National College of Ireland

# Irish Sign Language Gesture Classification Modelling Using Skeleton Tracking Techniques

MSc Research Project
Data Analytics


Cian Vaughan
Student ID: 17139953

School of Computing
National College of Ireland


Supervisor: Dr. Catherine Mulwa

**National College of Ireland**
**Project Submission Sheet**
**School of Computing**

| | |
|---|---|
| **Student Name:** | Cian Vaughan |
| **Student ID:** | 17139953 |
| **Programme:** | Data Analytics |
| **Year:** | 2019 |
| **Module:** | MSc Research Project |
| **Supervisor:** | Dr. Catherine Mulwa |
| **Submission Due Date:** | 12/08/2019 |
| **Project Title:** | Irish Sign Language Gesture Classification Modelling Using Skeleton Tracking Techniques |
| **Word Count:** | 8656 |
| **Page Count:** | 25 |

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

**ALL** internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

| | |
|---|---|
| **Signature:** | |
| **Date:** | 8th August 2019 |

## PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

| | |
|---|---|
| Attach a completed copy of this sheet to each project (including multiple copies). | ☐ |
| **Attach a Moodle submission receipt of the online project submission**, to each project (including multiple copies). | ☐ |
| **You must ensure that you retain a HARD COPY of the project**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. | ☐ |

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

| **Office Use Only** | |
|---|---|
| Signature: | |
| Date: | |
| Penalty Applied (if applicable): | |

# Irish Sign Language Gesture Classification Modelling Using Skeleton Tracking Techniques

Cian Vaughan
17139953
MSc Research Project in Data Analytics

12th August 2019

**Abstract**

Irish Sign Language (ISL) is a recognised national language and the native language of the deaf and hard of hearing people of Ireland. Despite this, it remains disconnected from technology. It is of significant importance that there exists a means for the deaf community to communicate fluently with their devices. The communication of sign language utilises many features, and as such interpreting sign language requires many distinct components. One essential component that is currently lacking in research is the analysis of movement. This research project used a depth-camera to capture skeleton tracking data of ten commonly-signed ISL gestures and trained neural networks to classify them. The predictive models generated combined the strengths of convolutional and recurrent neural networks to create structures suitable for the analysis of spatio-temporal data. All models achieved an above-90% accuracy in gesture predictions, with the best model (ConvLSTM) incorporating both recurrent and convolutional operations to achieve an accuracy 96.408%. These models compare favourably with other ISL-gesture classification research in the field (71% Hernández (2018)). The intention is that this work can be coordinated with existing ISL image analysis research to create a complete ISL classifier. The project also featured a review of the relevant literature and concluded on the research gaps concerning ISL-gesture analysis.

## 1 Introduction

Rapid progress is being in the domain of Human-Computer Interaction (HCI) and society is becoming increasingly entangled with technology. It is becoming essential that everyone feels equally comfortable interacting fluently with computers. Huge advances in speech analysis and translation services have allowed people to communicate with their devices as seamlessly as they would another human being. However, certain members of society are being left behind. For the deaf community of Ireland, Irish Sign Language is their native language. It is used by over 5,000 deaf people in Ireland, as well as 50,000 non-deaf people (Leeson and Saeed, 2012). The recognition of sign language by computers is a multi-faceted problem; one that requires many distinct working components to create a holistic model. This research project focused on movement and the recognition of Irish Sign Language gestures. Recognition involved sequence classification performed by machine learning algorithms on a specially created skeleton tracking dataset.

## 1.1 Motivation and Project Background

Most sign language interpretation is performed using either still images, specialised gloves (colour-coded or using a network of sensors), or computationally-demanding video analysis. Unfortunately, each of these approaches comes with its own limitations. The gloves, in particular those with the in-built sensors, require the user to be wearing expensive specialised equipment while signing. The cost and expertise required can be prohibitive. Of the non-invasive solutions, image analysis doesn't acknowledge movement. Movement is one of the five components used to communicate a signed word - the others being *handshape*, *orientation*, *location*, and *facial expression* (Stokoe, 1965). Of the solutions currently on the market, those perceiving movement and contextually-aware analysis are lacking. Not only is there strong business potential in being the first to market with an accurate Irish Sign Language recogniser, but it is a matter of accessibility and equal opportunities for all citizens. The Irish government recognised ISL as an official state language in 2017, and has stated its commitment to providing services for the deaf community in its native language.

The problem of detecting gestures is difficult for a number of reasons. Firstly, there is the challenge of handling an extremely high-dimensional dataset. Regardless of the form the data takes, it must capture the full range of motion of the signer, for several seconds per word. The variance between each instance of a signed word, and between one signer and the next, can be quite large. Then, there is the matter of sequence classification which is more complex than regular classification. This is due to the added significance of the order of the data. Uniquely, Irish Sign Language has also been recognised for having almost mutually unintelligible languages between the male and female communities (LeMaster, 1998). This phenomenon was likely due to segregation of the boys' and girls' schools in Dublin during the 80s, but was noted to be diminishing by the late-90s.

Video analysis is a very computationally-intensive process, requiring computers with high-end graphics processing units (GPUs). This can take a long time to train, and can be entirely beyond the capabilities of many researchers. Additionally, video analysis, being limited to two spatial dimensions, can struggle with many gestures that move along the z-axis (towards or away from the observer). In this case, depth perception is essential for full comprehension, and much research in the area omits these signs when testing their model's accuracy.

Research on American Sign Language (Stokoe, 1965) identified twenty distinct movement types that are used in the language. These include, but are not limited to: lateral movements, twisting, flexing the wrist, circling, opening and closing the hand, touching, crossing, and wriggling. It also showed the speed, abruptness and intensity mattered for word comprehension. All these crucial movements are lost when the problem is reduced to static image analysis.

## 1.2 Research question

The research question investigated the gaps in the relevant literature and delivered an innovative solution to the device-interaction issues faced by the deaf community in Ireland. The scope of this research project was to classify ten commonly-used gestures of Irish Sign Language, covering a variety of these movements, including movements along the z-axis. These gestures signified: how, name, jump, get up, sit down, please, and thank you; and the letters J, X and Z. The research question that follows addressed the gaps in the body of knowledge regarding ISL-classification problems.

RQ: *How can recognition of Irish Sign Language gestures be improved through the use of time-varying 3D positional data and machine learning techniques (MLPs, LSTMs, CNN-LSTMs, and ConvLSTM)?* Different neural network architectures will be implemented, evaluated and the results compared for accuracy in recognising Irish Sign Language gestures.

Sub-RQ: *To what extent can pre-trained machine learning models be used to recognise Irish Sign Language gestures from an unseen signer?* A new dataset will be created, with a different person signing the gestures. The developed machine learning models will be tested and evaluated on this new dataset, having only been trained on the previous dataset. This will be referred to as *signer independent testing*.

