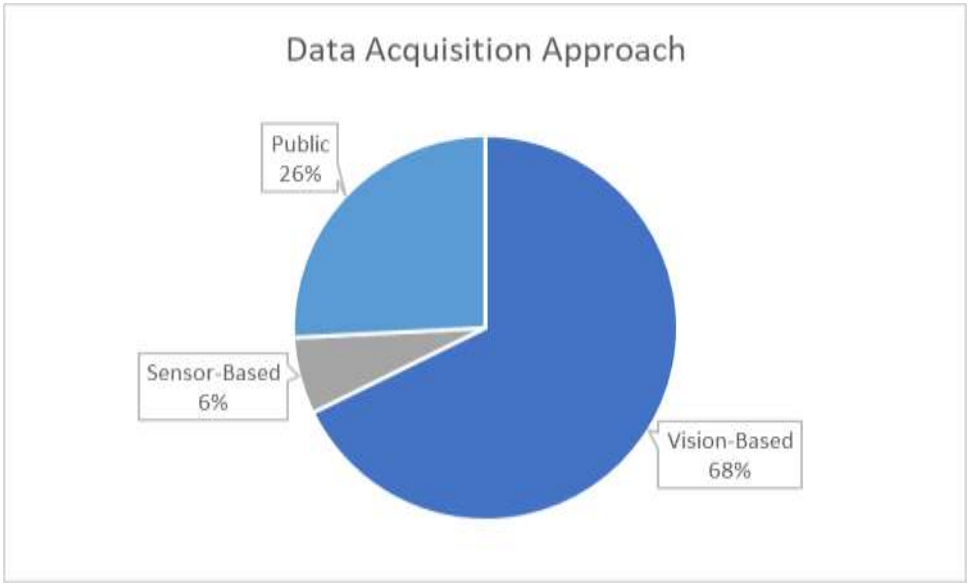


Figure 5 Data Acquisition Approach

For each Acquisition method, a variety of tools can be utilized. In the case of Vision-Based collection tools, we have:

- Camera: a camera device is used to capture images or videos.

- Leap Motion Controller (Fig. 6): an infrared light-based stereoscopic camera that tracks hands information. The software will then analyze the 2D frames and map it to a 3D representation of the hand position.

Figure 6 Leap Motion Controller (UltraLeap, n.d.)

- Microsoft Kinect (Fig.7): an infrared light-based device that utilizes 3D depth cameras. Originally created by Microsoft for Xbox, the device is able to distinguish body movement and facial features.

Figure 7 Kinect (Microsoft, n.d)

Figure 8 showcases the percentage of each collection tool. As we can see, camera devices are the most popular method, followed by LMC. The Microsoft Kinect device is not as popular and may be used along with another device like a camera.

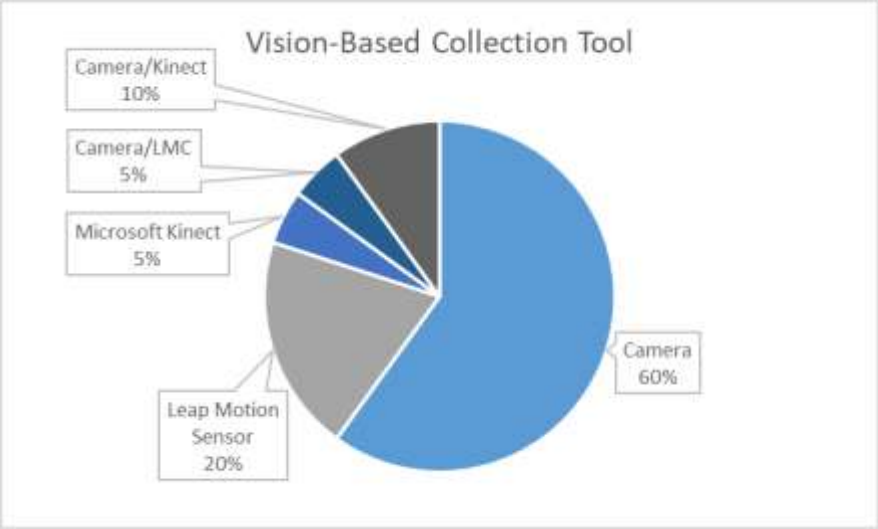


Figure 8 Vision-Based Collection Tool

For Sensor-Based recognition, we only have two articles that utilized Sensor-Based collection tools, the two tools used are:

- Electromyography (EMG): detects finger motion through the electric signal created by muscle movement.
- DG5-V Hand data gloves (Fig. 9): sensors are attached to the gloves the signer wears. The sensors will extract the required features such as hand movement, rotation, finger bending, etc.

Figure 9 DG5 V Hand (DG Tech, n.d.)

Below table 4 shows the advantages and disadvantages of each acquisition approach and collection device as we observed.

	Advantages	**Disadvantages**
Vision-Based		
Camera	• Low cost • Convenient/ease of use and setup	• Greatly affected by environment, skin color, light conditions • Requires pre-processing to make use of images • Needs large amount of data to train model
Leap Motion Controller	• Suitable for hand and finger tracking • High accuracy and speed • Small and portable	• Image appears in grayscale • Sensitivity to infrared light, thus must be used indoors • Requires additional software to analyze images
Microsoft Kinect	• Suitable for whole body tracking • Can distinguish facial features • Big range • Can work in the dark	• Sensitivity to infrared light, thus must be used indoors • Can be affected by complex backgrounds and noise
Multiple Devices	• Collect multiple data at the same time (one device for hands and the other for body movement, etc) • Mitigate the issues with one device	• Extra cost and setup • Need of pre-processing and method of combining the data from two devices to allow feature extraction and classification
Sensor-Based		
Electromyography (EMG) *DG5-V Hand data gloves*	• Not affected by external environment • Feature extraction is easier compared to vision-based methods • Highly accurate	• Inconvenient • Very expensive, especially on a large-scale data collection
Public Dataset		
	• Saves time and effort in collecting data	• Limitation in data variety

Table 4 Advantages and Disadvantages of Data Collection Methods

Next, we will discuss the second approach in Arabic Sign Language recognition.

Classification Category relates to the complexity of the signs that system will be developed to recognize. As units increase, the difficulty of the task increases. The system could go from trying to recognize simple signs such as alphabets, to more complex signs such as isolated words, to the most difficult task of recognizing full sentences.

1. Alphabet Sign Language Recognition: The signs consist of each alphabet independently. The Arabic Language consists of 28 letters, which is the same case for ArSL. However, researchers might include other orthographical representations of the alphabets, thus increasing the number to 39 signs (Fig. 10).
2. Numbers: The signs consist of each separate number, usually 0-9.
3. Isolated Word Sign Language Recognition: The system will aim to recognize signs that form separate words, and the data may consist of multiple images to perform one sign.
4. Continuous Sign Language Recognition: The system will analyze full continuous sentences, which is a real-life application of Sign Language Recognition. The difficulty lies in developing a system that can detect transitions between one sign and the next within the sentence.

Figure 10 ArSL Alphabet ("The Arabic Dictionary of Gestures for the Deaf," 2007)

Most research focused on Alphabets, followed by Isolated words. In figure 11 below, the "All" category indicates the use of Alphabets, numbers, and Isolated Words. None of our reviewed papers did research on Continuous ArSL recognition, which can be attributed to challenges with detecting transitions in between words to form the sentence. This shows that there is a research gap.

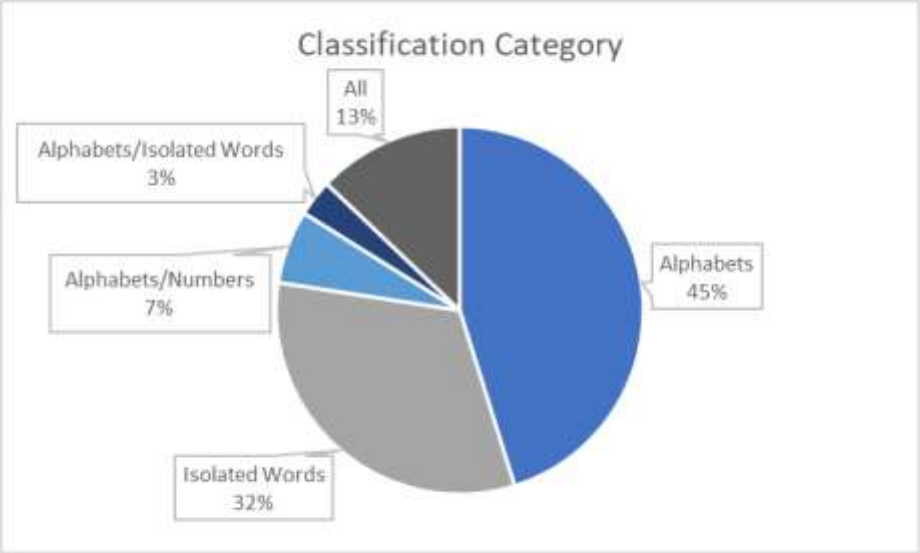


Figure 11 Classification Category

RQ3: What data was used?

As discussed in the previous question, the data can either be collected through vision-based or sensor-based devices, or can be a public dataset. From there, the dataset could be made of signs for alphabets, numbers, isolated words, or a mix of all.

Upon closer inspection, we find that the data can be further classified into the following:

a. Static/Dynamic (Fig. 12)

Static gestures refer to data input of still and unmoving frames. Usually, each frame would consist of a single sign where the hand itself would remain stationary while the finger would change direction and shape. Normally, no movement is involved.

Dynamic gestures on the other hand refer to data input of continuous frames, normally from a video. The position of the hand and fingers is in motion in a given space and time.

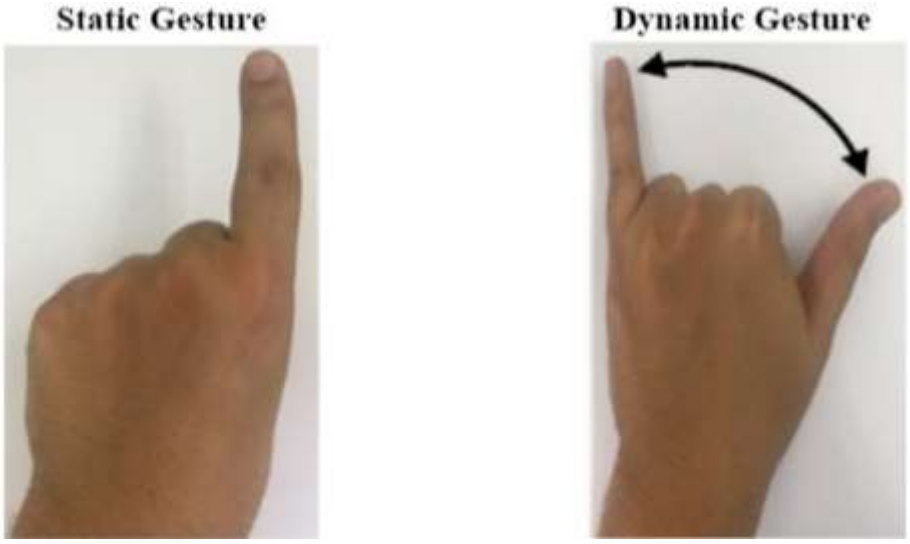


Figure 12 Static vs. Dynamic Gestures (Al-Shamayleh et al., 2020)

b. Sign Language Symbols

According to (Wadhawan and Kumar, 2021), sign language symbols are classified into single handed symbols and double handed symbols. Single handed symbols refer to gestures performed by one hand and double handed symbols refer to signs performed by two hands. Both cases can be static or dynamic.

To have a better idea of the data types used in our research papers, figure 13 displays the distribution of data type across three levels:

1. Static/Dynamic
2. Classification Category
3. Symbols

Based on the chart, we can see that more than half the research is done on static gestures. From there, most research was done on static, single handed gestures of the Alphabet. This can be explained by the fact that it can be easiest to perform pre-processing, feature extraction and classification on still images.

In the dynamic category, we can see that more research has been done on isolated words using both hands. While few research has been done on both static and dynamic data.

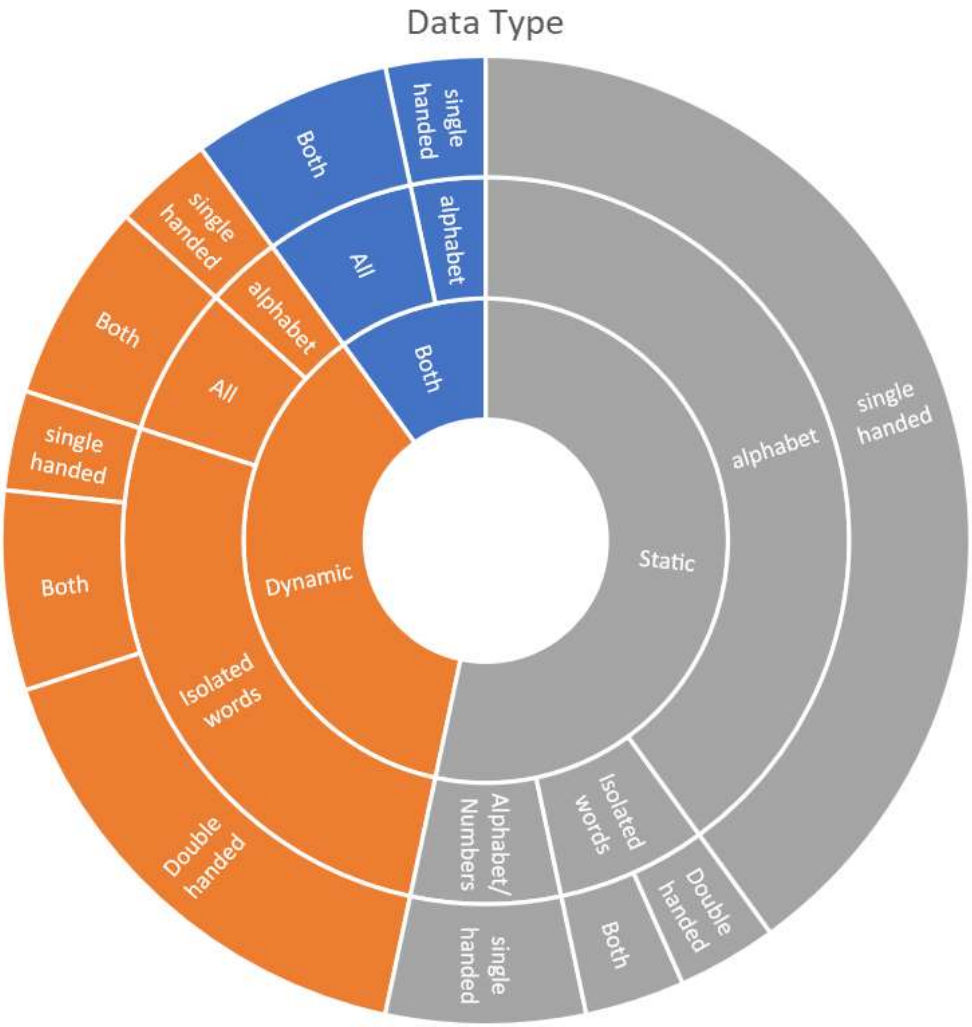


Figure 13 Data Type

Further to this, we would like to briefly discuss some of the public datasets mentioned previously. These include the Halwani ArSL dataset and ArSL2018. Results are summarized in table 5.

Dataset	Papers	Content	Remarks
Halawani Arabic Sign Language (ArSL) (Al-Jarrah and Halawani, 2001)	(Sadeddine, Chelali and Djeradi, 2015), (Sadeddine et al., 2018), (Hamou and Chelali, 2021), (Sadeddine et al., 2021)	gray-scale static images of the alphabet signs (30 gestures) each sign with 60 frames from 60 different people	Balanced but number of data is relatively small. Variation in distance, size, and orientation
ArSL2018 (Latif, 2019)	(Alani and Cosma, 2021), (Alshomrani, Aljoudi and Arif, 2021), (Luqman, El-Alfy and BinMakhashen, 2020), (Dabwan and Jadhav, 2021)	gray-scale static images of the alphabet signs (32 gestures), totaling 54,049 images from 40 different people	Unbalanced dataset: Number of images for each class is not the same, thus would require preprocessing to balance the data. Not enough light variation

Table 5 Public ArSL Datasets

RQ4: Which deep learning methods were used?

Multiple deep learning models were used in the studies. As we discussed the details of each paper in RQ1, below we discuss the main models that were used most frequently according to figure 14.

CNN: In the chart, we can see that Convolutional Neural Networks were the most used model, with some variations of the model. CNN is a feedforward network typically used in computer vision tasks; it is most useful for processing images. As we established that most papers trained their model on static images, it is no surprise that CNN were most used. We can see that different variations were used such as 3DCNN, 2DCNN, RCNN, etc. These will mostly depend

on the kind of input fed to the model, the features involved, and the type of layers used to build the model architecture.

MLP: a Multilayer perceptron network is another feed forward neural network and the second most used network after CNN. It can be useful for complex tasks as it is flexible and can deal with different types of data. It can also be used as a baseline model to test and compare against other, more complex models.

PNN: a Probabilistic neural network is also a feedforward network based on . They are faster than MLPs and can be more accurate but can be slower in classifying new cases and require more memory.

LSTM: Long Short Term Memory network is a type of Recurrent Neural Network, meaning it has an internal memory for processing data sequences through feedback connections that regulate data flow, unlike a CNN or MLP which are feedforward only. LSTMs are especially useful for learning long-term dependencies in sequence prediction tasks.

BiLSTM: Another variation of LSTM that has an additional layer reversing the dataflow. It is useful when wanting to preserve both past and future information for training the model as the data runs in both directions. As such, it is normally applied to Natural Language Processing tasks as it understands context better.

Both LSTM and BiLSTM by (Aly and Aly, 2020), (Alnahhas et al., 2020), (Suliman et al., 2021), (Abdul et al., 2021) to classify dynamic signs with promising results.