Movement is a critical component for the communication of information from one deaf person to another. This ICT solution satisfies a niche in the market for an ISL-gesture classifier, which helps address these issues faced by the deaf community in Ireland.

## 1.3 Project Objectives and Contributions

In order to address the research questions, project objectives were defined and executed as shown in Table 1.

Table 1: Project objectives

| Obj. | Objectives | Methods | Evaluation Metrics |
|---|---|---|---|
| 1 | The contributions to the literature will be results of critically reviewed literature on Irish sign language recognition and existing techniques (2007-2019) | | |
| 2 | Create a dataset of ISL words using depth-camera and skeleton tracking technology | | |
| 3 | Build machine learning models to classify Irish Sign Language gestures | | |
| (3.1) | Implement, evaluate and results of an MLP architecture (LSTM) | MLP, LSTM, LSTM+, CNN-LSTM, Conv-LSTM | Accuracy, Categorical Cross-Entropy Loss, Confusion Matrix |
| (3.2) | Implement, evaluate and results of an LSTM architecture. | | |
| (3.3) | Implement, evaluate and results of an Stacked-LSTM architecture (LSTM+). | | |
| (3.4) | Implement, evaluate and results of a CNN-LSTM architecture. | | |
| (3.5) | Implement, evaluate and results of an convolutional-LSTM architecture (ConvLSTM). | | |
| (3.6) | Implementation, evaluation and results of signer independent testing | | |
| 4 | Comparison of developed models (Objective 3) | | |
| 5 | Comparison of developed models (Objective 4) vs. existing models | | |

These objectives cover a comprehensive critique of the current literature on the topic, a contribution of a dataset of ISL-signed gestures to the body of knowledge, and the

development of innovative modelling techniques for classifying the sequential skeleton-tracking data.

**Contributions:** Multiple models (MLP, LSTM, Stacked-LSTM, CNN-LSTM, Conv-LSTM) for recognising sign language gestures; A dataset of ISL signed words and phrases (500 samples, 10 classes) captured in skeleton tracking data points; an additional dataset of 100 samples from a different signer.

# 2 Literature Review on Sign Language Recognition 2005-2019

## 2.1 Introduction

This literature review will be broken into the following sub-sections: (2.2) critique of Irish Sign Language classification methodologies, techniques and identified gaps; (2.3) an investigation of Sign Language gesture classification techniques and identified gaps; (2.4) a comparison of techniques, models and results of Irish Sign Language recognition; and (2.5) conclusion.

Comprehension of sign language utilises various different techniques. Of the supervised training techniques, a typical example (Dong et al., 2015) would be utilising labelled data (in this case American Sign Language alphabet letters) and train a classification algorithm to recognise unseen letter shapes. Other potential techniques could be time series analysis (predicting future position of gestures), association rules, outlier detection, and clustering (grouping similar gestures together). The potential use cases of these are somewhat limited, and as such the vast majority of research involving sign language recognition utilises classification techniques. Unsupervised techniques can be considered in tandem with the aforementioned classification techniques. Such techniques include dimensionality reduction for reducing a large sample space (Zaki and Shaheen, 2011), outlier detection for removing stray data points (Estiri and Murphy, 2018), and independent component analysis for identifying key contributing features of sign language gestures. For the purposes of solving this research question, the machine learning method of classification was the most suitable. The scope of the following review of research in the area of sign language classification was focussed on applications to Irish Sign language.

## 2.2 A Critique of Irish Sign Language Classification Methodologies, Techniques, and Identified Gaps

Existing solutions in the area of sign language recognition can be categorised twice: whether they utilise static or time-dependent data; and whether their inputted feature space is vision-based or sensor-based. Of these two methodologies, Oliveira et al. (2017) chose the vision-based approach. In this case, a video dataset was created, however, it was used as a source for the extraction of still images. The classification model was trained purely on static images of hand-shapes of the alphabet and so does not explore time-dependent models. This leaves a gap for a classification tool to recognise the three ISL letters involving dynamic gestures (J, X and Z). The research was also limited to purely 2D vision.

Kelly (2010) developed a solution from a specially-created ISL dataset, and produced a new framework for recognising hand-shapes independent of the person signing. Un-

fortunately, this ISL-classification research project was also confined to the analysis of static images, which limited the ability of the model to learn words and phrases involving movement. Further work (Kelly et al., 2009) to address this issue of gesture recognition involved signers wearing coloured gloves and using this to extract features from the movement. Hernández (2018) also completed image recognition models to high accuracy, but in addition to image analysis, she also produced methodologies for extracting and analysing features from videos. This was performed through the use of convolutional neural networks (CNNs). CNNs have proven very successful in the domain of image classification and can be augmented to consider time-varying data.

Stein et al. (2017) trained a model on videos of both American Sign Language (ASL) and ISL sentences. Motion was acknowledged using a tracking algorithm guided by hand velocity and trajectory. This involved scaling the analysed region down to a $20 \times 20$ pixel window. Their model yielded a lowest overall word error rate (WER) of 17.9%. However, since the results and conclusions of the ASL- and ISL-models cannot be disentangled from each other, it is omitted from our comparison of ISL classifiers.

While there is evidence of research producing impressive results for image recognition of Irish Sign Language words, it is clear that there is a gap for a methodology that effectively recognises ISL gestures with minimal computational overhead. The research of time-series gestural data that was conducted involved training highly-intensive convolutional neural networks. This can be prohibitively costly and time-consuming, so alternative techniques for gestural recognition are in high-demand.

## 2.3 Investigation of Sign Language Gesture Recognition Techniques and Identified Gaps

Research into the recognition of gestures can be separated by their choice in methodology in dealing with the time-dependent component of the data. Much of the research in the domain of sign language classification focuses on either on the classification of static images of signed words or reduces videos to a series of static images. Classification of sequential data requires different considerations from purely static image classifiers. Such static image analysis (Potkin and Philippovich, 2018) of sign language, as with most modern image classification, is usually performed through convolutional neural networks. The area of gesture recognition, however, has an additional requirement of contextual awareness - the order of the time-series data is crucial. In this case, traditional image classification techniques will no longer suffice as treating video purely as a series of independent images neglects the sequential nature of the data.

Darwish et al. (2016) created gestural recognition models using features extracted from images, which the author concedes left the model "prone to be influenced by illumination changes and complicated backgrounds", among other factors. Rahagiyanto et al. (2017) trained a neural network model on the Indonesian Sign language alphabet to outperform a k-Nearest Neighbours model for classification accuracy by 93.08% to 82.31%. The analysis was performed on sensor data, gathered from the accelerometer and gyroscopes from a Myo Armband.

While 2D CNNs are suitable for flat image classification, adding the component of time requires analysis along the temporal feature space. Huang et al. (2015) created a 3D CNN structure to handle the sequential stacking of the layers. The models achieved an accuracy of 94.2% in classifying 25 words from a training set of 450 samples. One methodology (Pahlevanzadeh et al., 2007) extrapolated from image recognition techniques

and used CNNs to tackle the spatio-temporal challenge of motion recognition. Here, two layers of CNNs were used: one for hand-shape recognition, and another dedicated to detecting and tracking hand motion. A similar solution was devised by Tran et al. (2017) which they dubbed the (2+1)D CNN. When applied to video this can be a very intensive process. Research by Sigit et al. (2016) improved the spatial tracking of sign language gestures through optical flow tracking in three dimensions. This was done by tracking the variation in light intensity. As with Darwish et al. (2016), this is likely to create a model vulnerable to variable lighting conditions.

Muhammad et al. (2016) present a prototype use-case that would use Microsoft Kinect camera for sign language recognition. The paper functions adequately as a proof of concept, however, no testing or accuracy measurements are offered. Similar research by Ahmed et al. (2016) used Microsoft's skeleton tracking software to recognise signed words, with an accuracy of 87% based off 100 samples from three different signers. All analysis was performed using Microsoft Kinect V2's Continuous Gesture Builder which meant the research was locked-in to the application and lacked the ability to construct the architecture and tune the hyperparameters. HMMs were used and perhaps could have benefited from more complex neural networks models. Research by (Dong et al., 2015) used the Kinect to capture depth-images of the ASL alphabet and then extracted features for training. Although the depth feature of the camera was used to extract hand-shape for still images, the principle could feasibly be applied to depth movements in video. In the research of Zhao et al. (2017), action tracking was performed by projecting a skeleton object onto the subject and using this to track their motion. An ideal solution would utilise the Kinect for feature extraction, pulling the raw skeleton data. By getting access to the raw data it would allow the development of bespoke models outside of the confines of the Microsoft Kinect environment.

Research by Liu et al. (2016) experimented with using this skeleton tracking data with a type of artificial neural network architecture called a recurrent neural network (RNN). Their reasoning for this was that the recurrent feature of the RNN architecture allows for a memory of previous states to inform the neural networks' decisions. Thus giving the network contextual awareness, essential for time-series data analysis. The particular architecture of RNN applied here is called a Long Short-Term Memory (LSTM) model. In this research, it showed significantly improved accuracy over standard HMMs (Hidden Markov Models). Using high-resolution glove-sensor data, Cate et al. (2015) also employed LSTM models as a significant improvement on their SVM models in order to recognise the temporal features of their dataset. Yang and Zhu (2017) designed what could be considered a best-of-both-worlds solution, by incorporating the strengths of a fully-connected CNN layer for image analysis with the ability of LSTMs to learn from sequential data. Donahue et al. (2015a) recognises the increasing importance of moving beyond static image analysis and proposes another architecture: Long-Term Convolutional Networks (LTCN).

An ISL models discussed in section 2.2, (Kelly, 2010), used Hidden Markov Models to classify the signed words. There is research to suggest that for limited datasets HMM can outperform LSTM models for activity recognition(Alp and Keles, 2019), however the complexity of neural networks architectures allows the performance of the LSTM model to continue increasing for increased training samples. Panzner and Cimiano (2016) showed that once the training sets reached beyond 23 samples the LSTM models began to outperform the HMMs, and create ultimately more robust classification models. This is backed up by recent research in the area (Parcheta and Hinarejos, 2018) which has

shown that the results from HMMs can generally be outperformed by neural networks.

This review of the research in the area suggests there is a gap in the literature for a solution utilising a combination of 3D video and dimensionality reduction at point of data capture. The resulting sequential data would likely be best interrogated by RNN or CNN models, or some combination of the two.

## 2.4 A Comparison of Techniques, Models and Results for Irish Sign Language Recognition

Of the discussed experiments in ISL-recognition techniques, some yielded very high accuracy models. However, there are a number of limitations inherent in the design of each of these experiments. The initial experiments of Hernández (2018) involved using only static images to produce recognition models. This omits much of the language that involves gesture and motion. Of the video analysis experiments by Hernández (2018), 3D CNNs were used to extract spatial and temporal data for model training. This is a highly computationally intensive process and could be avoided entirely if the dimensionality reduction is performed at the moment of data-capture. None of the research utilised 3D video to capture motion along the z-axis and provide more robust gesture tracking.

Table 2: Comparison of ISL-Classification Research Projects

| **Features** | **Classifiers** | **Accuracy** | **Author** |
|---|---|---|---|
| Images (Alphabet) | SVM | 97.3% | Kelly (2010) |
| Videos (Gestures) | GTHMM | 97.7% | Kelly et al. (2009) |
| Images (Alphabet) | PCA & SVM | 99.875% | Oliveira et al. (2017) |
| Images (Alphabet) | CNN | 99.98% | Hernández (2018) |
| Videos (Gestures) | 3D CNN & LSTM | 79.66% | Hernández (2018) |

Table 2 presents a comparison of all the Irish Sign Language classifiers developed to date. The table lists the feature space analysed (image or video), as well as the algorithms used (SVM, CNN, LSTM, etc.), and their accuracies. The highest accuracy achieved incorporating movement was by Kelly (2010), who achieved 97.7%, however, this involved the signer wearing specialised coloured gloves. The most robust spatio-temporal computer-vision classification model was by Hernández (2018). This achieved an accuracy of 79.66% in classifying videos, through the use of 3D CNNs to classify sequences from spatial and temporal data.

## 2.5 Conclusion

Based on this review of the literature there is a definite gap in the field for a serviceable methodology in classifying Irish Sign Language gestures. There are identified gaps in 3D tracking, dimensionality reduction at the point of data capture, and temporal-gestural analysis. This marks the completion of Objective 1 from Table 1, Section 1.3. The following chapter will describe the scientific methodology used for the development of said sign language classification algorithm.

# 3 Scientific Methodology Approach Used and Design

## 3.1 Introduction

Data mining research typically follows either a KDD or CRISP-DM methodology. As this project was focused on knowledge discovery, as opposed to the iterative business process of product development, the approach was designed from a modified KDD methodology. This new methodology is presented in Figure 1. The architecture presented in Figure 2 consists of a three-tier structure (a client tier, application tier, and the data persistent tier) and section 3.3 will discuss the design decisions involved in choosing this. The resulting methodology will henceforth be referred to as the Irish Sign Language Recognition Methodology. The dataset used was unsupervised; created specifically for this research project.

## 3.2 Irish Sign Language Recognition Methodology

The Irish Sign Language Recognition Methodology (Figure 1) was modified from the Knowledge Discovery and Data Mining Methodology. This methodology consists of the following stages: (i) data recording: where the data (skeleton tracking) is captured by a Kinect camera and saved in .txt format (ii) feature selection: data is then collated, cleaned, and structured in a format suitable for the next stages of analysis, (iii) pre-processing: the data is then prepared into a format suitable for analysis, (iv) transformation: this data is then transformed to optimise model performance, (v) data mining: the model architectures are then designed and the models trained on the data using Keras and Tensorflow, (v) evaluation: the findings are then evaluated and the results interpreted.