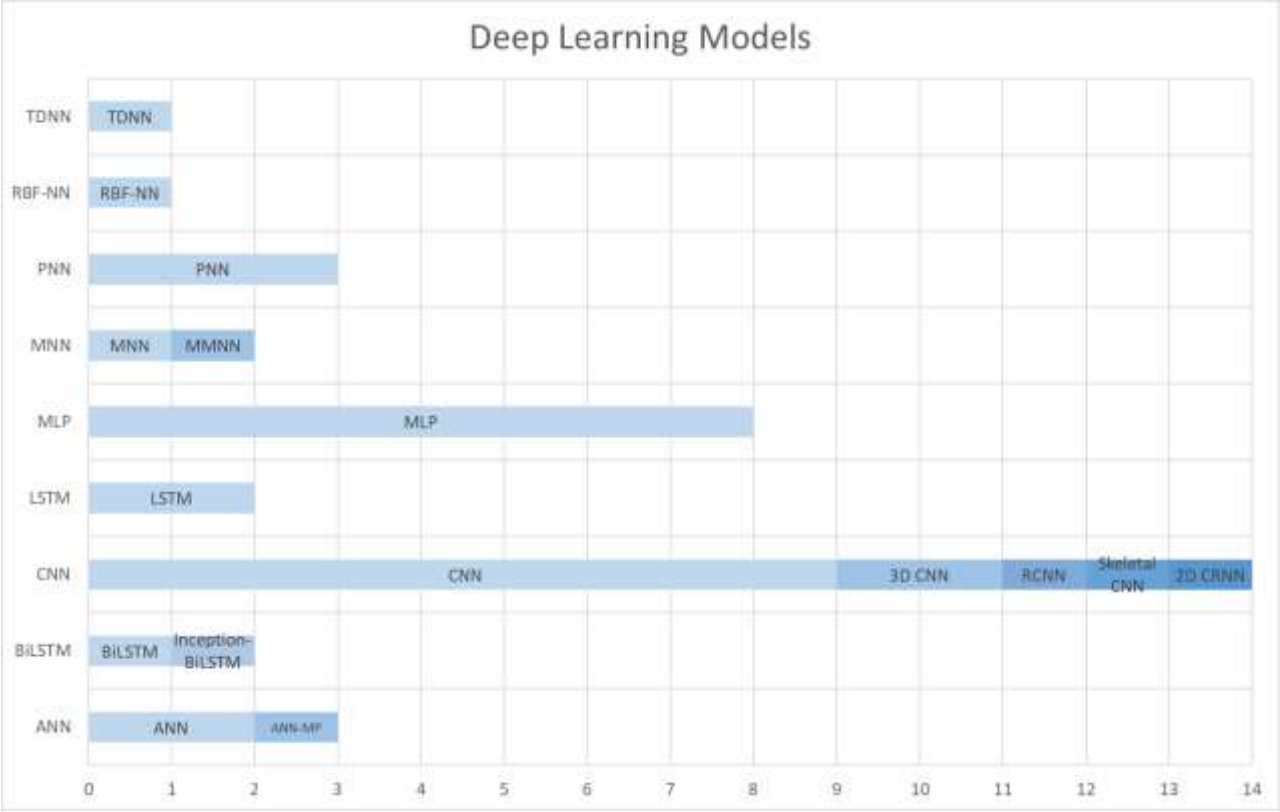


Figure 14 Deep Learning models by type

RQ5: How did the models perform?

a. Recognition Accuracy

In figure 15, we have a chart where we divide the recognition rate into ranges of 70-79, 80-89, 90-95, and >=96. From there we view each model performance and how often they achieved a recognition range within one of these ranges. To better observe the results, variations of a model will be treated grouped together. For example, CNN, 3DCNN, Skeletal CNN, etc, will all be counted under CNN. For details for each model can be viewed in Appendix A at the end.

We can see that CNN models consistently achieve high recognition accuracy. We must again consider that this is because mostly, CNN was tested on static images and is our mostly used model.

Although we cannot generalize due to the small sample, but both ANN and LSTM models achieved high accuracy when applied. In addition, the LSTM was applied on dynamic data.

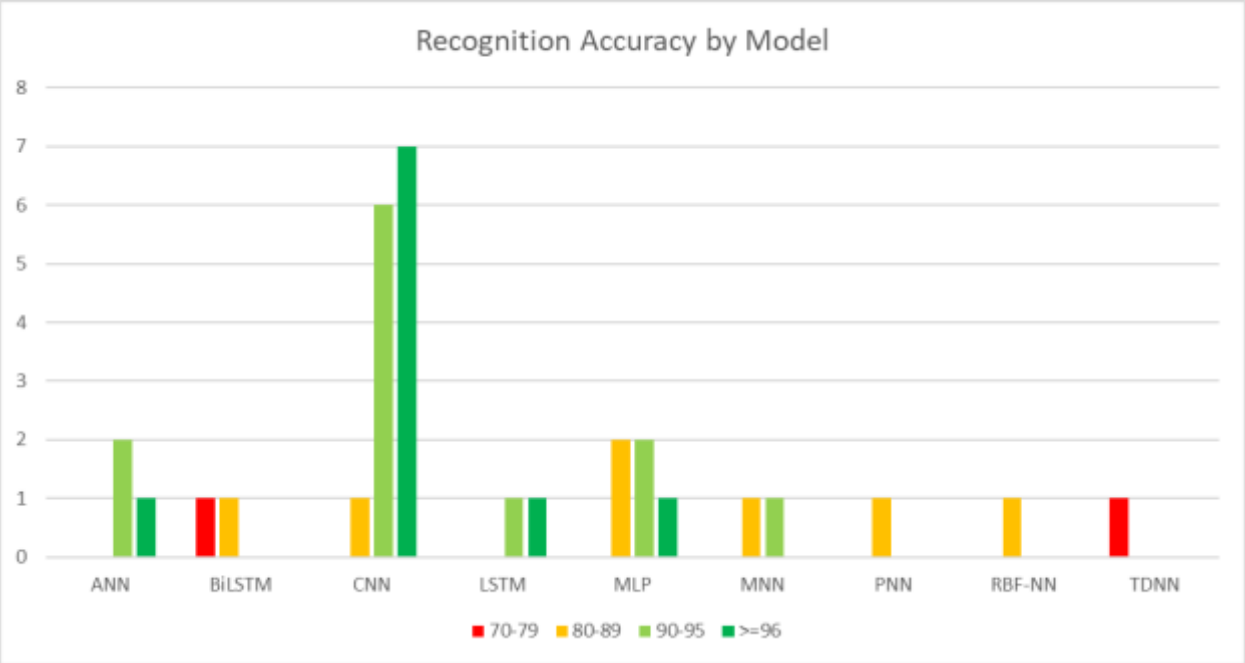


Figure 15 Recognition Accuracy by Model

b. Testing Methods (Signer Independent and Real-Time testing)

Another aspect we would like to discuss is the testing method. Most papers would use the same dataset and split it into training, testing, and sometimes validation datasets. The training data is used to train the mode and then the model is tested on the testing dataset, which contains new and unseen data for the model.

This is a good approach; however, we need to consider that ultimately, these systems should have real-world application. Meaning it should be tested on signers other than the ones the data was trained on. Additionally, the model should be tested in real-time to mimic a real-world scenario.

In figure 16, we can see that most of the papers tested in signer dependent mode while the same number of papers used singer independent mode or both modes of testing. This indicates a research gap in literature.

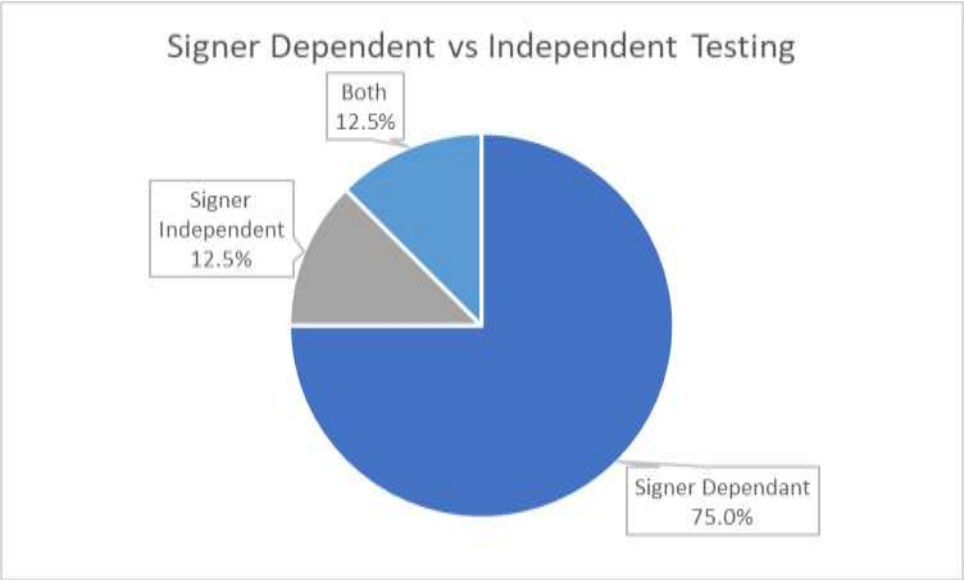


Figure 16 Signer Dependent vs Independent Testing

For real-time recognition, again we can see in figure 17 that most paper do not perform real-time recognition, with only around a fifth of the papers applying it.

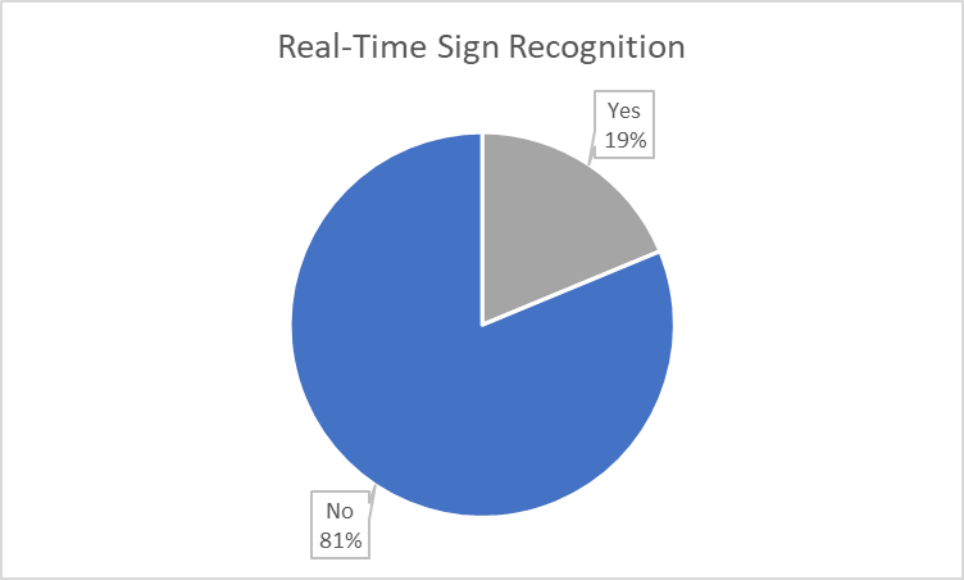


Figure 17 Real-Time Recognition

2.3 Discussion

2.3.1 Implications for research

We can identify common themes across papers in the task of ArSL recognition, including data collection, preprocessing impact, feature extraction, model performance, and testing methods. We further showcase these trends in figure 18 below.

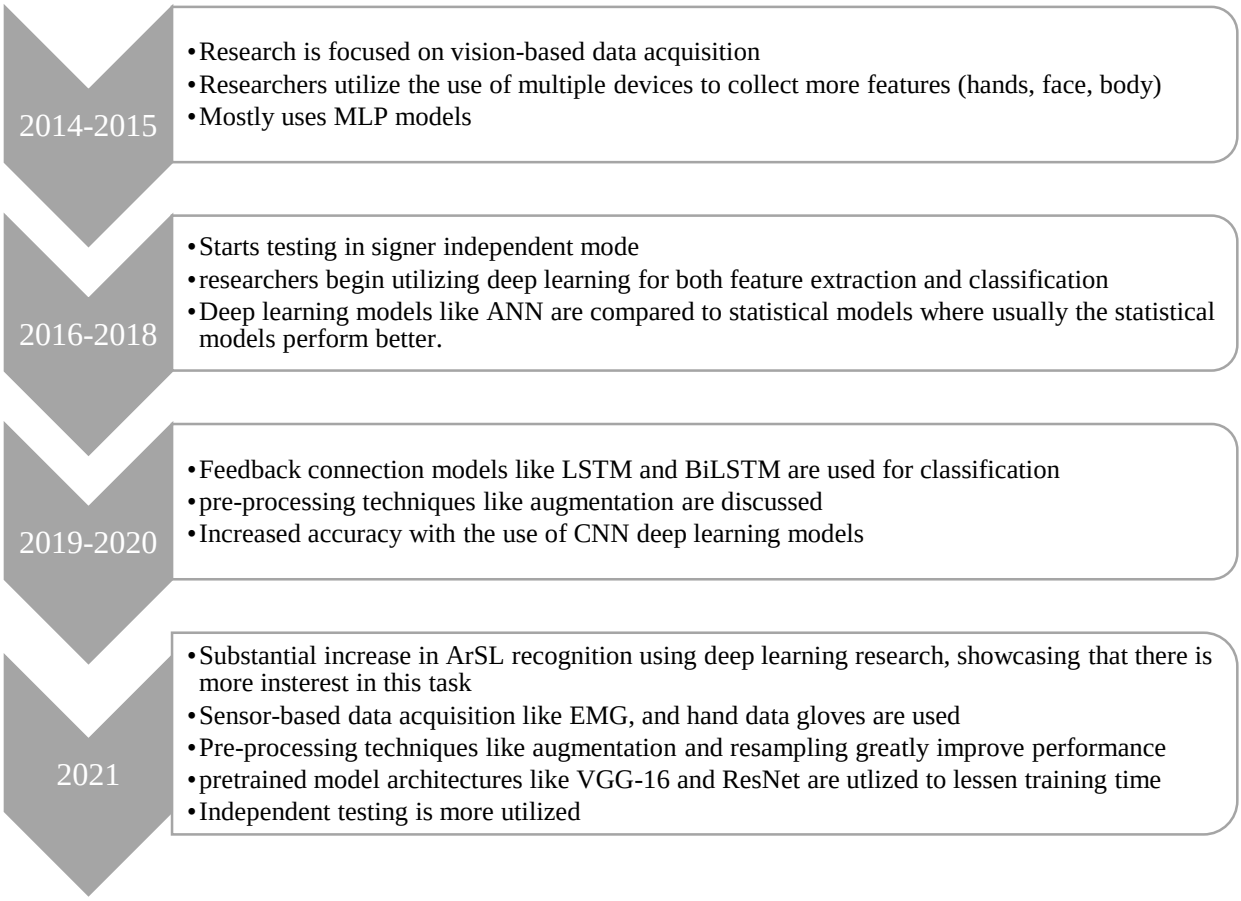


Figure 18 Research Trends over the years

We can see that earlier research was exploring the use of novel devices like the Leap Motion Sensor for data acquisition. This is done in the hopes of solving traditional problems with ArSL data collection via vision-based methods, such as environment noise, lighting, quality of images, etc. As we consider the fact the deep learning needs a huge amount of data to achieve

good results, we realize that the lack of standardized datasets is setting back research in this area of ArSL recognition.

Even within the same dataset, (Hayani et al., 2019) notes that by changing the training-testing data split to include more training data achieves better results. This is normally the case for deep learning models, as they are very powerful and can achieve state-of-the-art results but will need a huge amount of data to achieve this.

Another point observed throughout the studies is the lack of real-world application in the ArSL recognition domain. Meaning that most of the research is focused on improving the classification accuracy of the deep learning models in the task of ArSL recognition, usually through improving data acquisition, feature extraction or the classification techniques. Few papers clearly state that their model has been applied to real-time recognition. Researchers might choose to deploy their models to a web or mobile application to try and achieve this.

We also note that most of the papers were signer dependent in that the data used for training and testing were collected from the same signers. To truly assess the model performance and imitate real world application, researchers would need to test their system on a separate dataset of signers that were not involved in the testing dataset.

Many researcher papers like (Boukdir et al., 2021) utilized the use of deep learning for feature extraction and not just classification. Interestingly, they are also the first paper to use a dataset that includes dialectal words in Moroccan Sign Language. Similarly, (Dabwan and Jadhav, 2021) uses isolated words from Yemeni Sign Language.

2.3.2 Limitations

The most pressing issue in ArSL recognition is the lack of a benchmark dataset. The problem is that most research is still focused on alphabet classification, mainly using static images. In

addition, creating a dynamic dataset has its own difficulties as you will need to either record many videos from many participants or utilize the use of sensor-based devices, which can be quite expensive.

(Bencherif et al., 2021) notes that the lack of a standardized dataset makes the task of comparing recognition models even harder. This is because researchers may use similar models but the data itself used for training and testing can differ in their collection method, amount, environment in which it was collected, extracted features, and many others. Meaning researchers are trying to solve the same issue but with no golden standard to compare their results against.

Even public datasets like Halawani and ArSL2018 have their limitations as explained previously. For example, ArSL2018 has unbalanced data for each class in the dataset. This will then require researchers to apply different pre-processing techniques like resampling to properly train the model as the case with (Alani and Cosma, 2021). If different researchers apply different preprocessing techniques we can assume that each model would train on a slightly different subset of the data. Thus comparing these models is not really accuracte.

Many researchers had to resort to data augmentation to alleviate their dataset and be able to train their model to an adequate level. Papers like (Kamruzzaman, 2020) and (Latif et al., 2020) created artificial images from the original dataset to diversify their dataset, resulting in a notable increase in performance. (Latif et al., 2020) states that increasing the number of data only has an effect on improving performance until a certain point, after which the model performance won't be affected. However, considering that (Latif et al., 2020) uses a 50,000 dataset compared to (Kamruzzaman, 2020) who uses a dataset of 100, we note small datasets can benefit from augmentation. However, this would increase the computational power and processing time.

Another limitation is the lack of research on continuous sign language recognition for ArSL. Of course, this can be attributed to our previous points regarding the limitations of data collection. Another thing to consider is the increased time, computation power, and memory required to train a model on more complex data, which increases cost in turn as noted by (Latif et al., 2020).

As we mentioned in the beginning of this research paper, different countries established their own sign language based on their own dialects. In daily life, signers would their dialectal sign language and not the official ArSL, similar to how dialects are used in place of the official Arabic Language. While research has expanded to included Isolated Word recognition, it is still limited in not using dialectical datasets.

2.3.3 Proposed future work

One possible area of research would include a method of collecting data. For example, a deep learning model can be utilized within an application that allows people who use sign language to contribute on a larger scale. For example, looking at Google Translate Tool which allows users to add new translations. A similar approach can be taken with sign language where users can add their own signs with the translations and then be reviewed by other verified contributors. Once approved, the new sign would be added to the database from which the model can learn.

Our review identified a research gap in continuous ArSL recognition using deep learning. Again, researchers would need a method that allows to collect dynamic data of full sentences that can be easy to collect and process by deep learning models. Promising results were achieved with the use of skeletal data collected through the Kinect device (Bencherif et al.,

2021). This is an interesting approach as the skeletal data won't require as much pre-processing or computational power.

To that end, researchers will need to consider increasing their scope of sign language recognition to include facial features and body language. As signers will make use of them as well for communication.

Another area of research would be the dialects. More research would need to be done on dialectical sign language if we would like to apply these systems in a real-world setting and truly benefit deaf people around the Arab world.