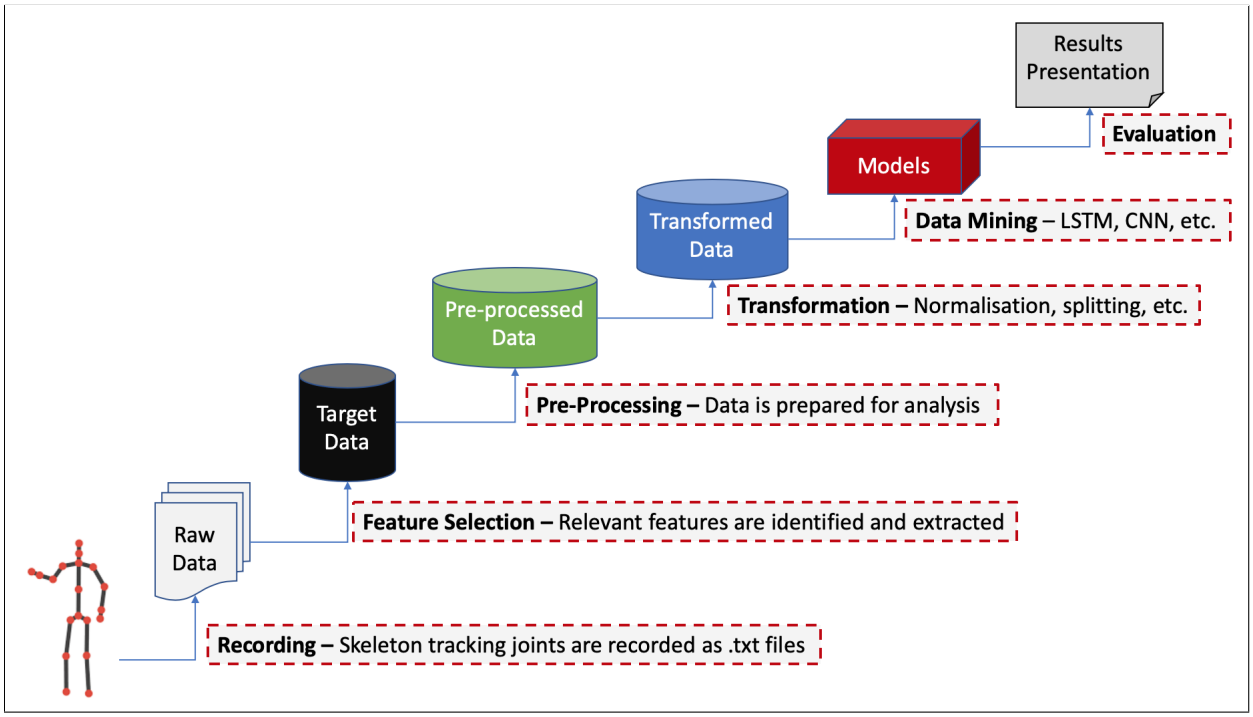



Figure 1: Irish Sign Language Recognition Methodology

## 3.3 Architectural Design

The project architectural design (Figure 2) for classifying Irish Sign language gestures consists of a client tier, a business logic tier, and a data persistent tier.

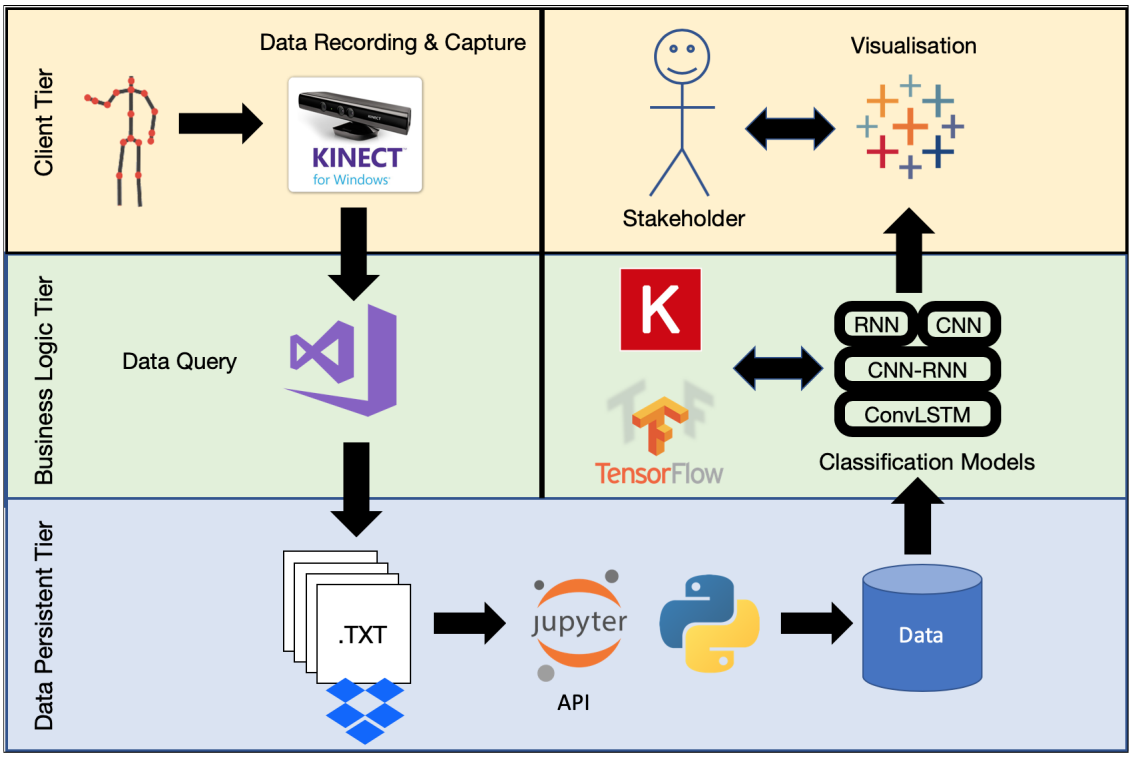



Figure 2: Project Multitier Architecture

The decision was made to begin the data lifecycle journey in the client tier to involve the user in the feedback nature of the classifier. The flow would begin with the user signing ISL gestures into a Kinect camera. From here, the data would be queried in the business logic tier using Microsoft Visual Basic, Kinect Studio and code sourced from GitHub. This is done to capture the data, and ultimately pass it to the third tier: the data persistent tier. The cloud sharing platform Dropbox was chosen as a convenient way to store all captured files in a location that could be easily accessed from any device. Once in the data persistent tier, the data was pre-processed and exported for use by the next tier. The data is then queried by the business logic tier, where it was normalised, transformed, reshaped, and used for model training. The models were neural network classifiers: a multi-layer perceptron, s recurrent neural network, a convolutional neural network, a convolutional-long short-term network, and experimental modifications and hybrids. This was performed using a TensorFlow back-end, accessed via the Keras API. Within this tier, models are trained and tested on the processed dataset. The results are then returned to the client tier where the stakeholder can interact with them via Tableau dashboards.

## 3.4 Conclusion

A specialised methodology, the Irish Sign Language Recognition Methodology, was designed for this research project. The 3-tier architecture was used as the project architecture, with the data flowing from the client tier, down to the data persistent tier, and

returning back to the client tier. The implementation, evaluation, and results of the generated ISL gesture classification models are presented in the next section.

# 4 Implementation, Evaluation and Results of Irish Sign Language Classification Models

## 4.1 Introduction

The implementation, evaluation and results of the models used in classifying Irish Sign Language gestures from skeleton tracking data are discussed in this section. This section will cover the creation of the dataset, the pre-processing and transformation of the feature space, and the implementation of the generated models. The models will be used as classification agents for the dataset. A critical comparison of the models was performed using classification accuracy as a metric. Further study was done on the models through the examination of confusion matrices.

Firstly, a synopsis of the aspects of implementation that were common across all models will be presented here. This includes an overview of the implementation models, activation functions, optimisation techniques, and evaluation methodologies.

**Implementations:** The models implemented were artificial neural network architectures. When building a neural network there are a number of different parameters that must be considered. Most importantly is the internal architecture of the layers themselves. These may be designed so as to focus on solving a particular problem, or on handling a particular type of dataset or feature space. Structures of neural networks implemented were multi-layer perceptrons (MLPs), convolutional neural networks (CNNs), recurrent neural networks (RNNs), and various combinations and modifications of these. CNNs are typically the default model for image analysis with research tracing structural design to that of the visual cortex in animals(**?**). RNNs introduce memory to the network and so are useful for time-series data. CNN-LSTM combines elements of both. Finally, Conv-LSTM is an LSTM layer that applies convolutional operations to its input. Once the defining structure of the layer is chosen, the choice remains of how many layers to include, and how many nodes to include in each layer.

**Activation Functions:** The activation function is responsible for converting the aggregated input to the node into an output. This takes the form of a mathematical function, such as sigmoid or tanh. An unfortunate side-effect of the aforementioned functions is that while adjusting their weights during backpropagation their non-linearities often result in producing a vanishing gradient (Glorot et al., 2011). The rectified linear activation unit, or ReLU for short, has become a popular solution to this problem.

$$f(x) = \begin{cases} 0, & \text{for } x < 0 \\ x, & \text{for } x \geq 0 \end{cases} \quad (\textit{Glorotet al.}, 2011) \tag{1}$$

This means the function is linear for all positive values, and 0 for all negative values. The function has seen much success recently due to its computational simplicity and linear behaviour. For outputting values, the softmax activation function normalises its input into a probability distribution and is often used in the final layer for associating a probability in predicting labels.

**Optimisation:** Gradient descent is the means by which the weights of the nodes in the neural network are updated so as to minimise the output error. The weightings are

updated at the end of each batch process. Deciding how frequently to update these values is adjusted through this batch size parameter. A form of gradient descent that is often used when dealing with very large datasets, is to set this batch size to 1, i.e. have the weightings adjusted after every training instance. This is found to train the model more quickly and requires fewer passes through the dataset to converge on coefficients. Another algorithm, Adam (adaptive moment estimation) (Kingma and Ba, 2014), combines the benefits of two other optimisation algorithms - Adaptive Gradient Algorithm (AdaGrad) and Root Mean Square Propagation (RMSProp). It has become quite popular in recent times and is recommended by Karpathy (2019) as the first choice optimiser for deep learning applications. A final note on batch sizes: when choosing a values for this parameter it is recommended stick with orders of 2 (16, 32, 64, etc.), as these values are shown to require less computational overhead and subsequently results in shorter training times. Evidence supports the use of smaller batch sizes ($\approx 32$), as larger batch sizes tend to converge quickly to sharp minima, hamper the model's ability to generalise. (Keskar et al., 2016)

**Evaluations:** The performance of a classifier is typically measured by its accuracy, such as Hernández (2018). The accuracy is the number of correct classified samples divided by the total number of samples. Formally,

$$Accuracy = \frac{\text{No. of Correct Predictions}}{\text{Total No. of Predictions}} \quad (\textit{Hernandez}, 2018) \tag{2}$$

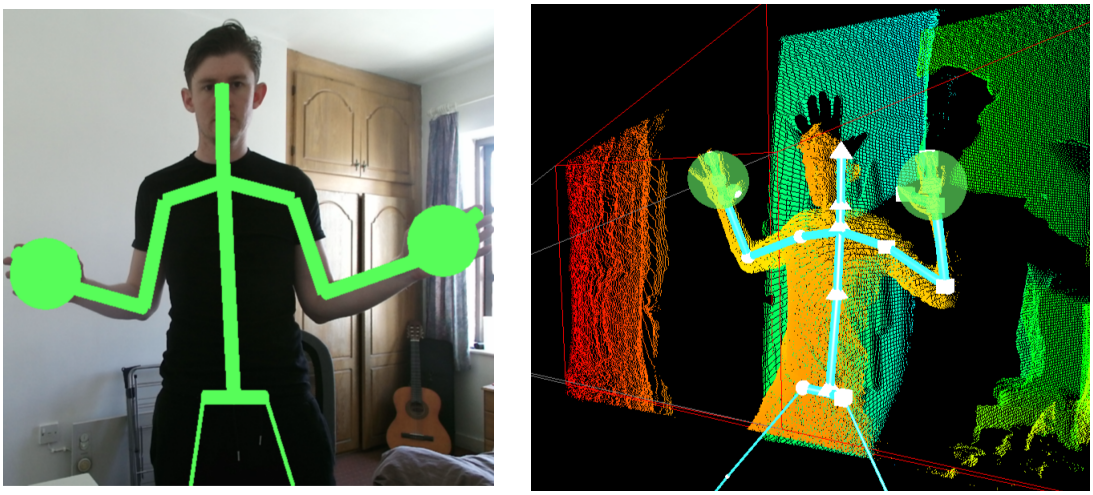
For the multiclass problem presented in this paper, the accuracy will be computed for each class and then averaged. To avoid cherry-picking the best results for each model generated, the model's will were each trained and tested ten times, and the average accuracy (and standard deviation) taken. The results can also be graphed as a heat map of actual classifications vs. predicted classifications. This is known as a confusion matrix. The more values along the diagonal (i.e. actual classes = predicted classes) the more accurate the model's predictions are. However, the format has the added benefit of being able to analyse the predictions on a more granular level. It allows, for instance, to see what two actions the model is confusing for one another. In the case of binary classification, precision and recall scores can be used to give further insight into the model's performance. However, in the case of multiclass prediction, several one-v-all tests would need to be performed. This can be useful up to a point, however, having 10 classes would make this analysis cumbersome and provide limited insights. Also, considering the ultimate intention is to scale this up to be a full language classifier, this metric would not be appropriate. While accuracy is a very interpretable metric of performance, useful when communicating to business or the public, it does not take into consideration the surety of a model in its predictions. For instance, one model could be very certain of an incorrect prediction whereas another would be unsure and incorrect. Viewing accuracy alone, both models appear to perform similarly, although the second model should be valued more highly. A more informative measure then is the categorical cross-entropy loss. This takes into consideration a model's confidence when making a prediction. Additionally, the learning rate of each model was measured. This shows the rate at which the model's accuracy improved with respect to training epochs.