To better compare the performance of deep learning models, a proposed idea is establishing community challenges that use strict guidelines and the same data to compare the models and their performance.

3 Methodology

In this section we will be discussing our methodology to address some of the problems identified in the previous section. Namely we offer a solution to the following research gaps:

- Data Collection: We will be building a system that utilizes MediaPipe framework to collect landmarks directly from the webcam without the use of extra devices.
- Data Type: Our system will collect dynamic data through an action detection-based system that uses a sequence of input data rather than a single frame. We will also include data from the hands, face, and body.
- Dialects: We will be collecting and testing our system on signs from the Emirati Sign Language, which has never been done previously to our best knowledge.

3.1 Data

As discussed previously, one of the main issues faced by researchers is the data itself. For one, there needs to be a huge amount of data to train a good model. In the case of vision-based approach, the images/videos are affected by the environment, and the researchers will need to utilize segmentation methods, adding to the time and cost. In the case of sensor-based approach, the data is not affected by the environment, but the cost of the devices and time involved in collecting the data is inconvenient.

In our research, we found another approach that allows us to utilize the ease of data collection in vision-based methods along with the accuracy and impartiality of sensor-based methods. MediaPipe (Google, 2019) is an open-source framework developed by Google and offering cross platform machine learning solutions that can be customized. These solutions are basically a variety of pre-trained deep learning models for live and streaming media that can be utilized for different use cases as seen in figure 19.

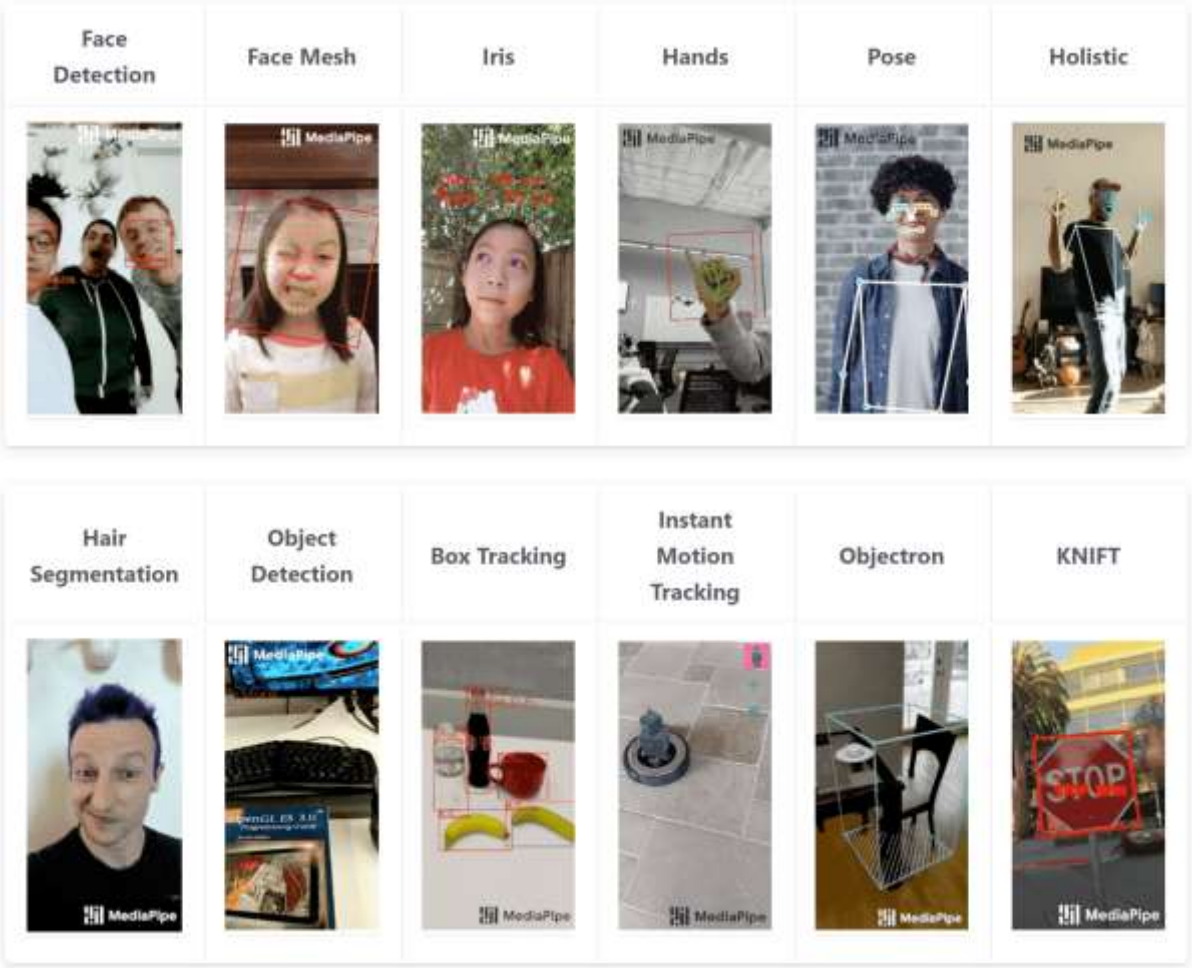


Figure 19 MediaPipe Solutions (Google, 2019)

For our case, we will be using the MediaPipe Holistic model, which is made of three different models to accurately detect landmark key points on face, hands, and pose components. Each model is optimized for its own domain. Because each model has different requirements, MediaPipe Holistic is designed as a multi-stage pipeline. We detail the stages below:

1. As we can see in figure 20, the human pose is the first to be estimated.
2. The pose is first detected to infer pose landmarks.
3. These inferred landmarks will be used to derive three regions of interest for the two hands and the face.
4. It then applies a re-crop model to improve these regions.

5. The full resolution input frame will then be cropped to these ROIs
6. Face and hand models estimate the corresponding landmarks.
7. All landmarks are merged to produce the 540+ landmarks.

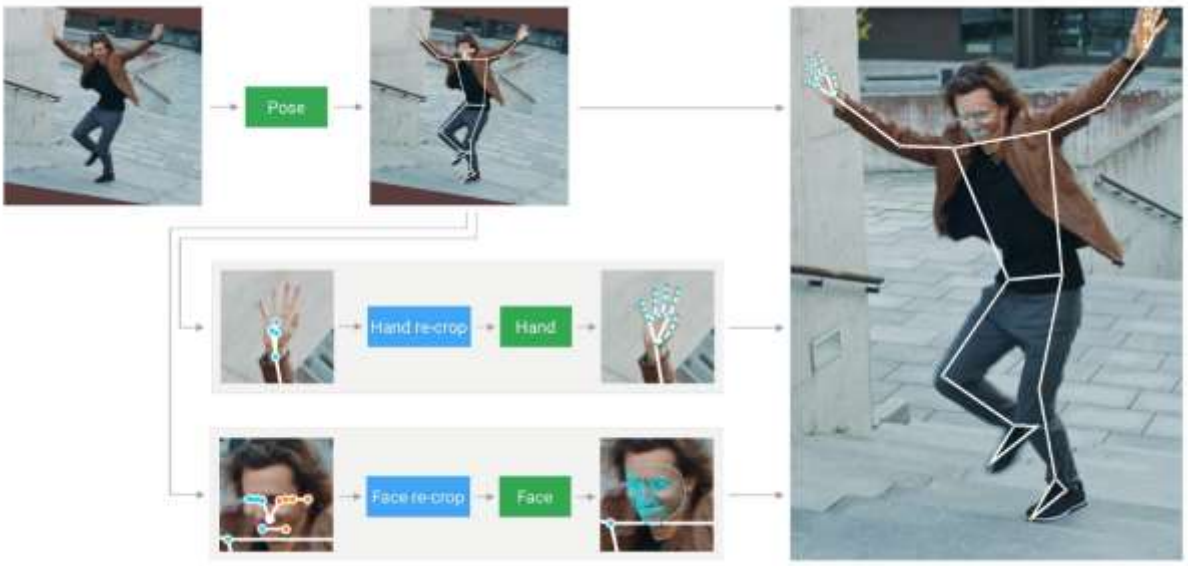


Figure 20 MediaPipe Holistic Pipeline

The pipeline utilizes a tracking approach for face and hands that uses the previous frame as a guide to estimate the object region of the current frame. It also uses pose prediction on every frame to help the model react faster to fast movement that might cause the tracker to lose its target. It also allows the model to maintain consistency across the models, preventing mixing with body parts from someone else or between the right and left hands.

3.1.1 Data Collection

In figure 21, we display the landmarks that can be detected by MediaPipe Holistic, which will be rendered directly on the user as shown in figure 20. As such, we will simply use our computer camera to collect the data. MediaPipe will render landmarks for the face, hands, and pose directly on the frame. In total, this method will allow us to create and collect 1662 landmarks

for each frame. We will then use OpenCV library (Bradski and Kaehler, 2000) to capture the continuous frames in a video sequence.

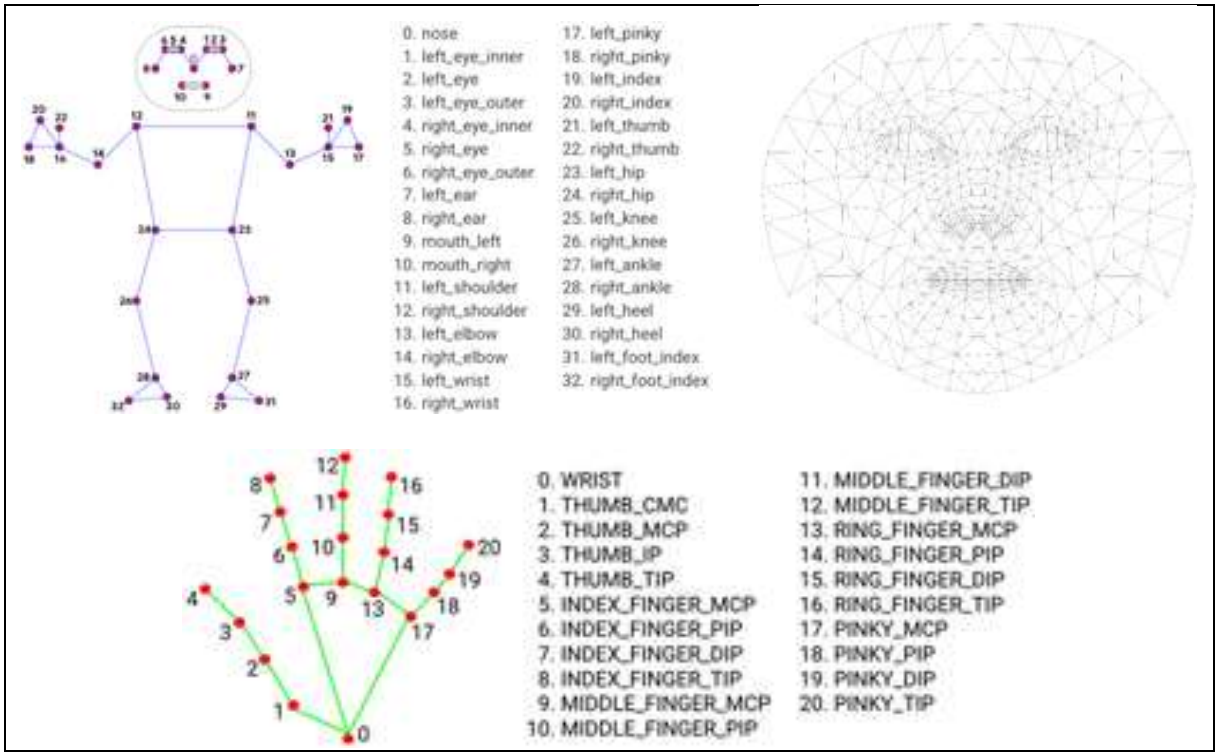


Figure 21 MediaPipe Landmarks for Pose, Face, and Hand (Google, 2019)

The signer will be performing the signs for the seven emirates (MOCDUAE, 2020). Each sign will be performed 30 times as a video sequence. Each video sequence will contain 30 frames of data and each of these frames will contain the 1662 landmarks mentioned above. These landmarks will be used to extract the key points that the model will be trained on. The data will be split 80-20 for training and testing: 24 video sequences for training and 6 for testing.

Another signer will perform the signs a total of six times for each sign, with each sign still having 30 frames per video. This dataset will be held-out and not used to train the model at all. We will use it for the purpose of signer independent recognition accuracy. We choose six times as this will be the same number as the testing dataset.

3.1.2 Single vs Double Hands

To clarify regarding the hand landmarks, our total of 1662 landmarks include the landmarks for two hands. The signs for the Seven Emirates only require one hand each. However, we chose to utilize the full capability of MediaPipe Holistic and include both hands in our system as we are aiming to provide a system that can be applied in a real-world scenario.

3.1.3 Dataset

Table 6 shows the exact details of our dataset in accordance the categories and types discussed in Section 2 for consistency.

Sign Language	Emirati Sign Language
Acquisition Method	Vision-Based Recognition
Classification Category	Isolated Words
Sing Language Symbol	Single hand
Dynamic/Static	Dynamic
Contents	Signs of the 7 Emirates
Datasets	*Signer-Dependent* Signer performs the 7 signs 30 times as video input (210 videos), with each video containing 30 frames, each frame containing 1662 landmarks Training: 80% of dataset – total of 168 videos Testing: 20% of dataset – total of 42 videos Validation: We will utilize a validation split for training
	Signer-Independent Another signer performs the 7 signs 6 times Total of 42 videos (same as testing dataset)
Features	1662 landmarks of the face, hands, and pose, keypoints of the landmarks will be used as input

Table 6 Dataset Details

Our dataset is comparable in size to existing research, for example, in (ElBadawy et al., 2015), two signers perform 20 dynamic signs, including facial features and body movement with the

use of two cameras (face and body) and a leap motion sensor (for hands). In (ALI, et al., 2014), one signer performs 5 signs with 200 samples each, noting that their approach is a traditional vision-based approach with static images, creating the need for more samples than our approach. (Mohandes, Aliyu and Deriche, 2014) collects 12 hand features from 10 samples for each of the alphabet, with each sample containing 10 frames. While our data collects 1662 landmarks from 30 samples (videos), each sample containing 30 frames.

3.2 Pre-processing

Keypoint Extraction. We collect all landmarks and then extract keypoint positions from each land mark into an array. The keypoints are collected as sequences and will be our features, Our input data for the model is a series of 30 arrays, each of which contains 1662 values (30, 1662). Each one of these 30 arrays represents the landmark values (1662) from a single frame.

Preparing data for training. As this is a classification task, we create one label map for our seven classes and another map for our sequence data. We will then be using one-hot encoding on our class array using *to_categorical* function in order to provide clear input for the network for our Y data.

We will also combine all of our data sequence arrays (the signs) into one array that will be used as our X data. Therefore, our array will have 210 videos (7 signs multiplied by 30 videos each), with 30 frames each, with 1662 key points in each frame.

3.3 Classification

We will be utilizing an LSTM model for the classification task. This is because LSTMs require less data and are generally faster to train. It has fast detection as well and it achieved good results in ArSL recognition based on our review in Section 2. The model will be predicting the action or the sign from a number of frames and not just a single one as the data is dynamic,

which an LSTM is well suited for as it is able to process sequence data with its feedback connections.

The exact architecture is not set as we will be experimenting with different hypermeters and tuning the model but overall, we will have the following layers:

- LSTM layer to process sequences of data.
- Dropout to help the network generalize better by dropping a number of neurons and avoid overfitting.
- Dense Layer for our connections
- Flatten with Softmax Activation function so that the output will have the properties of a probability distribution of 0 to 1.

3.4 System Architecture

The whole program will be built on Jupyter Notebook (Kluyver et al., 2016) using python. We will be leveraging Tensorflow (Abadi et al., 2016) and Keras (Chollet et al., 2015). Libraries include OpenCV (Bradski and Kaehler, 2000) and MediaPipe (Google, 2019). We utilize a GitHub repository by (Renotte, 2021)[1] and modify it to fit our needs[2]. Figure 22 was created though (Lucidchart, n.d.) and displays our system process.

1 https://github.com/nicknochnack/ActionDetectionforSignLanguage

2 https://github.com/HamdaAlmahri/ArabicSignLanguageRecognition/blob/main/Arabic%20Sign%20Language%20Recognition.ipynb

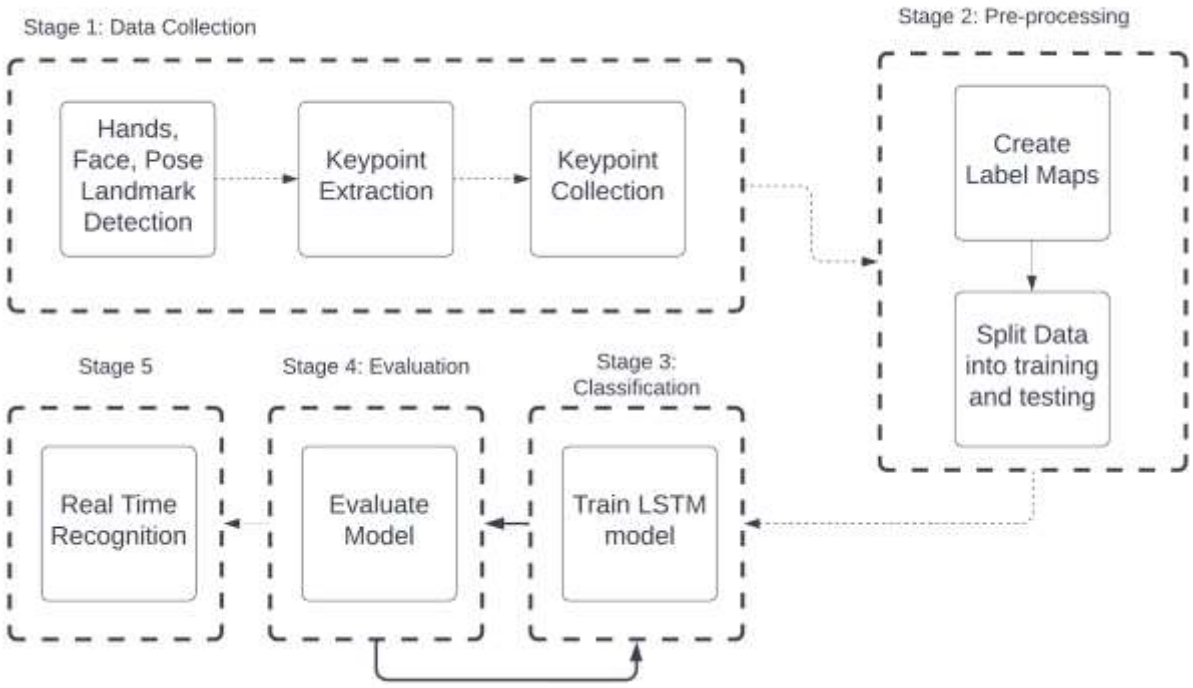


Figure 22 System Architecture

First, we install and import the required libraries and dependencies.