## 4.2 Data Preparation

This project will be utilising an unsupervised dataset. The dataset was created with the express purpose of solving the primary research question.

### 4.2.1 Data Recording

A Microsoft Kinect camera was set up to capture the entire signer in shot. It was essential that all joints were being tracked properly. Microsoft Visual Basic Studio was used to interact with the Kinect camera. Microsoft's intention is that the users record and analyse their data within the Kinect Gesture Builder and as such, they do not natively offer access to the raw tracking data. However, a piece of code from a GitHub repository (Xiaozhuchacha, 2017) was used to extract this raw skeleton tracking data. This took a snapshot of the xyz-coordinates of each joint, thirty times per second. Ten distinct signs were recorded in front of the Kinect camera, while it was capturing. Each sign was recorded for 3 seconds. Each gesture was captured 50 times to give a sufficiently large dataset. This yielded a training dataset of 500 samples, equally balanced across 10 classes.



(a) Kinect Skeleton Tracking

(b) Depth-camera maps the subject in 3D

Figure 3: Monitoring of Kinect v2 RGB-Depth Footage

The Kinect tracking algorithm records the movement of 25 points on the human body Fig, in $xyz$-coordinates. The movements were recorded at 30 frames per second, for 3 seconds. This gives a total of 6750 (25 features $\times$ 3 $xyz$ $\times$ 30 frames $\times$ 2 seconds) values per sample. The Kinect records each frame as a separate .txt file, which were collated into a single dataframe.

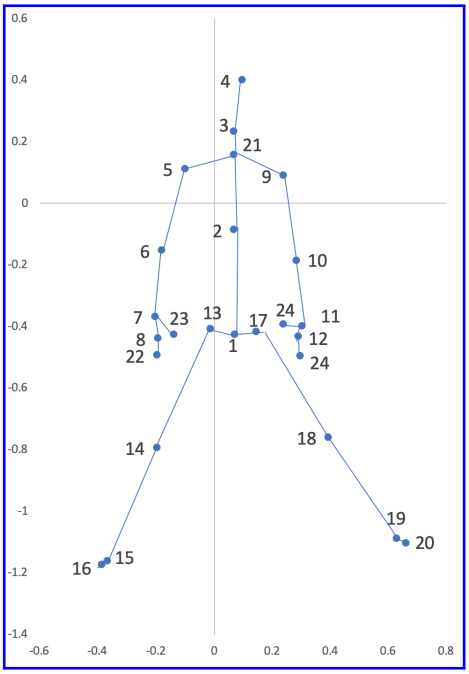
### 4.2.2 Dataset Validation

To ensure no sequences run too long, or too short the dataset was validated to make certain that all samples contained an equal number of timesteps. It is a pre-requisite of Keras that the input tensors share similar dimensionality. This meant trimming the samples to all be the same duration - in this case, 90 time steps were chosen (3 seconds at

30Hz). To do this, Python scripts were executed over the dataset, trimming any samples that ran longer than 90 timesteps, and padding any that ran short.

```
45674779
0 0.0705014 -0.428685 1.10989 2
1 0.0700175 -0.0886462 1.15535 2
2 0.0684063 0.232054 1.17893 2
3 0.098719 0.397844 1.16723 2
4 -0.10012 0.111209 1.16196 2
5 -0.17909 -0.15527 1.16352 1
6 -0.202636 -0.369836 1.0591 1
7 -0.193646 -0.44102 1.02609 1
8 0.239933 0.0875571 1.16111 2
9 0.285205 -0.189171 1.19016 1
10 0.304295 -0.400142 1.07872 1
11 0.291936 -0.433858 1.05824 1
12 -0.0110046 -0.409676 1.07013 2
13 -0.195863 -0.794683 1.01727 1
14 -0.367648 -1.16287 0.997024 1
15 -0.386146 -1.17434 0.862561 1
16 0.147337 -0.420026 1.07804 2
17 0.395179 -0.761959 1.06734 1
18 0.629451 -1.09119 1.08733 1
19 0.660519 -1.1043 0.957608 1
20 0.0689489 0.154578 1.17579 2
21 -0.194604 -0.49548 0.998188 2
22 -0.137115 -0.42831 1.0222 2
23 0.29734 -0.498396 1.02242 2
24 0.240669 -0.395357 1.0366 2
```

(a) Raw txt file (01001001.txt)




(b) Data plotted as scatterplot

Figure 4: Skeleton Tracking Data

The data was then validated visually by means of a scatter plot. Here a random time step was selected, (01001001.txt) and plotted as an XY scatter plot in Excel. Utilising Excel allowed for the creation of connecting lines, the skeleton's limbs, to visualise the signer's form (Figure 4(b)). From here, the data points from the raw text file could be linked to their corresponding skeleton joints.

### 4.2.3 Data Pre-processing

The raw text files had long unintuitive filenames. The first step taken was to rename all raw *.txt* files in sequential order. The filename of each step was based on a composite of the actions being performed, the sample number, and the time-step. For instance: the letter J (01), the first sample (001), and the first timestep (001) was labelled file 01001001.txt - shown in Figure 4(a). The text files are stripped of all superfluous data so that just the 25 joints in xyz-coordinates are extracted. The samples are then separated into training and testing sets (80% training and 20% for testing). This is then combined into a format suitable for processing by TensorFlow - in this case, a 3-dimensional *numpy* array of shape (*samples, features, timesteps*). Regarding the Y dataset, the categorical labels were convertd into 10 binary "on/off" variables. This is knows as *one-hot encoding*.