Stage 1: Data Collection

1. Use MediaPipe to detect and draw Face, Hand and Pose Landmarks
2. Extract keypoints of each landmark as preparation to collect the data
3. Prepare folders: Each Sign of the seven emirates will have a seperate folder. In each folder, we create 30 folders for each sequence we collect (30 video equal 30 folders). As each video will have 30 frames, we will have 30 keypoint sequences within these folders.

MP_Data > Class (7 folders) > Video Sequence (30 folders) > Keypoint Arrays (30 arrays)

As mentioned previously, each array will contain 1662 landmark information from a single frame

4. Collect Keypoint Sequences as arrays

Stage 2: Data Preparation and pre-processing

5. Create Label Maps for classes (our Y value) and keypoint sequences (our X value)
6. Split data into training and testing datasets

Stage 3: Classification

7. Build a deep learning LSTM model to train on the data

In here we will experiment with different values and hyper meters, more details mentioned in the next section.

Stage 4: Evaluation

8. Evaluate model using accuracy and confusion matrix on testing dataset. We will also test in Signer Independent Mode using a separate dataset collected from a different signer.

Stage 5: Real-Time Recognition

9. Once satisfised, we use both MediaPipe and our trained LSTM model to predict the signs in real-time using video sequences.

3.5 Evaluation Metrics

For our system, the action detection is a multi-class classification task; the action is predicted based on the highest class/sign probability. As such, we will use categorical cross-entropy accuracy and log loss to evaluate the model.

To optimize both the log loss and accuracy, we use Adam optimization algorithm for our gradient descent after each iteration.

As mentioned earlier, we will apply a validation split of 0.2 on the training data. This will help us with hyper tuning our model and prevent it from overfitting. We also utilize matplot (Hunter, 2007) for tracking.

- Signer Dependent mode: We will be reporting the accuracy of the model on the training, validation, and testing datasets throughout our paper.
- Signer Independent mode: once we train the model to a sufficient state, we will then apply it to the second dataset we collected, which has another signer performing the same signs. The purpose of this is to research the feasibility of using our approach in a real-world setting. Obviously, we are not aiming for 100% accuracy as the dataset collected is relatively small.
- Real-time Recognition: Finally, we will be using our model for real-time recognition. For this part, we evaluate the feasibility of Build a real-time recognition system using MediaPipe and deep learning models.

3.6 Model training strategy

We want to take a few measures to ensure that we are training correctly:

1. Our training and test datasets need to be representative of our original dataset. Meaning that both sets need to have the same number of samples for each class. For example, if we follow an 80-20 split, the training dataset would have 24 samples for each class, and the testing dataset would have 6 samples for each class.
2. We add a validation dataset during the training phase, this will help us compare the performance of the model loss and accuracy on the training dataset against a validation dataset, which in turn will help us identify if the model is overfitting or underfitting. It will also help us tune our model hypermeters in a systematic approach. The training-validation data split will also follow an 80-20 split.

3. Next, we will experiment with different hypermeters. Using a basic LSTM model, we experiment with the following:
    a. Epoch and batch size: We will use different epoch and batch size combinations and choose the combination that produces that highest accuracy.
    b. Layers number and activation
    c. Drop out rate

We must keep in mind that due to the stochastic nature of neural network models; results may differ even when training the same model on the same dataset. Our approach can be further improved by repeating each hypermeter combination a number of times and averaging the accuracy. However, this would require more resources including time and computational power. Our main aim is to observe the effect of tuning hypermeters on model performance.

4 Results

In this section, we will be implementing the methodology we discussed in section 3. We will follow the model training strategy we established in 3.6 and note the results.

4.1 Original Model

To start with, we first use the same model and hypermeters used in the original code. In that code, there are three classes and the model is trained for 2000 epochs. This will help us get an estimation of model performance on our specific dataset.

Important to note is that the original split is 95-5 in this model, where the testing dataset amount to 5% only.

```
model = Sequential()
model.add(LSTM(64, return_sequences=True, activation='relu',
input_shape=(30,1662)))
model.add(LSTM(128, return_sequences=True, activation='relu'))
model.add(LSTM(64, return_sequences=False, activation='relu'))
model.add(Dense(64, activation='relu'))
model.add(Dense(32, activation='relu'))
model.add(Dense(actions.shape[0], activation='softmax'))
```

We plot the loss and accuracy and display them in table 7. We note that the original code does not utilize a validation split, which is why we only display the training results.

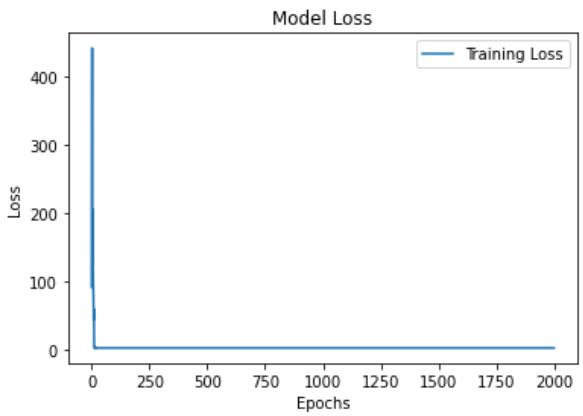
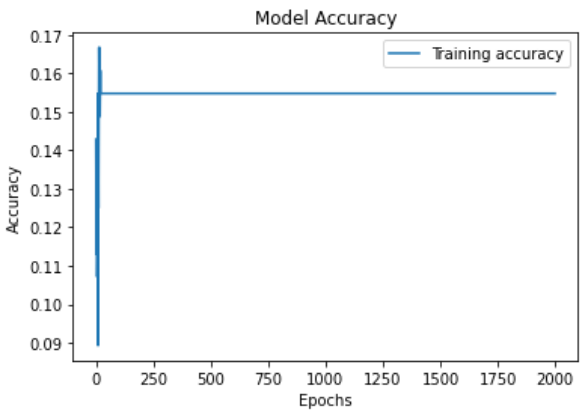
Loss	Accuracy
	

Table 7 Original Model Performance – 2000 epochs

Very abnormal, but the 2000 epochs prevent us from truly viewing the model changes in accuracy and loss, as it stabilizes and stops changing in under 250 epochs. Therefore, we will train to 100 epochs to get a better estimation of model behaviour and how to fix it. Results are displayed in table 8.

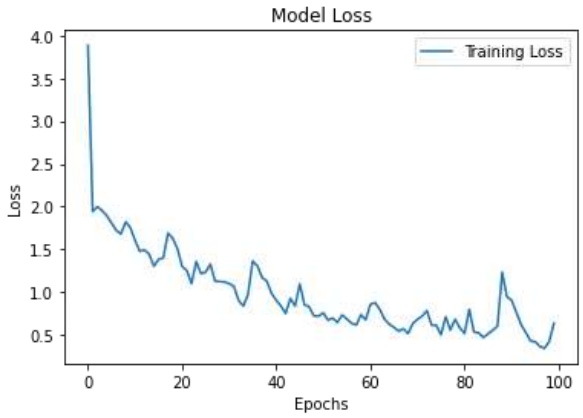
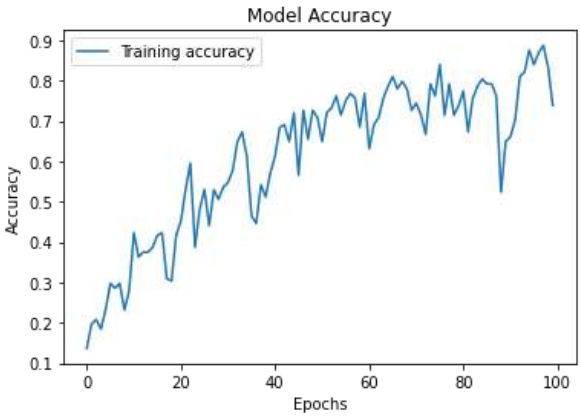
Loss	Accuracy
	

Table 8 Original Model Performance – 100 epochs

The model is behaving normally as the loss is decreasing and the accuracy is increasing. However, as we saw previously, performance won't improve as we train more. Table displays the training and testing accuracy for this iteration. Meaning the model is going in the right direction but not achieving the desired results.

Epochs	Training Accuracy	Test Accuracy
100	79%	69%

Table 9 Original Model Accuracy – 100 epochs

To gain better insight, we add a validation dataset. Our split so far is as follows:

Training: 80%, Validation: 15%, Test: 5%

We train the same model for 1000 epochs and note the results in table 10.

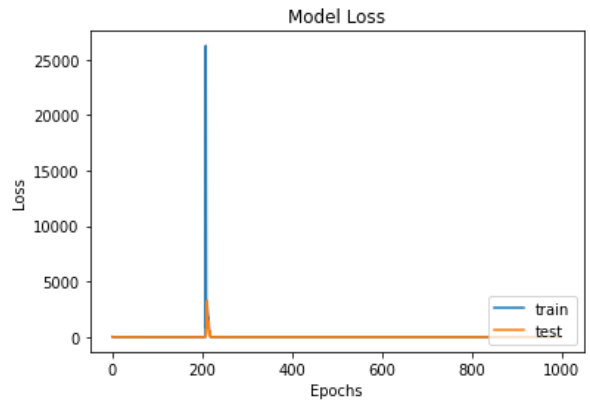
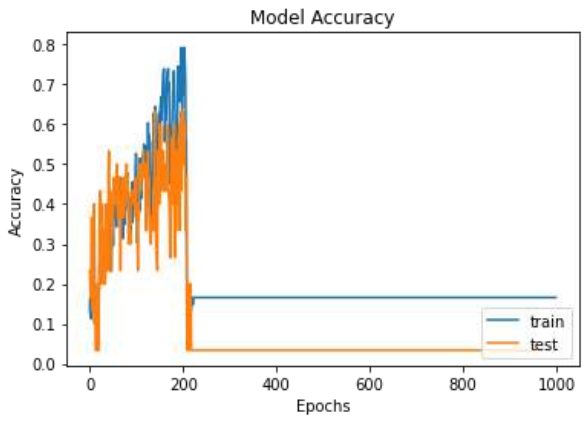
Loss	Accuracy
	

Table 10 Original Model Accuracy – with validation

We can see that the model completely fails. While it was doing well with the accuracy and increasing, we can see that the log loss behaves abnormally with a sudden increase in the middle.

From here, we abandon this model and rebuild our neural network from scratch to be tailored to our data.

4.2 Baseline Model

```
model = Sequential()
model.add(LSTM(100, input_shape=(30,1662)))
model.add(Dropout(0.5))
model.add(Dense(100, activation='relu'))
model.add(Dense(actions.shape[0], activation='softmax'))
```

We create a simple model with 1 LSTM layer, dropout layer, and 2 dense layers. To compare with the original model, we won't add a validation split. We note the model performance in table 11 and the accuracy in table 12, where we achieve an accuracy above 90%. Already we can see significant improvement in the training and test accuracy. We will now start hyper tuning our model.

- Batch Size and epochs
- Layer number

- Dropout rate
- Layer units

In all these experiments we utilize a 20% validation split unless otherwise noted.

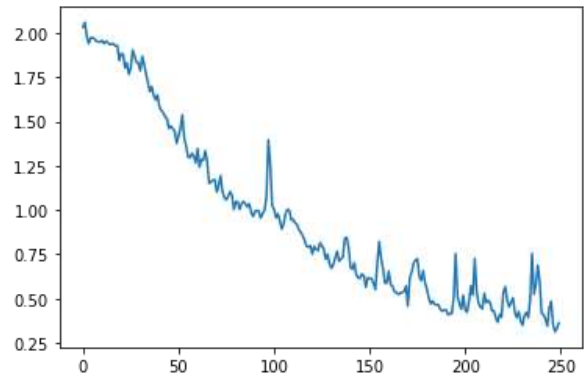
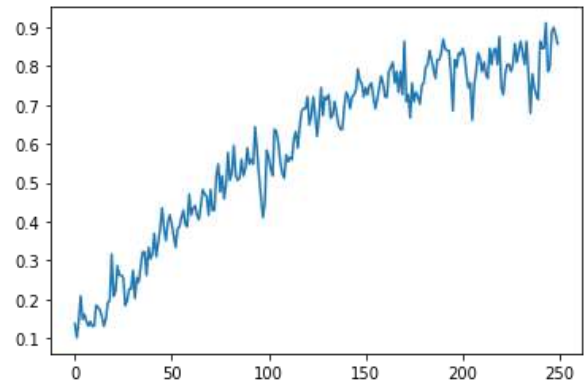
Loss	Accuracy
	

Table 11 Baseline Model Performance - 250 epochs

Epochs	Training Accuracy	Test Accuracy
250	94.05%	92.85%

Table 12 Baseline Model Accuracy – 250 epochs

4.2.1 Batch Size and Epochs

By default, the model will use a batch size of 4. We will be using different batch sizes of 2, 4, 8, and 16. We will also train each batch size 5 times using 100, 250, 500, 750, and 1000 epochs. Results are displayed in table 13 where we report the training loss and accuracy, and the validation loss and accuracy.

As models might fluctuate in results during training, we also plot the model performance to observe the model behaviour and how stable it is. As we are plotting for 20 different models, we will only be presenting the plots for batch sizes trained for 250 epochs in table 14, as we found it to be most stable. The rest of the plots can be viewed in Appendix B at the end.

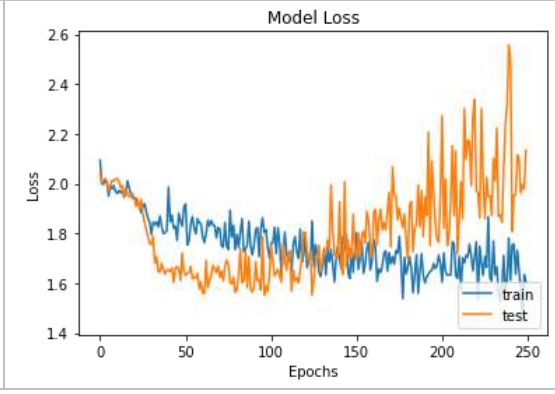
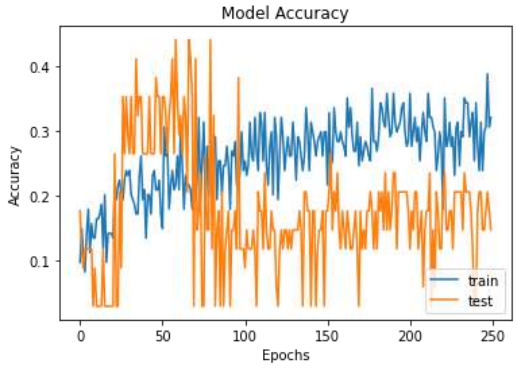
Batch Size	Epochs	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
2	100	1.9404	0.1665	1.9762	0.0294
2	250	1.5106	0.3802	2.1340	0.1471
2	500	1.5890	0.3406	2.4381	0.0882

2	750	1.9415	0.1621	2.0073	0.0294
2	1000	1.6453	0.2824	3.0407	0.1176
4	100	1.2807	0.3947	1.2264	0.3235
4	250	0.9220	0.6052	0.6088	0.8235
4	500	0.9158	0.5687	1.0538	0.5588
4	750	0.5419	0.7955	2.1078	0.5000
4	1000	0.6528	0.7790	1.7887	0.5294
8	100	1.0183	0.5185	0.8523	0.5588
8	250	0.4548	0.8214	0.3480	0.9118
8	500	0.1627	0.9474	0.1075	1.0000
8	750	0.2335	0.8948	0.0821	1.0000
8	1000	0.2330	0.8912	0.0123	1.0000
16	100	0.9693	0.5693	0.7458	0.8235
16	250	0.3467	0.8767	0.3306	0.8529
16	500	0.1416	0.9346	0.1213	0.9706
16	750	0.6539	0.8643	0.1106	1.0000
16	1000	0.0511	0.9745	0.0025	1.0000

Table 13 Model Performance by Batch Size and Epcohs

The best performance was observed at 250 epochs for batch size 2 and 4, 500 epochs for batch size 8, 1000 epochs for batch 16. However, by observing our graphs and looking at our data, we realize that the model is actually overfitting. It is learning the training data too well and is not able to generalize. Thus, we will be setting our training epochs to 250.

We decide on the batch size through the graphs in table 14, we can see that the model is most stable at batch size 8. While it fluctuates greatly in batch size 2 for example.

Training Iteration	**Loss**	**Accuracy**
Batch size = 2 Epochs = 250		

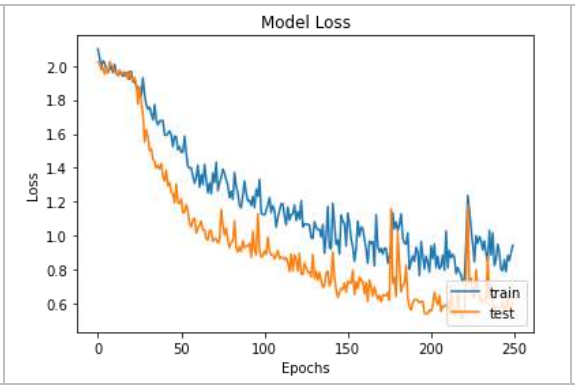
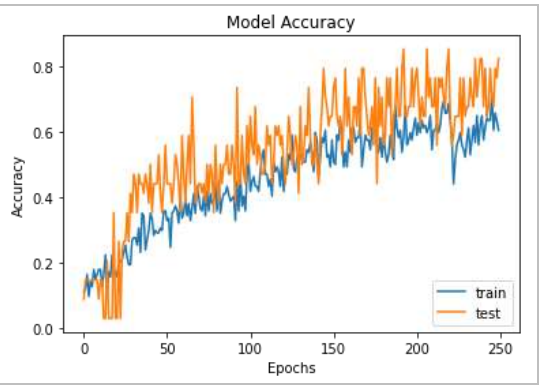
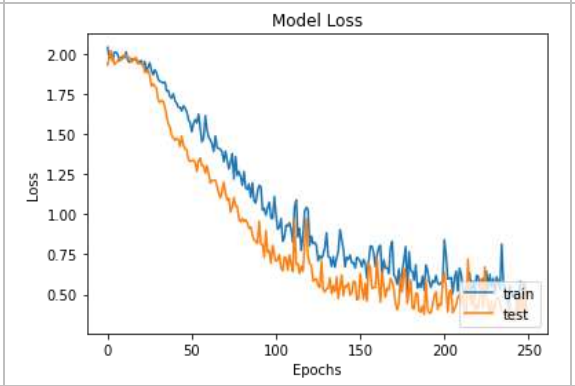
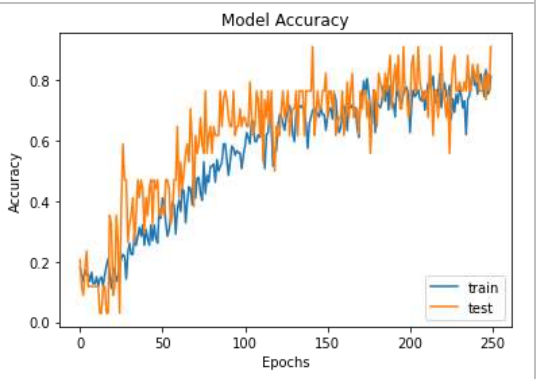
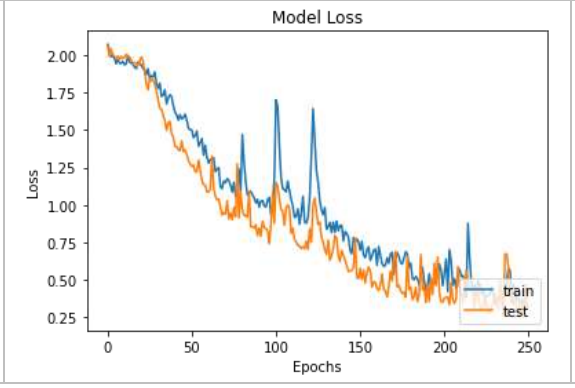
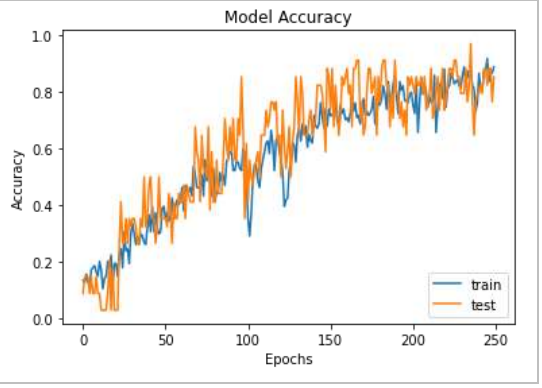
Batch size = 4 Epochs = 250		
Batch size = 8 Epochs = 250		
Batch size = 16 Epochs = 250		

Table 14 Model Performance for 250 epochs – Graph Comparison

To conclude this part of the testing, we choose batch size 8 and epoch 250 moving forward.

4.2.2 Different LSTM Layers

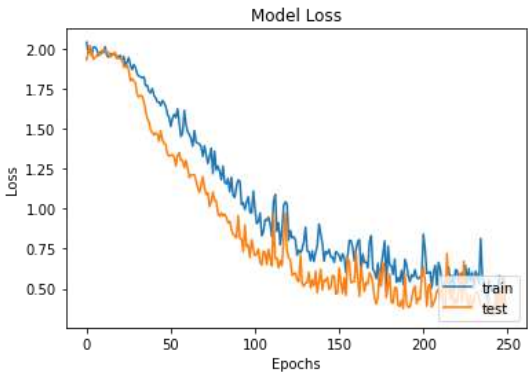
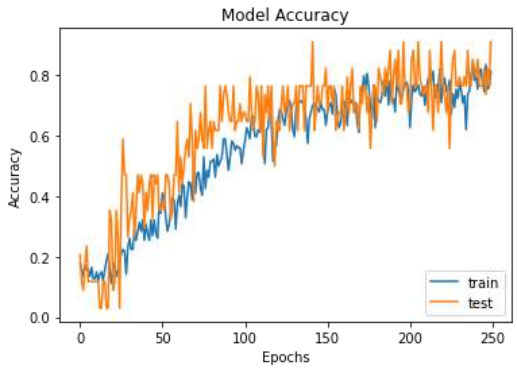
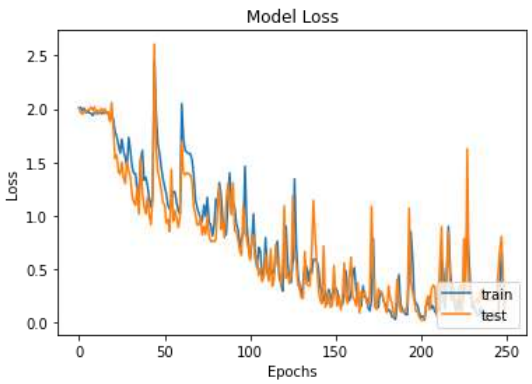
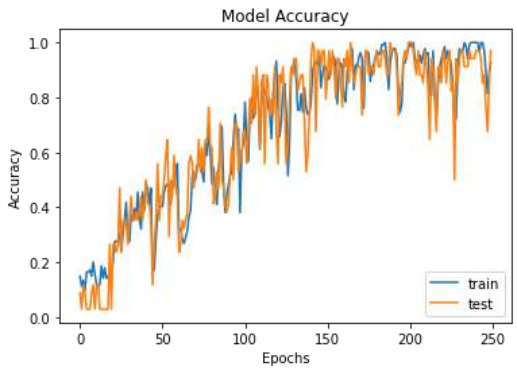
The number of layers helps the model better extract and learn features. As our baseline model only has one LSTM model, we will be experimenting with adding a second layer with the same number of units (100) as the first layers. In addition, the original model has a ReLU activation added to the LSTM layers, which helps manage the computational power needed to train the

model. Thus, we experiment with adding the activation to the first layer only and then to both layers. Results are reported in table 15.

LSTM Layer Number	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
1	0.4548	0.8214	0.3480	0.9118
2	0.1680	0.9333	0.1266	0.9706
2 with relu activation in 1st layer	1.9589	0.1726	1.9812	0.0294
2 with relu activation in both layers	25.6441	0.1722	27.5234	0.1176

Table 15 Model Performance by LSTM layers

We are able to see significant increase in performance by adding a second layer. As training accuracy increased from 82% to 93%, and the validation accuracy increased from 91% to 97%. Meanwhile, the model performs poorly once we add ReLU activation. The same can be seen in the graphs shown in table, where the model fails with ReLU activation.

Training Iteration	Loss	Accuracy
Batch size = 8 Epochs = 250 1 LSTM layer		
Batch size = 8 Epochs = 250 2 LSTM layer		

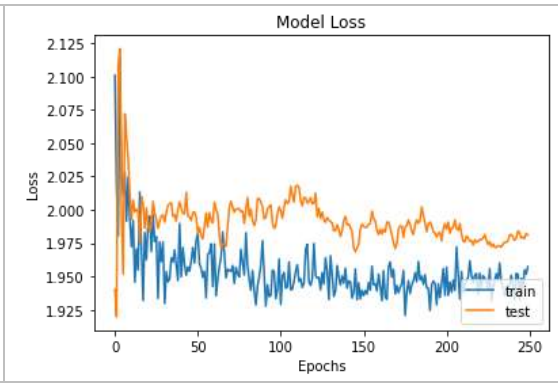
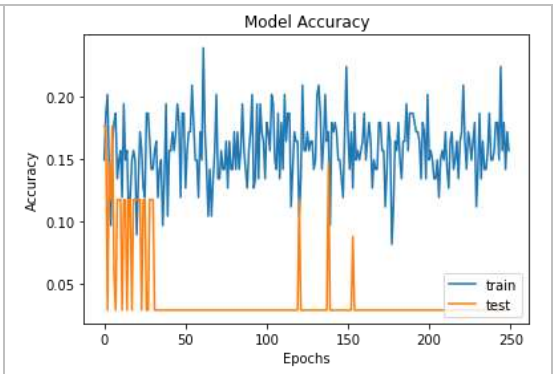
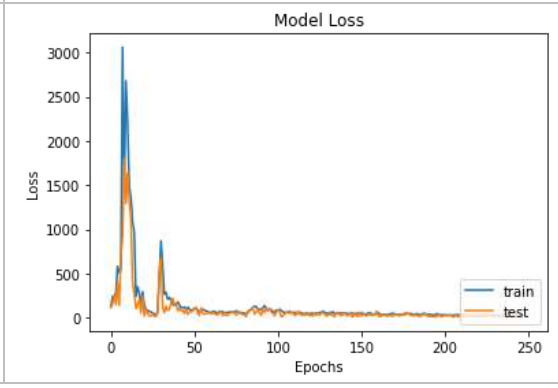
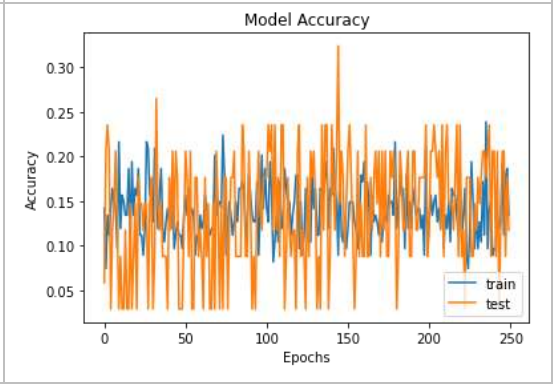
Batch size = 8 Epochs = 250 2 LSTM layers with relu activation in 1st layer		
Batch size = 8 Epochs = 250 2 LSTM layers with relu activation in both layers		

Table 16 Model Performance by LSTM layer – Graph Comparison

To conclude this part of the testing, we add a second layer to our model.

```
model = Sequential()
model.add(LSTM(100, return_sequences=True, input_shape=(30,1662)))
model.add(LSTM(100, return_sequences=False))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dense(actions.shape[0], activation='softmax'))
```

4.2.3 Different dropout rates

Drop out helps with regularization of the model and prevents overfitting. We set our dropout rate to 0.5. We will be experimenting with 0.2, 0.5, 0.6, and 0.8 dropout rates. Again, we are training on batch size 8 for 200 epochs using 2 LSTM layers. Results are shown in table 17.

Dropout Rate	Training Loss	Training Accuracy	Validation Loss	Validation Accuracy
0.2	0.2760	0.8443	0.5020	0.8529
0.5	0.1680	0.9333	0.1266	0.9706
0.6	0.5191	0.7637	0.4185	0.7941
0.8	0.3288	0.8688	0.3236	0.8529

Table 17 Model Performance by Dropout Rate

We observe that our model achieves the best result on the default 0.5 dropout rate we choose.

Reviewing the graphs in table 18 shows the model to be most stable at this rate as well.

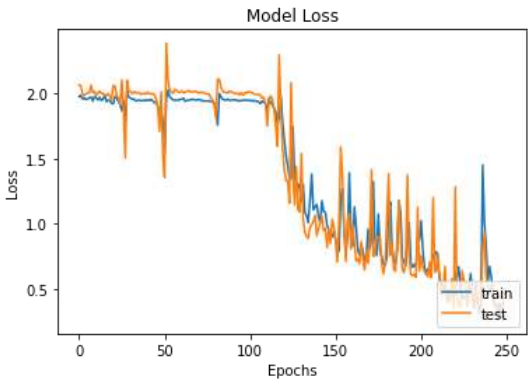
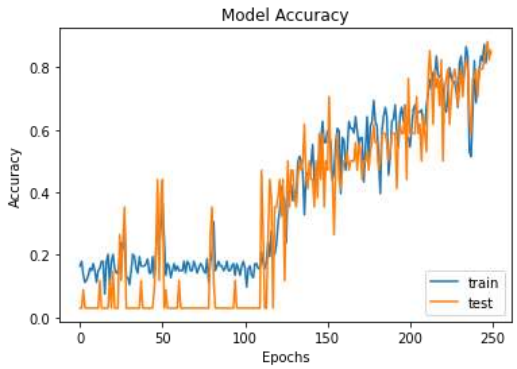
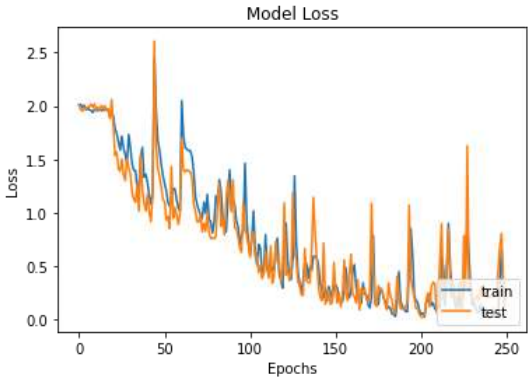
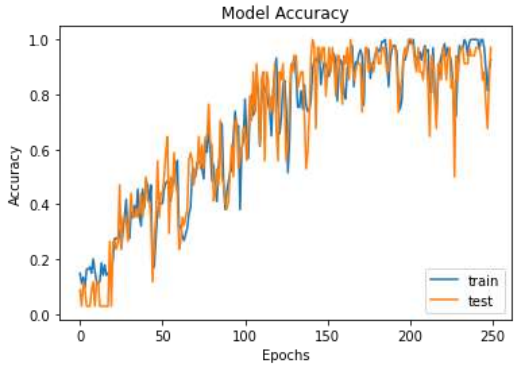
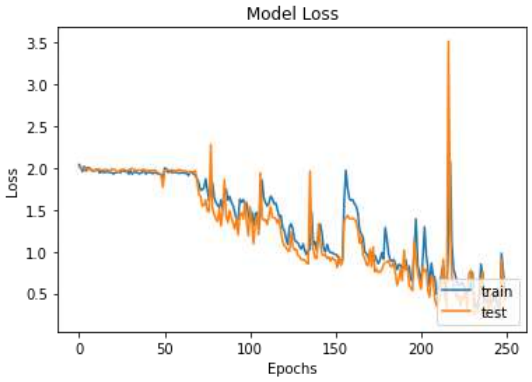
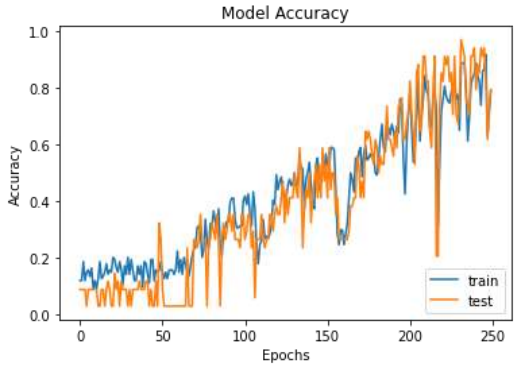
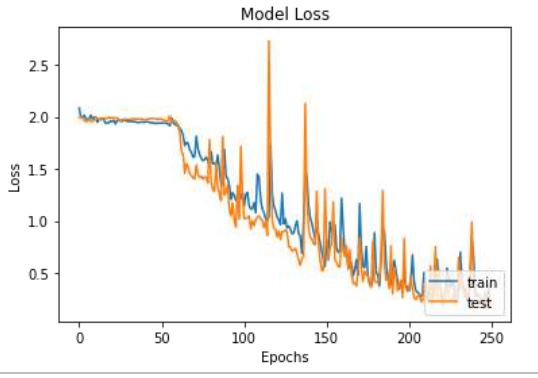
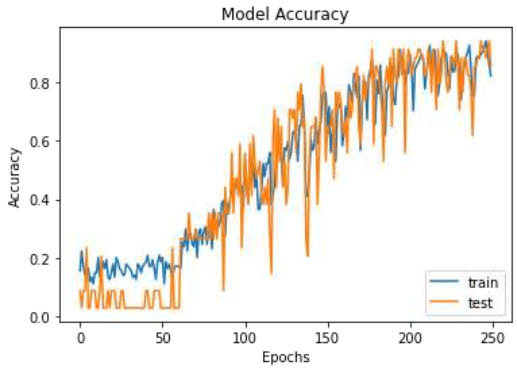
Training Iteration	**Loss**	**Accuracy**
Batch size = 8 Epochs = 250 2 LSTM layer Dropout: 0.5		
Batch size = 8 Epochs = 250 2 LSTM layer Dropout: 0.5		
Batch size = 8 Epochs = 250 2 LSTM layer Dropout: 0.6		
Batch size = 8 Epochs = 250 2 LSTM layer Dropout: 0.8		

Table 18 Model Performance by Dropout Rate – Graphs

To conclude the testing for our baseline model, our established hypermeters are batch size 8, 250 epochs, 0.5 dropout rate, and 2 LSTM layers.

4.3 Signer Independent Testing

We now finalize our model using the hypermeters we tuned. We now train the model using all our training data, and note the test accuracy. We then test it on our independent dataset. As mentioned previously, our split is 80-20 for training and testing. The independent dataset contains the same number of signs as the testing dataset.

Results are reported in table 19. We see that we achieved an excellent result for our signer dependent mode, accomplishing 100% accuracy. For our signer independent mode, we only achieve an accuracy of 50%.

Model	Signer Dependent Mode		Signer Independent Mode
	Training Accuracy	Test Accuracy	
Baseline Model: LSTM layer (100 units) LSTM layer (100 units) Epochs: 250 Training data: full data used for training	100%	100%	50%

Table 19 Model Accuracy for Signer Dependent and Independent Mode

4.3.1 Improving performance (Signer Independent)

We would like to further investigate on this and if we can improve performance further for the signer dependent mode. We decide to experiment with the units of the LSTM layers as they deal with model learning.

In our baseline model, we have 100 units for each layer. We will be instead using the same number of units used in the first two layers of the original model we first used (64 and 128 units).

```
model = Sequential()
model.add(LSTM(64, return_sequences=True, input_shape=(30,1662)))
model.add(LSTM(128, return_sequences=False))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dense(actions.shape[0], activation='softmax'))
```

Again, we will be using our established hypermeters of 250 epochs, batch size 8, dropout 0.5 and 2 LSTM layers. The only difference is the units' number. Results are reported in table 20.

Model	Signer Dependent Mode		Signer Independent Mode
	Training Accuracy	Test Accuracy	
LSTM layer (64 units) LSTM layer (128 units) Epochs: 250 Training data: full data used for training	98.21%	92.85%	35.71%

Table 20 Model Accuracy with updated units – 250 epochs

By looking only at the final results, we can see that the model performs even worse. However, while training, we notice that the model stabilizes at 200 epochs, after which accuracy lowers. Thus, we decide to try again and train the model for 200 epochs and note the results in table 21.

Model	Signer Dependent Mode		Signer Independent Mode
	Training Accuracy	Test Accuracy	
LSTM layer (64 units) LSTM layer (128 units) Epochs: 200 Training data: full data used for training	100%	97.61%	50%

Table 21 Model Accuracy with updated units – 200 epochs

We notice a significant increase for the signer independent mode at 50% compared to 35.71% just by lowering the number of epochs, as this is where the model stabilized. These results are actually very similar to our baseline model with only the test accuracy being different at 97.61% rather than 100% as in the baseline model. Though this might indicate that the model has better generalization.