### 4.2.4 Feature Selection

The text file contains additional information that was omitted for analysis (open/closed hands, the number of skeletons in view, etc.). The only features extracted from the text

file are the 75 tracking values (25 joints in xyz coordinates). In order to identify which data points corresponded to which features, the data from file *01001001.txt* was represented as an XY scatter plot - the third column (Z-axis) was ignored for this purpose. The 25 rows are plotted and labelled in Figure 4 - joining lines were added to aid comprehension. This allows for the selection of only the pertinent features. Since only the upper half of the body is going to contribute to information to the word being signed, it may be a waste of time and resources to include many of the hip and leg joints. This was since any of the feature vectors from the waist down were unlikely to contribute to the signed word, and more likely to incur noise. Feature reduction would reduce the number of independent features being tracked by 24, from 75 to 51.

Dimensionality reduction is a process by which a wide and sparse dataset is reduced to a more manageable number of dimensions while maintaining variance. This is known as the *curse of dimensionality*. 75 dimensions for a dataset of 500 records is not. Any form of dimensionality reduction involves the loss information, so generally, if it can be avoided it should be. The models were trained and tested using all 25 skeleton joints. Fortunately, there was no issue with training time and so all data points were kept for the training. It should be noted that in some cases it is possible for dimensionality reduction to increase model performance, although this is less common.

### 4.2.5 Data Scaling

All features are *xyz* coordinates represented by spacial positions. Since these are confined to the within strict boundaries, standardisation is not relevant. Experiments were performed on normalising the features to fit the values between 0 and 1. However, since the feature values are all of a similar scale there was no noticeable improvement in doing this. Data was split into train and test datasets, with an 80:20 train-test split ratio. This marks the completion of learning objective 2 from Table 1, Section 1.3.

The classifier types considered for this solution can be categorised as either binary or multilabel. Binary classifiers (such as SVM) can be developed into multiclass (OvO or OvA, and can be layered to create multiclass discriminator, not feasible for 10+ words). Also, since the words are signed one at a time, there was no need to implement a multioutput classifier.

## 4.3 Implementation, Evaluation, and Results of Multi-Layer Perceptron Model

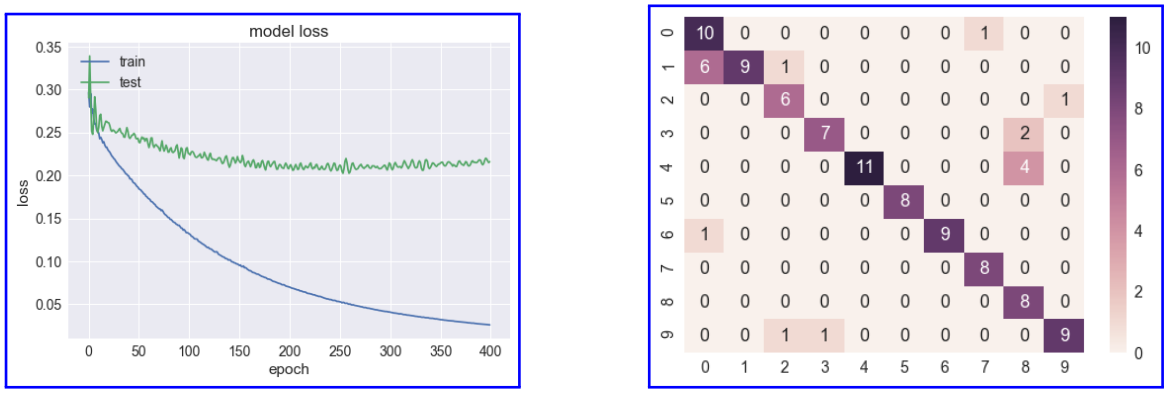
The most rudimentary form of deep learning contains just one hidden layer, between the input and output layers. This layer is fully connected (all cells connected to all cells) and allows it to model non-linear functions. This is known as a multi-layer perceptron.

### 4.3.1 Implementation

A network was created with a single hidden layer. MLP networks are not designed to utilise time-series data, and as such cannot accept input tensors in the form of (*samples, timesteps, features*). To allow time series data a flatten layer was added to reduce the dimensionality.

### 4.3.2 Evaluation and Results

The results show a quick decrease in the loss function over the first few epochs, a gentle decrease over the next 200 or so, followed by a slight increase would suggest that the model was beginning to overfit the training data. The gentle increase meant the model was quite stable and slow to overfit. Averaging accuracy over 10 iterations, an accuracy of 91.359% $\pm$ 5.035. Implementation of this model satisfies Objective 3.1, Table 1, Section 1.3.




(a) MLP Loss Function

(b) Confusion Matrix for MLP Model

Figure 5: MLP Evaluation and Results

## 4.4 Implementation, Evaluation, and Results of Long Short-term Memory Model

An unwanted consequence of feeding a sequence through an RNN network that updates its state after each time-step is that by the time the end of the sequence is fed through, the information gleaned about the initial sequence value is forgotten. This information can be essential, so a special effort needs to be made to retain this memory. The Long Short-Term Memory cell was proposed as a method of retaining valuable long-term information in the data. is a type of recurrent neural network useful for memorising long sequences, and classifying them. It takes in a 3D array of dimensionality [samples, time steps, features]. Each sample here has 90 timesteps and 75 features. The model was designed with only one hidden LSTM layer, and a dropout layer to prevent overfitting. Dropout is a function that assigns a certain probability to whether a neuron is activated during a training step. By continually shifting the neural network structure, the network effectively. Experiments were performed with and without dropout, and although models without sometimes achieved higher accuracies in once off-runs, when averaged out over a number of iterations their scores dropped. This high variance is a tell-tale sign of overfitting.

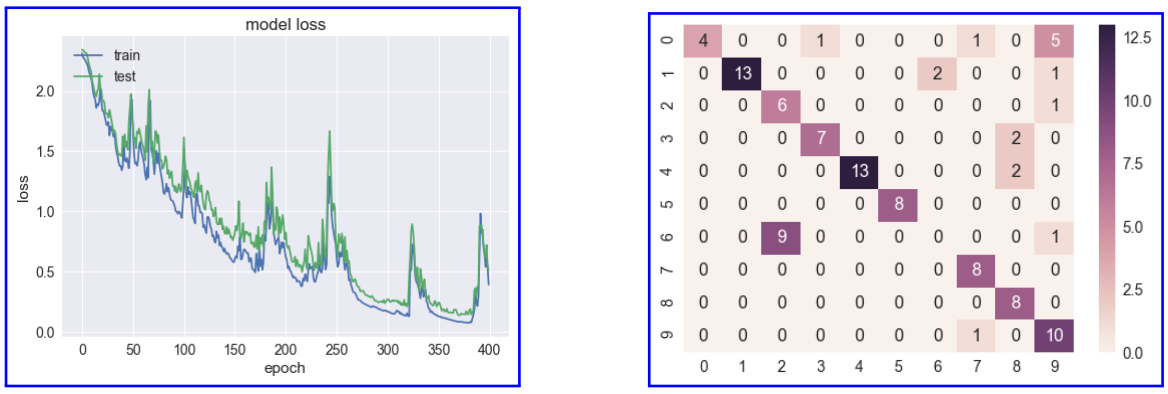
### 4.4.1 Implementation

The model consisted of an LSTM layer, a dropout layer of 0.5, a dense layer of 100, and a dense output layer of outputs (representing the 10 classes). Experiments were run

in batches of 64 samples, over 100 epochs. Each experiment was run 10 times and the average accuracy was taken.