To confirm this, we add a validation split of 20% to the training data for more generalization and note the results in table 22.

Model	Signer Dependent Mode		Signer Independent Mode
	Training Accuracy	Test Accuracy	
LSTM layer (64 units) LSTM layer (128 units) Epochs: 200 Training data: 80% data used for training	92.85%	88.09%	54.76%

Table 22 Model Accuracy with updated units and validation split – 200 epochs

Naturally, signer dependent mode accuracy is lower as we used less data to train the model. However, we notice an increase in the signer independent mode at 54.76% compared to 50% by lowering the training data amount.

4.3.2 Confusion matrix

To better understand our results. We will be looking at the confusion matrix for our signer dependent mode and signer independent mode. To simplify, we will only look the confusion matrix of the highest achieved results in each mode.

Table 23 shows the confusion matrix for the classes in the signer dependent mode. As we achieved 100%, we can see the same results in all the classes. The confusion matrix data is restricted to the True Positive and True Negative, meaning the model is predicting the result correctly for all sequences.

Table 24 shows the confusion matrix for the signer independent mode. Despite the lower score if 54.76%, we can see that the actual accuracy for each class is between 70-100%. Of course, the accuracy in the confusion matrix is calculated based on the True Positives and True Negatives of the class. Reviewing this chart is still useful in observing which classes performed better than the others. We can see that the highest accuracy was achieved in Ras Al Khaimah. While the lowest was in Ajman.

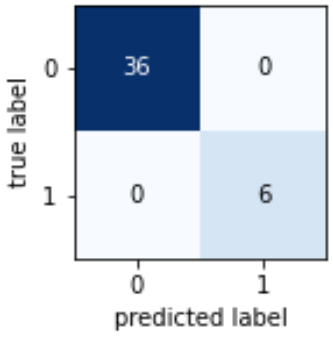
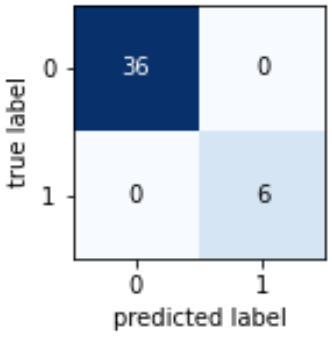
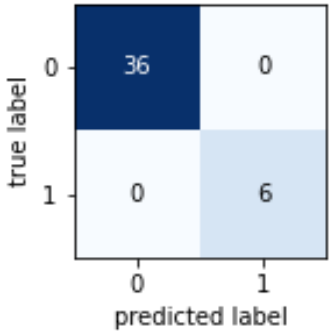
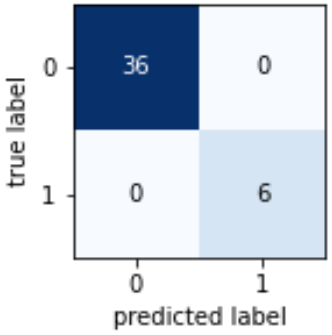
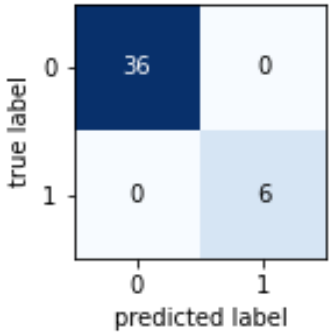
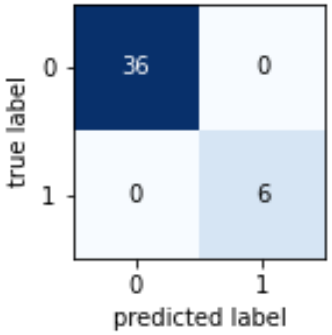
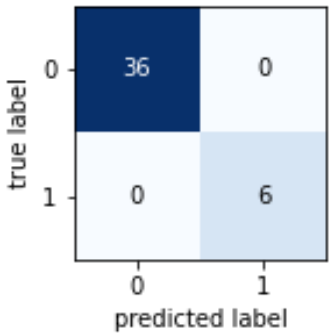
Sign	Confusion Matrix	Sign	Confusion Matrix
Abu Dhabi		**Dubai**	
	Accuracy 100%		**Accuracy 100%**
Sharjah		**Ajman**	
	Accuracy 100%		**Accuracy 100%**
Umm Al Quwain		**Ras Al Khaimah**	
	Accuracy 100%		**Accuracy 100%**
Fujairah			
	Accuracy 100%		

Table 23 Confusion Matrix for Signer Dependent Mode

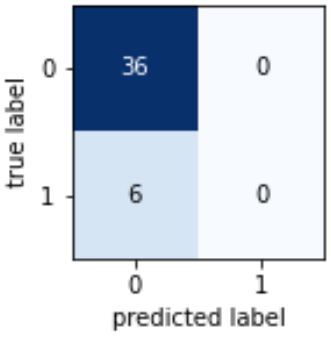
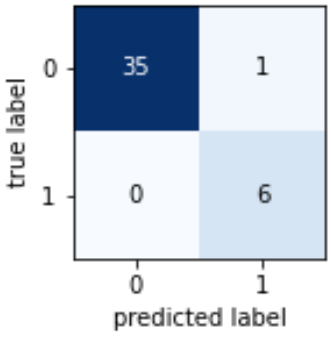
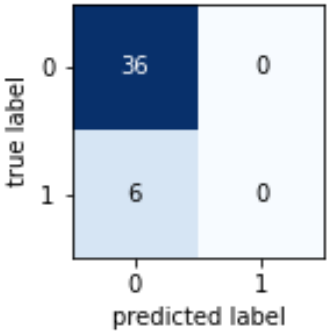
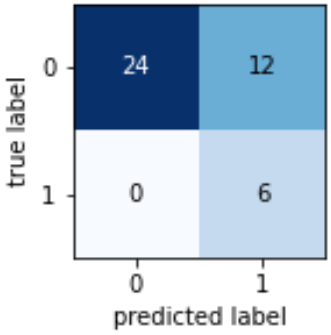
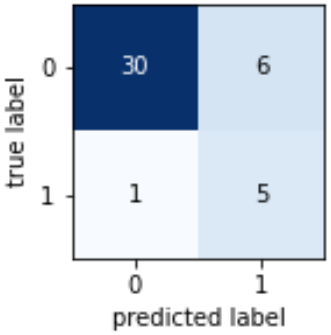
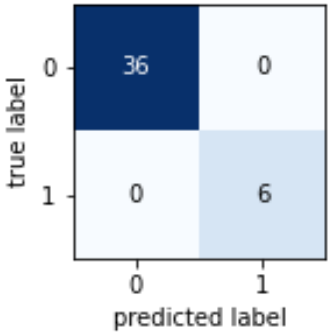
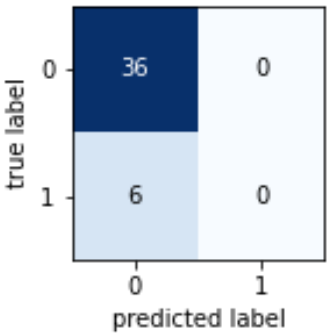
Sign	Confusion Matrix	Sign	Confusion Matrix
Abu Dhabi		**Dubai**	
	Accuracy 85.71%		**Accuracy 97.62%**
Sharjah		**Ajman**	
	Accuracy 85.71%		**Accuracy 71.43%**
Umm Al Quwain		**Ras Al Khaimah**	
	Accuracy 83.33%		**Accuracy 100%**
Fujairah			
	Accuracy 85.71%		

Table 24 Confusion Matrix for Signer Independent Mode

4.4 Final Results

We summarize our results in this table and indicate the highest the achieved performance.

Model	Signer Dependent Mode		Signer Independent Mode
	Training Accuracy	Test Accuracy	
Baseline Model: LSTM layer (100 units) LSTM layer (100 units) Epochs: 250 Training data: full data used for training	**100%**	**100%**	50%
LSTM layer (64 units) LSTM layer (128 units) Epochs: 200 Training data: full data used for training	100%	97.61%	50%
LSTM layer (64 units) LSTM layer (128 units) Epochs: 200 Training data: 80% data used for training	92.85%	88.09%	**54.76%**

Table 25 Final results for our models

4.5 Real Time Recognition

We then take our best performing model and use it along with MediaPipe to test in real time. We use a similar approach to how we collect the data in that we utilize MediaPipe Holistic to detect and render key points on the hands, face, and body in real-time through the computer camera. Instead of collecting the keypoints and testing the model on them, we incorporate the model to create an action detection system that predicts signs in real-time.

The seven signs are displayed on the screen and once the signer makes a sign, the system calculates the probability of which class the performed sign belongs to. This will reflect on the screen as a bar that increases and decreases in real-time for each sign based on probability. Once the system predicts the class, the class will show as text.

The system is predicting continuously, and text will show in sequence according to what is being predicted. Figure 23 shows an example of the display. This is a visual representation

where we see that the probability of the class being “Ajman” is highest as indicated by the dark blue bar. Thus, the word Ajman is printed in the light blue bar at the top of the screen.

Figure 23 Sample View for Real-Time Recognition

5 Discussion

To recap, our research has two aims:

1. Understand the current status of research in Arabic Sign Language Recognition using Deep Learning and find research gaps.
2. Build a deep learning recognition system that bridges the research gap.

We were able to answer our first question through our systematic review, for which we provide full analysis and discussion in section 2 of this dissertation along with our findings. We find that the main issue that set back research in deep learning models was the lack of a standard dataset that researchers can use. Available datasets are limited in that they don't have enough variety and only consist of the signs for the alphabet.

Deep learning has a lot to offer with regards to sign language recognition. For one, deep learning models can be used for both feature extraction and classification. However, to truly benefit from the deep learning architecture, we would need to train it on a huge amount of data. Of course, the higher the quality of the data, the better the model would perform.

Which brings us back to the issue with our Arabic Sign Language recognition. The research is mostly done on static images of the alphabets, which are not truly representative of the sign language in real-life. In reality, signs are dynamic and performed in a continuous manner. Deaf people would also use their country's sign language instead of the official Arabic Sign Language.

Which brings us to the second aim of our research. To bridge the research gap, we researched possible alternatives to the data collection methods discussed previously. We then found research on Google's MediaPipe, which is an open-source framework for live and streaming media that can be customized. One of its models, MediaPipe Holistic, is able to detect and

render landmarks on the hands, face, and body. It is similar to Leap Motion Controller but not limited to the hands only. Signers don't only use their hands to communicate, rather, they use facial features and body movement as well. Therefore, the benefit of using MediaPipe Holistic is that it does not need any extra devices like LMC and Kinect, and is more encapsulating in that it detects all three elements (hands, face, and pose) at the same time without resorting to capturing data through multiple devices. MediaPipe can also be implemented on different platforms.

We then build a system based on MediaPipe library as discussed in the Methodology section and work on optimizing the model for Emirati Sign Language. To the best of our knowledge, we are the first paper to apply a Sign Language Recognition system on the Emirati Sign Language. As we discussed in section 1 and 2, most research has been applied on Alphabets and Arabic Sign Language Isolated words while Signers around the Arab World use their own country's Sign Language.

From there, we work on optimizing our model through tuning our hypermeters. The main aim was to optimize the model and achieve a better accuracy and lower log loss. Due to the stochastic nature of deep learning models, results can vary even when training the same model on the same data. A model trained on a similar dataset may not perform well when applied on another dataset for the same task, which was the case for us.

To summarize the results of our hypermeter experiment, we note that deep learning models would need to be optimized for the specific dataset it was trained on. Deep learning models don't always generalize well unless we train and optimize them on a huge amount of data.

This was the case for us once we tested our model on a signer independent dataset. On our own dataset, we were able to achieve 100% accuracy on the testing dataset. But when tested on the

signer independent dataset, we only achieved 50%. To clarify, this actually is comparable to existing research. (Suliman et al., 2021) used Kinect device to collect their data and applied an LSTM model for classification. They achieved 95.9% accuracy (signer-dependent) and 43.62% (signer-independent). Meaning our LSTM model achieved a higher accuracy in both cases. We still did tests to see how we can improve the model and were able to increase the signer independent accuracy to 54.76% by training the model on less data and for less time, although the signer dependent accuracy became much lower at 88.09% for the testing dataset. This indicates that models with higher accuracy might not always generalize better for unseen data.

We also use confusion matrix to get a better idea on where the model is predicting wrong for the signer independent classification. We find that “Ras Al Khaimah” had the highest accuracy at 100% while “Ajman” had the lowest at 71.43%. This can be explained by the fact the sign for Ras Al Khaimah is done by raising your arm and placing your pointer finger at the top of your head, while all the other signs are done on or near the face, which is the case for Ajman and might explain the low accuracy.

That being said, another reason would be the limited amount of data, it is true that we collect 30 videos for each sign where each video has 30 frames with each frames having 1662 landmarks. However, this might not be enough to generalize to other datasets by other users. The input for our model is the key points of the landmarks, their position in other words. Therefore, difference in sign position and speed in the signer independent dataset would create different data from the signer dependent dataset the model was trained on. This is a limitation that can be resolved by training the model on a larger amount of data from many different signers and with different speeds and positions.

Our model certainly achieves results comparable, if not better, than published research. However, the model cannot be truly compared against these models as we are not training on the same dataset. Similarly, our model was trained on a small dataset and is not representative of how it would perform on a larger sample. The nature of deep learning makes it so that the model would need to be optimized according to your specific data. It is also why we cannot truly perform a statistical significance test, as we are not hypothesizing that our model performs better than other existing studies.

That being said, the aim for our methodology was to test the feasibility of using a completely different approach to what has already been tried in current research. We have already established that data acquisition and lack of a standard/baseline dataset was one of the major issues that set back research on Arabic Sign Language Recognition.

Kinetic and LMC may offer similar approach but we need to consider that this means in order to collect data, the signer must be in a place where the device is set up. In our approach, no device is needed other than a computer with a webcam. Provided the collection is deployed through a Web or Mobile application, we can potentially collect data from signers all over the world. This is far more convenient in terms of cost. As the world recovers from COVID-19, researchers must employ the use of innovative technologies to further the research.

Our model has high potential for data collection and recognition on a large scale based on our results. It is also very cost effective as the MediaPipe library is open-source and can be implemented on many platforms. No extra devices are required beyond the computer camera used for collecting the data. It can be run on CPU without using GPU.

Further to this, we were also able to achieve our goal of performing sign language recognition in real-time using our system. The system produces predictions in real-time in a continuous manner. The system has potential to be used for continuous sign language recognition as a real-world application of this system.

6 Conclusion

In this dissertation, we were able to review and identify gaps in current research. We find that data acquisition is the main issue in the task of Arabic Sign Language recognition using deep learning. We then aim to solve this issue and bridge the research gap by building a recognition system based on MediaPipe library for data collection.

To summarize, we contribute to research by creating a system that is able to collect dynamic signs though collecting the landmarks rendered on the face, hands, and body. This allows the collection of hand movement as well as facial features and body movement, which are essential in a real-world scenario for Sign Language Recognition. It eliminates issues found in other vision-based collection devices such as environment, background, illumination, distance, skin tones in the case of camera, and sensitivity to light, need of extra devices in the case of LMC/Kinetic.

Our model performs predictions in real time by following a prediction-based approach that allows it to continuously predict the signs being performed and displayed them on screen. We were able to achieve 100% accuracy in signer-dependent mode and 54.76% in signer independent mode, which is a result comparable to current research. Finally, we apply the system for Emirati Sign Language Recognition, a task that has not been applied before to the best of our knowledge.

6.1 Proposed Future Work

Building from this approach, a proposed future work would be to apply this method on a large-scale system. As MediaPipe works on many platforms, the system could be deployed to a web or mobile application that serves to collect data from many users across the world. To ensure the data is labelled correctly, experts could review the signs collected by the system to ensure high quality of data. Another option is to allow collaborative reviewing where users rate and categorize the signs performed by other users. From there, a deep learning model would continuously build on the new data and update the network and weights for higher accuracy.

Looking at real-world examples, Google gives an option for volunteers to contribute to their "Google Translate" service, which is based on neural networks, to improve translation accuracy for many languages (Google, n.d.). A similar collaborative system can be utilized for Sign Language Recognition. Users would perform a sign and the system gives them a prediction. The user can then rate if the system predicted the sign correctly, and the neural network would learn from the user input and update its weights and improve the accuracy.

The task of building a useful and applicable Arabic Language Recognition System may seem daunting. However, collaborating and working together as a community is what would allow us to breakthrough our individual limitation.

References

"The Arabic Dictionary of Gestures for the Deaf," 2007. [Online]. Available at: http://www.menasy.com/arab%20Dictionary%20for%20the%20deaf%202.pdf

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. and Kudlur, M., 2016. {TensorFlow}: A System for {Large-Scale} Machine Learning. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)* (pp. 265-283).

Abdul, W., Alsulaiman, M., Amin, S.U., Faisal, M., Muhammad, G., Albogamy, F.R., Bencherif, M.A. and Ghaleb, H., 2021. Intelligent real-time Arabic sign language classification using attention-based inception and BiLSTM. *Computers & Electrical Engineering*, *95*, p.107395.

Adeyanju, I.A., Bello, O.O. and Adegboye, M.A., 2021. Machine learning methods for sign language recognition: A critical review and analysis. *Intelligent Systems with Applications*, *12*, p.200056.

Ahmed, A.M., Alez, R.A., Tharwat, G., Ghribi, W., Badawy, A.S., Changalasetty, S.B., Belgacem, B. and Al Moustafa, A.M., 2018, December. Gestures Arabic Sign Language Conversion to Arabic Alphabets. In *2018 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)* (pp. 1-6). IEEE.

Al Mashagba, F.F., Al Mashagba, E.F. and Nassar, M.O., 2014. Automatic Isolated-Word Arabic Sign Language Recognition System Based on Time Delay Neural Networks. *Research Journal of Applied Sciences, Engineering and Technology*, *7*(11), pp.2261-2265.

Alani, A.A. and Cosma, G., 2021. ArSL-CNN: a convolutional neural network for Arabic sign language gesture recognition. *Indonesian Journal of Electrical Engineering and Computer Science*, *22*(2), p.1096.

Alawwad, R.A., Bchir, O. and Ismail, M.M.B., 2021. Arabic Sign Language Recognition using Faster R-CNN. *International Journal of Advanced Computer Science and Applications, 12*(3).

ALI, A.H.A., Kamal, A.F., Mostafa, M.S. and Hassan, D.A., 2014. Arabic Sign Language Recognition Using Neural network. *Journal of the ACS, 8*.

Al-Jarrah, O. and Halawani, A., 2001. Recognition of gestures in Arabic sign language using neuro-fuzzy systems. *Artificial Intelligence, 133*(1-2), pp.117-138.

Alnahhas, A., Alkhatib, B., Al-Boukaee, N., Alhakim, N., Alzabibi, O. and Ajalyakeen, N., 2020. Enhancing the recognition of Arabic sign language by using deep learning and leap motion controller. *International Journal of Scientific and Technology Research, 9*(4), pp.1865-1870.

Al-Shamayleh, A.S., Ahmad, R., Jomhari, N. and Abushariah, M.A., 2020. Automatic Arabic sign language recognition: A review, taxonomy, open challenges, research roadmap and future directions. *Malaysian Journal of Computer Science*, *33*(4), pp.306-343.

Alshomrani, S., Aljoudi, L. and Arif, M., 2021. Arabic and American Sign Languages Alphabet Recognition by Convolutional Neural Network. *Advances in Science and Technology. Research Journal*, *15*(4).

Aly, S. and Aly, W., 2020. DeepArSLR: A novel signer-independent deep learning framework for isolated arabic sign language gestures recognition. *IEEE Access, 8*, pp.83199-83212.

Bencherif, M.A., Algabri, M., Mekhtiche, M.A., Faisal, M., Alsulaiman, M., Mathkour, H., Al-Hammadi, M. and Ghaleb, H., 2021. Arabic sign language recognition system using 2D hands and body skeleton data. *IEEE Access, 9*, pp.59612-59627.

Boukdir, A., Benaddy, M., Ellahyani, A., Meslouhi, O.E. and Kardouchi, M., 2022. Isolated Video-Based Arabic Sign Language Recognition Using Convolutional and Recursive Neural Networks. *Arabian Journal for Science and Engineering*, *47*(2), pp.2187-2199.

Bradski, G. and Kaehler, A., 2000. OpenCV. *Dr. Dobb's journal of software tools*, *3*, p.2.

Chollet, F. & others, 2015. Keras. Available at: https://github.com/fchollet/keras.

Dabwan, B.A. and Jadhav, M.E., 2021, December. A Deep Learning based Recognition System for Yemeni Sign Language. In *2021 International Conference of Modern Trends in Information and Communication Technology Industry (MTICTI)* (pp. 1-5). IEEE.

DG Tech, n.d. *DG5 VHand 3.0* [online] Available at: http://www.dg-tech.it/vhand3/index.html.

ElBadawy, M., Elons, A.S., Sheded, H. and Tolba, M.F., 2015. A proposed hybrid sensor architecture for arabic sign language recognition. In *Intelligent Systems' 2014* (pp. 721-730). Springer, Cham.

ElBadawy, M., Elons, A.S., Shedeed, H.A. and Tolba, M.F., 2017, December. Arabic sign language recognition with 3d convolutional neural networks. In *2017 Eighth international conference on intelligent computing and information systems (ICICIS)* (pp. 66-71). IEEE.

Elons, A.S., 2014, December. GPU implementation for arabic sign language real time recognition using multi-level multiplicative neural networks. In *2014 9th International Conference on Computer Engineering & Systems (ICCES)* (pp. 360-367). IEEE.

Elons, A.S., Ahmed, M., Shedid, H. and Tolba, M.F., 2014, December. Arabic sign language recognition using leap motion sensor. In *2014 9th International Conference on Computer Engineering & Systems (ICCES)* (pp. 368-373). IEEE.

Google, n.d. *Improve Translate*. Available at: https://translate.google.com/intl/en/about/contribute/.

Google, 2019. MediaPipe. Available at: https://github.com/google/mediapipe.

Hamid Yousuf, F., Bushnaf Alwarfalli, A. and Ighneiwa, I., 2021, October. Arabic Sign Language Recognition System by Using Surface Intelligent EMG Signal. In *The 7th International Conference on Engineering & MIS 2021* (pp. 1-6).

Hamou, A.O. and Chelali, F.Z., 2021, November. Hand Gesture Recognition Based on LPQ and MRELBP Descriptors. In *2021 International Conference on Artificial Intelligence for Cyber Security Systems and Privacy (AI-CSP)* (pp. 1-6). IEEE.

Hayani, S., Benaddy, M., El Meslouhi, O. and Kardouchi, M., 2019, July. Arab sign language recognition with convolutional neural networks. In *2019 International Conference of Computer Science and Renewable Energies (ICCSRE)* (pp. 1-4). IEEE.

Hisham, B. and Hamouda, A., 2017. Arabic Static and Dynamic Gestures Recognition Using Leap Motion. *J. Comput. Sci.*, *13*(8), pp.337-354.

Hunter, J. D., 2007. Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering, vol. 9*, no. 3, pp. 90-95.

Ismail, M.H., Dawwd, S.A. and Ali, F.H., 2021. Static hand gesture recognition of Arabic sign language by using deep CNNs. *Indonesian Journal of Electrical Engineering and Computer Science*, *24*(1), pp.178-188.

Kamruzzaman, M.M., 2020. Arabic sign language recognition and generating Arabic speech using convolutional neural network. *Wireless Communications and Mobile Computing*, *2020*.

Kitchenham, B., 2004. Procedures for performing systematic reviews. *Keele, UK, Keele University*, *33*(2004), pp.1-26.

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B.E., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J.B., Grout, J., Corlay, S. and Ivanov, P., 2016. *Jupyter Notebooks-a publishing format for reproducible computational workflows* (Vol. 2016, pp. 87-90).

Latif, G., Mohammad, N., Alghazo, J., AlKhalaf, R. and AlKhalaf, R., 2019. Arasl: Arabic alphabets sign language dataset. *Data in brief*, *23*, p.103777.

Latif, G., Mohammad, N., AlKhalaf, R., AlKhalaf, R., Alghazo, J. and Khan, M., 2020. An automatic Arabic sign language recognition system based on deep CNN: an assistive system for the deaf and hard of hearing. *International Journal of Computing and Digital Systems*, *9*(4), pp.715-724.

Lucidchart, n.d. Available at: www.lucidchart.com

Luqman, H., El-Alfy, E.S.M. and BinMakhashen, G.M., 2021. Joint space representation and recognition of sign language fingerspelling using Gabor filter and convolutional neural network. *Multimedia Tools and Applications*, *80*(7), pp.10213-10234.

Microsoft, n.d. *Kinect* [online]. Available at: https://developer.microsoft.com/en-us/windows/kinect/.

MOCDUAE, 2020. *تعلم لغة الإشارة- أسماء الإمارات السبع* [Learn Sign Language – The Seven Emirates] [Online]. Available at: https://www.youtube.com/shorts/wZVJDXMnwxM.

Mohandes, M., Aliyu, S. and Deriche, M., 2014, June. Arabic sign language recognition using the leap motion controller. In *2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE)* (pp. 960-965). IEEE.

Moher, D., Liberati, A., Tetzlaff, J. and Altman, D.G., 2010. Preferred reporting items for systematic reviews and meta-analyses: the PRISMA statement. *Int J Surg*, *8*(5), pp.336-341.

Mustafa, M., 2021. A study on Arabic sign language recognition for differently abled using advanced machine learning classifiers. *Journal of Ambient Intelligence and Humanized Computing*, *12*(3), pp.4101-4115.

Ouzzani, M., Hammady, H., Fedorowicz, Z. and Elmagarmid, A., 2016. Rayyan—a web and mobile app for systematic reviews. *Systematic reviews*, *5*(1), p.210.

Renotte, N., 2021. *Action Detection for Sign Language* [Source code]. Available at: https://github.com/nicknochnack/ActionDetectionforSignLanguage.

Ritonga, M., Abd El-Aziz, R.M., Alazzam, M.B., Alassery, F., NagaJyothi, A., Abd Algani, Y.M. and Balaji, S., 2021. Machine Learning Approach for Gesture Based Arabic Sign Language Recognition for Impaired People. *International Journal of Advanced Computer Science and Applications*, *12*(12).

Sadeddine, K., Chelali, F.Z. and Djeradi, R., 2015, May. Sign language recognition using PCA, wavelet and neural network. In *2015 3rd International Conference on Control, Engineering & Information Technology (CEIT)* (pp. 1-6). IEEE.

Sadeddine, K., Chelali, F.Z., Djeradi, R., Djeradi, A. and Benabderrahmane, S., 2021. Recognition of user-dependent and independent static hand gestures: Application to sign language. *Journal of Visual Communication and Image Representation*, *79*, p.103193.

Sadeddine, K., Djeradi, R., Chelali, F.Z. and Djeradi, A., 2018, May. Recognition of static hand gesture. In *2018 6th International Conference on Multimedia Computing and Systems (ICMCS)* (pp. 1-6). IEEE.

Samir, A., Aboulela, M. and Tolba, M., 2014. A Proposed Model for Standard Arabic Sign Language Recognition Based on Multiplicative Neural Network. *The Egyptian Journal of Language Engineering*, *1*(2), pp.1-10.

Schardt, C., Adams, M.B., Owens, T., Keitz, S. and Fontelo, P., 2007. Utilization of the PICO framework to improve searching PubMed for clinical questions. *BMC medical informatics and decision making*, *7*(1), p.16.

Suliman, W., Deriche, M., Luqman, H. and Mohandes, M., 2021, December. Arabic Sign Language Recognition Using Deep Machine Learning. In *2021 4th International Symposium on Advanced Electrical and Communication Technologies (ISAECT)* (pp. 1-4). IEEE.

Tharwat, G., Ahmed, A.M. and Bouallegue, B., 2021. Arabic Sign Language Recognition System for Alphabets Using Machine Learning Techniques. *Journal of Electrical and Computer Engineering*, *2021*.

UltraLeap, n.d. *Leap Motion Controller*. [online] Available at: https://www.ultraleap.com/product/leap-motion-controller

Wadhawan, A. and Kumar, P., 2021. Sign language recognition systems: A decade systematic literature review. *Archives of Computational Methods in Engineering*, *28*(3), pp.785-813.

Appendices

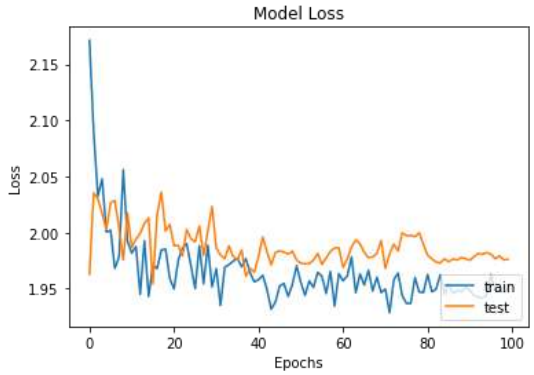
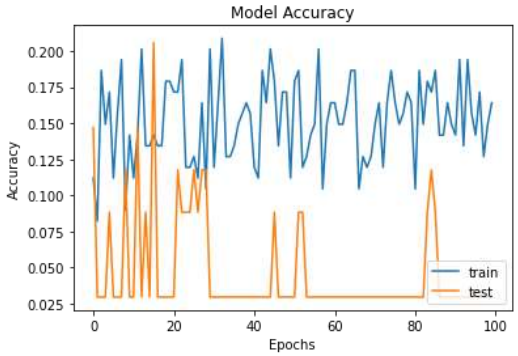
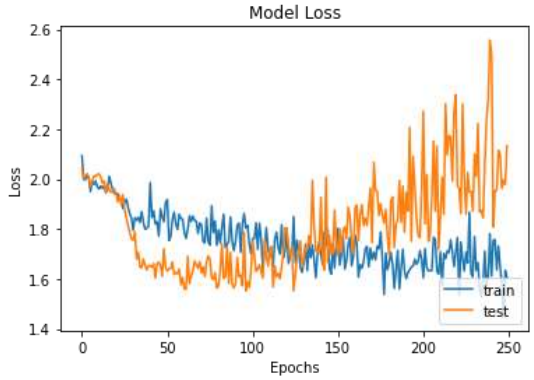
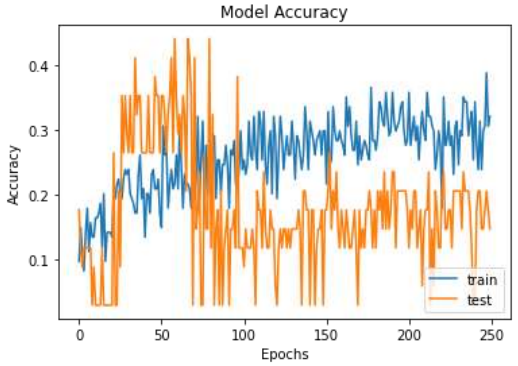
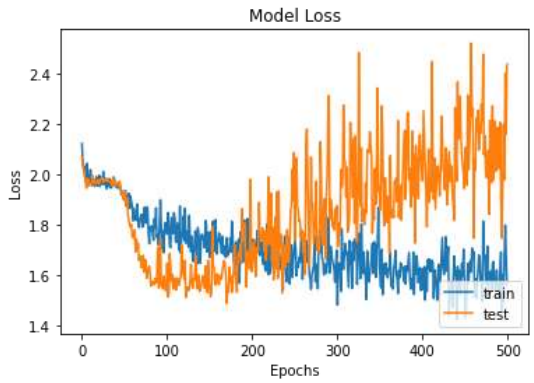
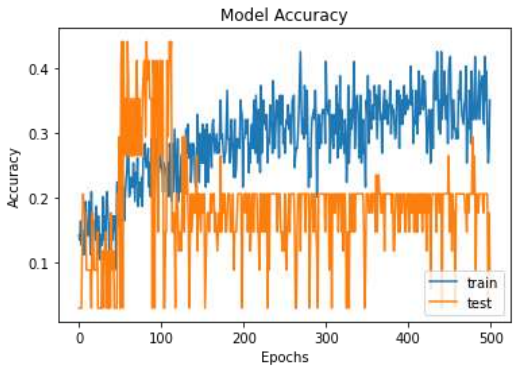
Appendix A

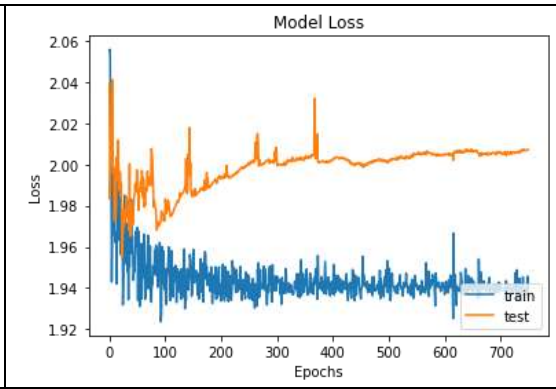
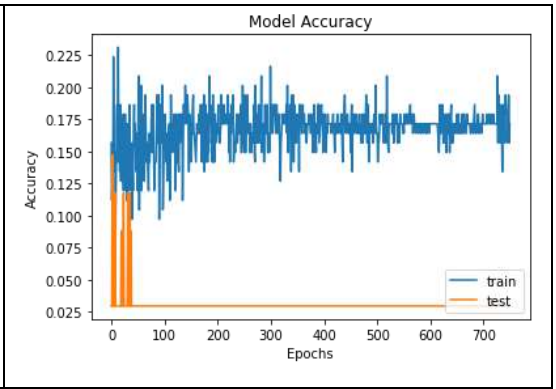
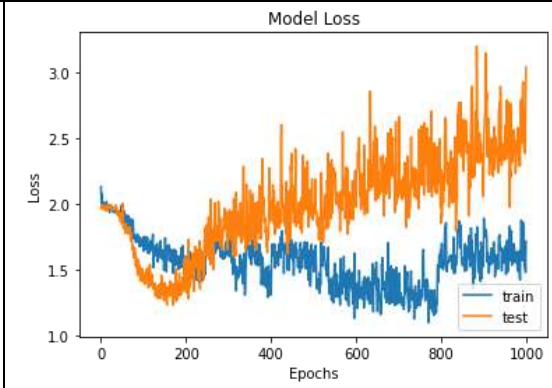
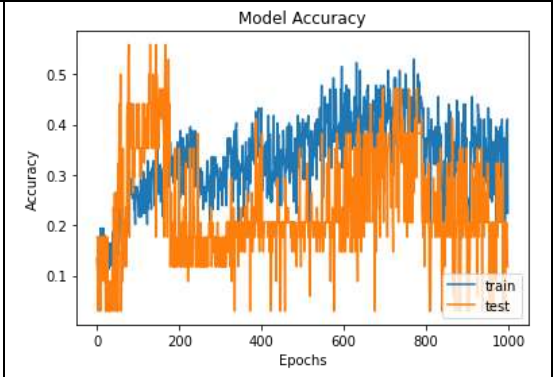
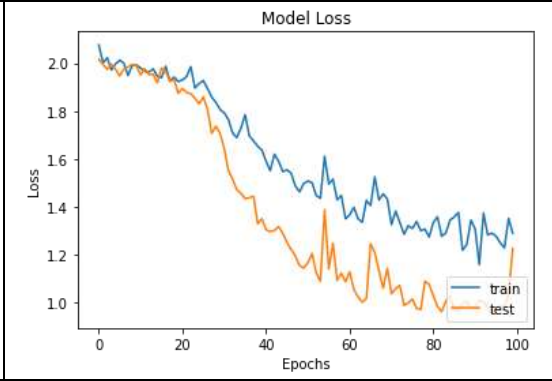
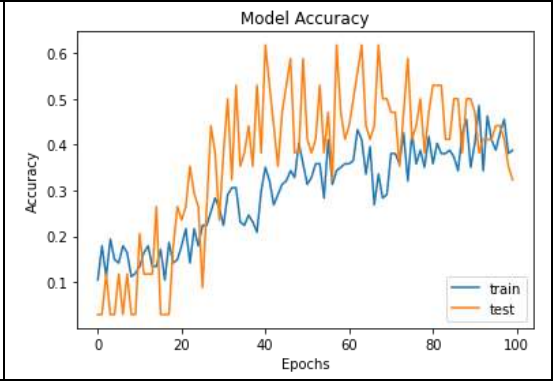
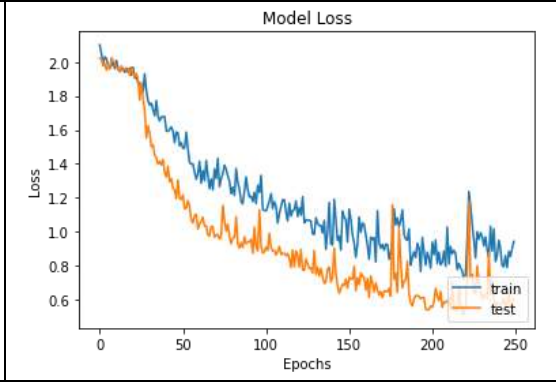
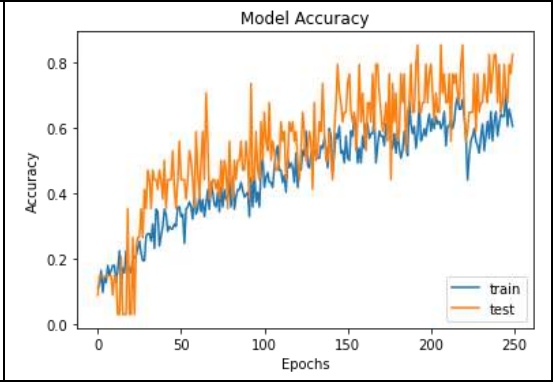
Below table presents the exact performance for all models presented in figure 15 from lowest to highest.

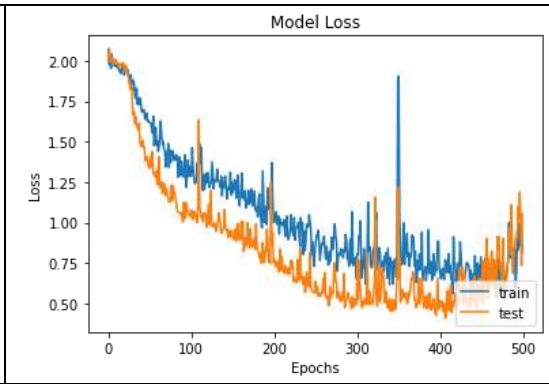
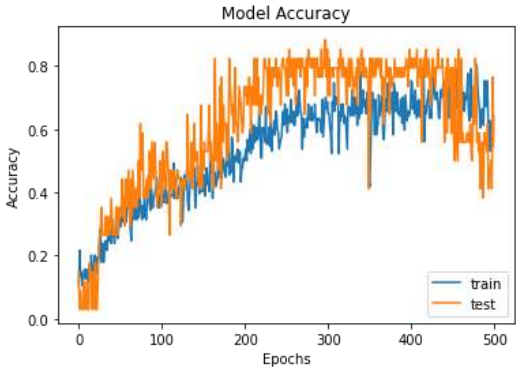
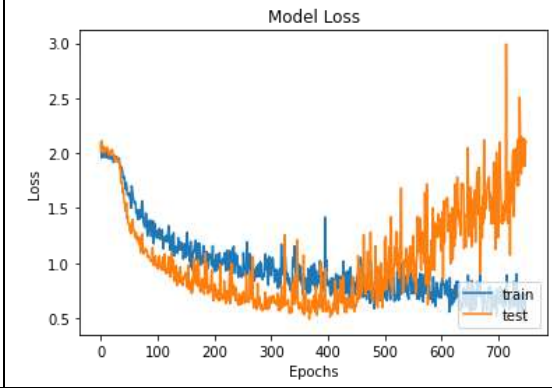
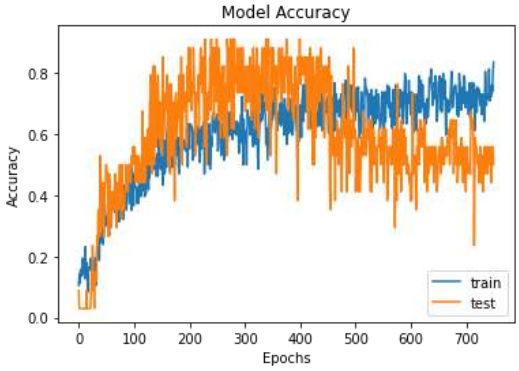
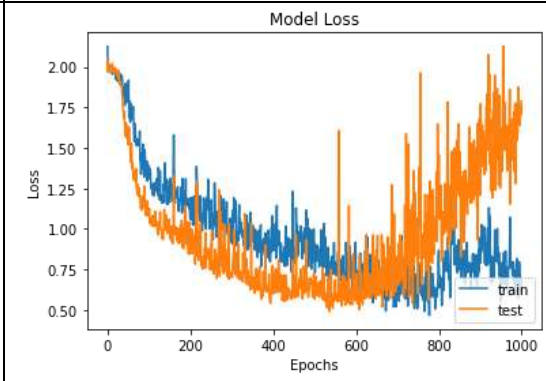
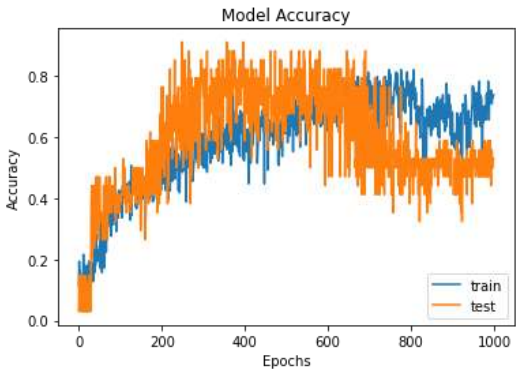
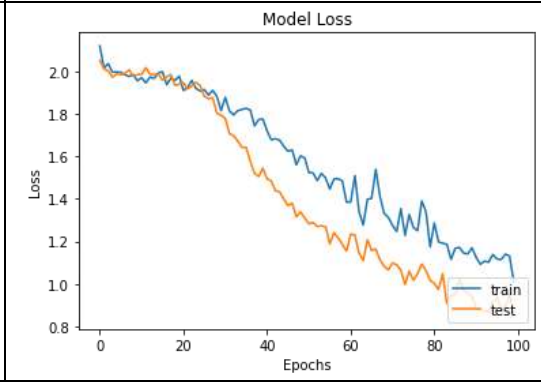
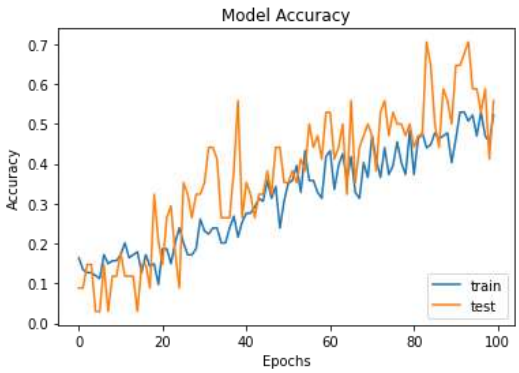
Classifier	Recognition Rate
TDNN	70.00%
Inception-BiLSTM	78.90%
PNN	80.20%
RBF-NN	81.33%
MNN	83.00%
3DCNN	85.00%
MLP	87.60%
MLP	88.00%
BiLSTM	89.50%
CNN	90.00%
CNN	90.00%
CNN	90.02%
ANN	90.66%
2DCRNN	92.00%
ANN	92.95%
RCNN	93.00%
CNN	94.00%
MMNN	94.00%
MLP	94.53%
MLP	95.00%
LSTM	95.90%
LSTM	96.00%
CNN	96.40%
ANN-MP	96.90%
CNN	97.29%
CNN	97.60%
Skeletal CNN	98.39%
3DCNN	99.00%
MLP	99.00%
CNN	99.05%
CNN	100.00%

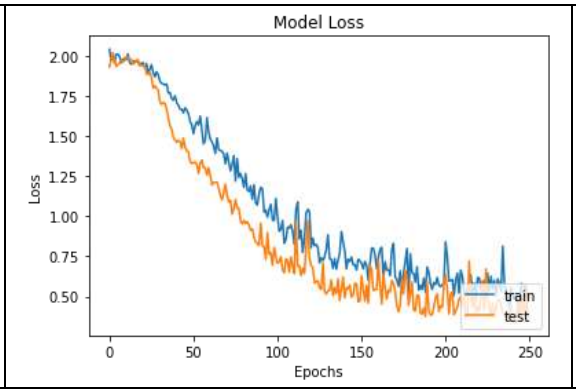
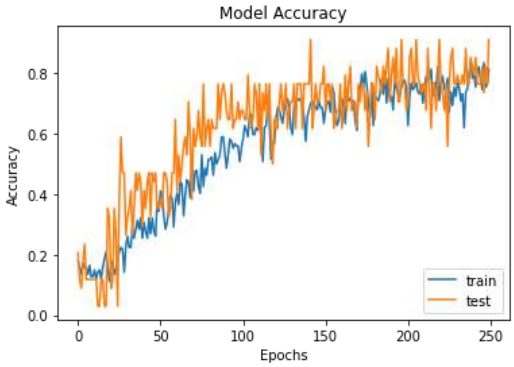
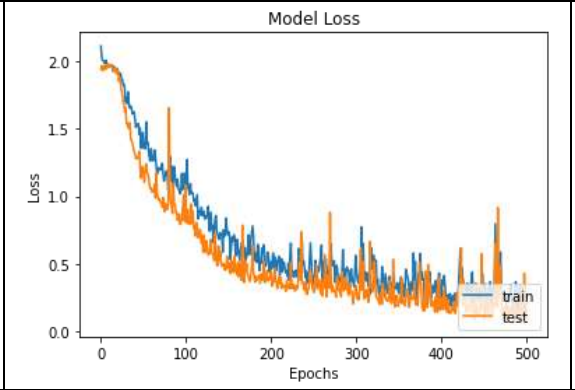
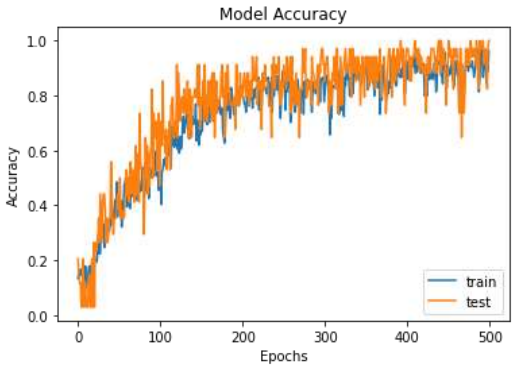
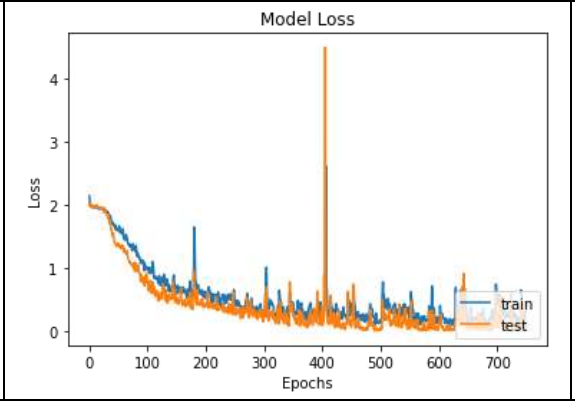
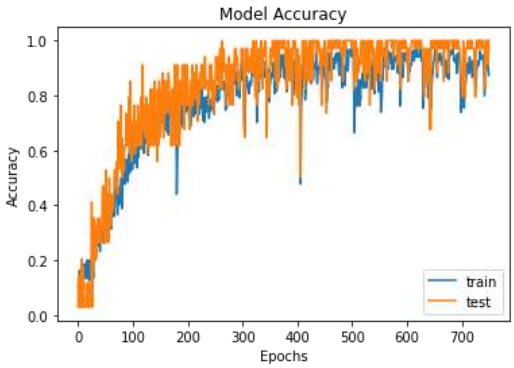
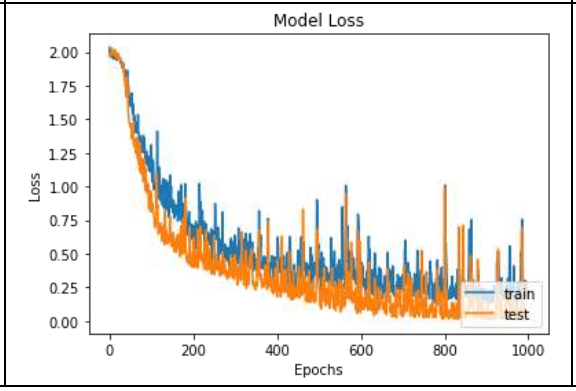
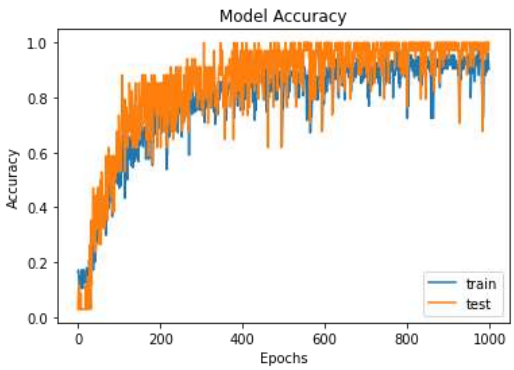
Appendix B

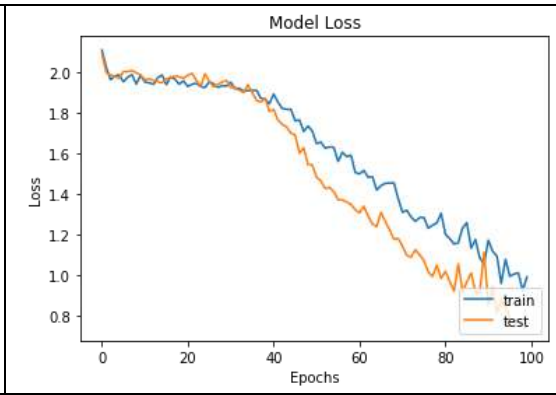
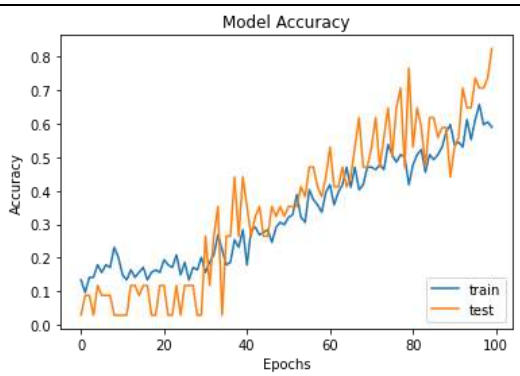
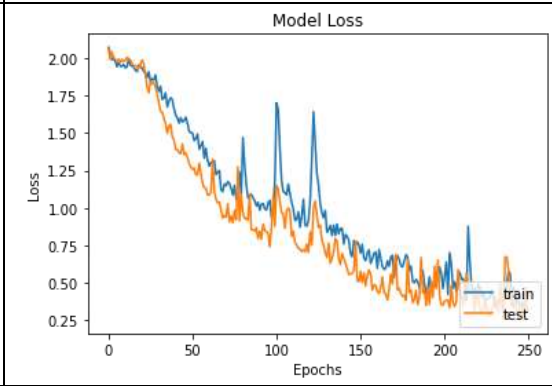
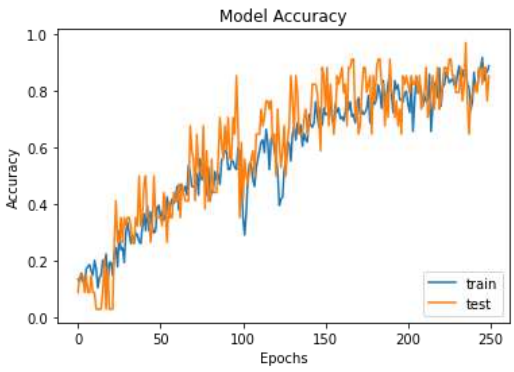
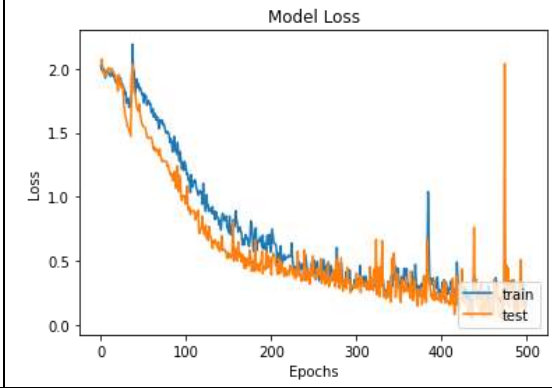
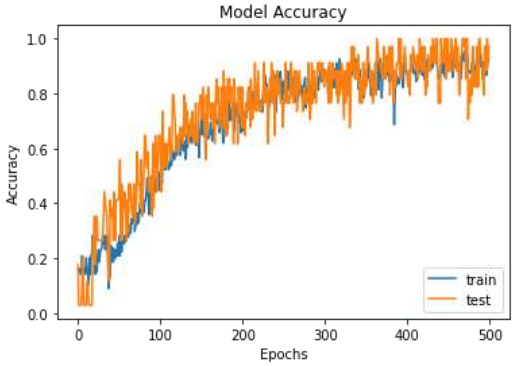
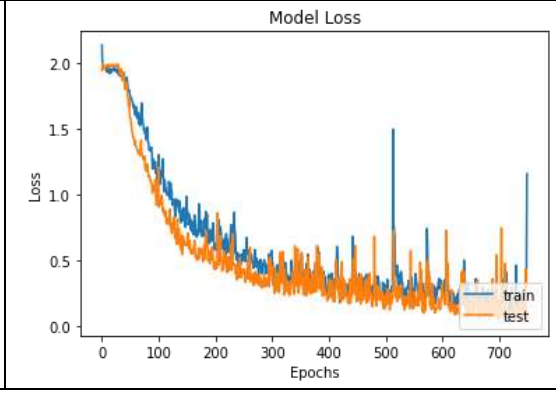
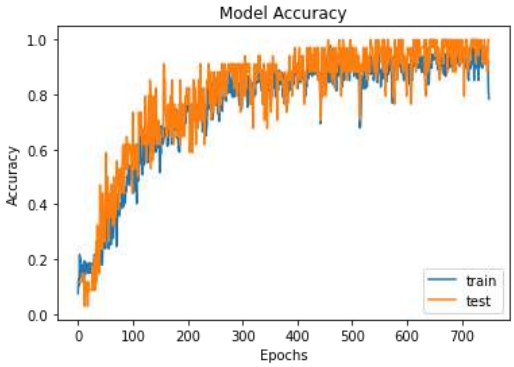
Below tables show the loss and accuracy graphs for the different batch sizes and epochs presented in table 13.

Training Iteration	Loss	Accuracy
Batch size = 2 Epochs = 100		
Batch size = 2 Epochs = 250		
Batch size = 2 Epochs = 500		

Batch size = 2 Epochs = 750		
Batch size = 2 Epochs = 1000		
Batch size = 4 Epochs = 100		
Batch size = 4 Epochs = 250		

Batch size = 4 Epochs = 500		
Batch size = 4 Epochs = 750		
Batch size = 4 Epochs = 1000		
Batch size = 8 Epochs = 100		

Batch size = 8 Epochs = 250		
Batch size = 8 Epochs = 500		
Batch size = 8 Epochs = 750		
Batch size = 8 Epochs = 1000		

Batch size = 16 Epochs = 100		
Batch size = 16 Epochs = 250		
Batch size = 16 Epochs = 500		
Batch size = 16 Epochs = 750		

Batch size = 16 Epochs = 1000