### 4.4.2 Evaluation and Results

The best results were achieved using the Adam optimser and batches of 32 samples. To visualise a model's training improvements with each epoch a loss function is plotted. This compares the change in error in predicted guesses with respect to each epoch that the model undergoes. From Figure 6 (a), a sharp decline can be seen over the first 50 epoch. From here, the train line continues to decrease, however, little improvement is made to the test line. Considering no improvements to the performance are being made, the model was not run for any longer for fear of overfitting. The average accuracy over ten iterations for this model was 90.291%. Looking at the confusion matrix for the LSTM model in Figure 6(b), it can be seen that the main deviations from the diagonal line occur for sign number 6 (Name) it confused with sign number 2 (Z). This is likely due to the swift double movement in both gestures. Also, sign 9 (Thank you) was often mistaken for sign 1 (J).




(a) LSTM Loss Function

(b) Confusion Matrix for LSTM Model

Figure 6: LSTM Evaluation and Results

## 4.5 Implementation, Evaluation, and Results of Stacked Long Short-Term Memory Model

Introduced in 2013, the stacked LSTM model is a type of neural network consisting of multiple hidden LSTM layers. This retains the advantages of an LSTM model while allowing it to learn representation at higher levels of abstraction. Where added convolutional layers can allow computer vision algorithms to learn increasing abstract spatial structures, in the context of time series data this means learning features of varying time scales. (Pascanu et al., 2014)

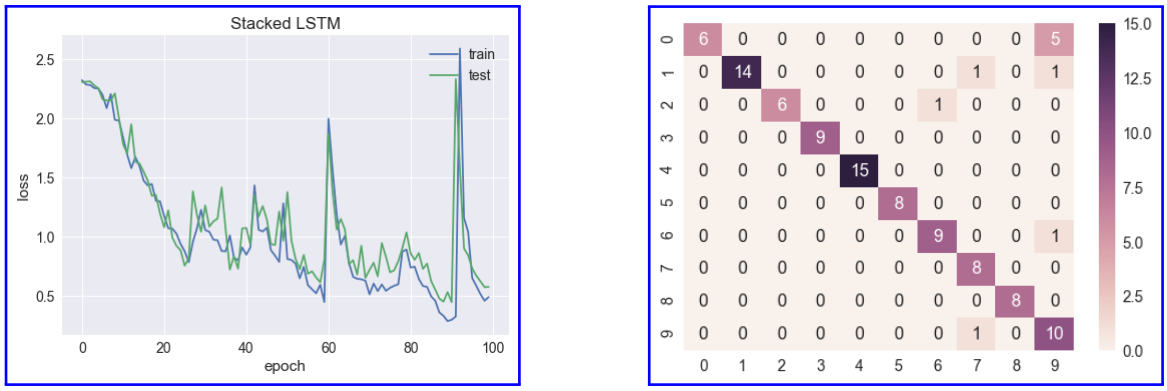
### 4.5.1 Implementation

The best results were obtained by adding in one additional LSTM layer to the model. With each subsequent layer, the performance dropped off significantly, showing strong

indications of overfitting. Numerous experiments with dropout layers were run to combat the overfitting, however, the best model had a single dropout layer at 0.5 was added after the new additional LSTM layer.

### 4.5.2 Evaluation and Results

The model achieved an accuracy of 86.99% with a standard deviation of 7.07%. Analysing the log loss function shows a clear decline for 20 epochs before the decline becomes overwhelmed by noise. Sudden spikes indicate parts in the gradient descent where the optimiser found local minima. Increasingly deep LSTMs would appear to be well-suited for longer form sequential classification problems. For the current dataset, a stacked-LSTM model of two hidden LSTM layers is as much as is required to show improvement. This can be seen in the high variance in the tested models. No arrangement of a stacked-LSTM showed an improvement over a single layer LSTM.




(a) Stacked-LSTM Loss Function

(b) Stacked-LSTM Confusion Matrid

Figure 7: Stacked-LSTM Evaluation and Results

## 4.6 Implementation, Evaluation, and Results of Convolutional Neural Network-LSTM Model

Considering the input feature-space has a spatial component (point-and-joint geospatial co-ordinates) it was prudent to consider incorporating the benefits of convolutional layers. CNN-LSTM architectures were designed to combine the strengths of CNNs for feature extraction with the sequence-learning capabilities of LSTMs (Donahue et al., 2015b). Other research (Sainath et al., 2015) proposing a similar architecture refers to this as a CLDNN architecture.

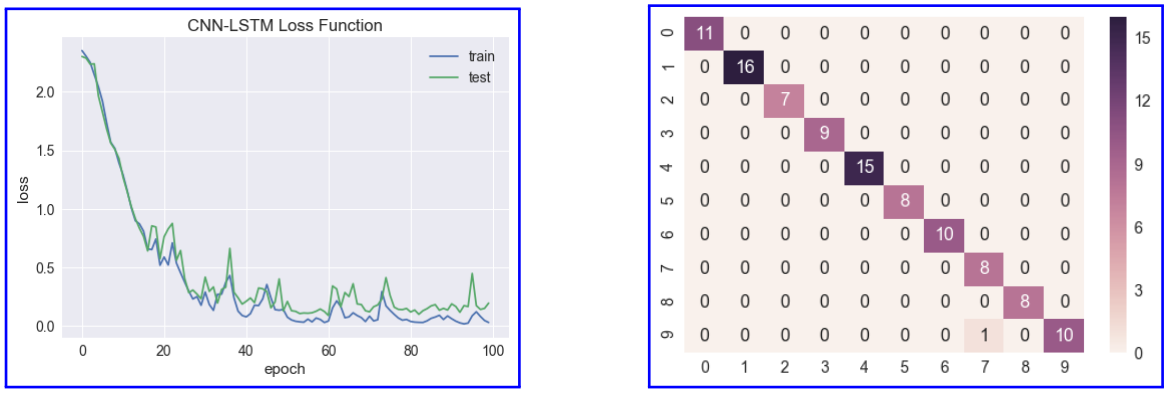
### 4.6.1 Implementation

A CNN-LSTM involves a CNN front-end and an LSTM back-end. The common approach of using two consecutive CNN layers was utilised. The introduction a CNN layer presents two further parameters to adjust: the number of filters, and the kernal size. The best results were achieved by applying 64 $3 \times 3$ filters. A dropout of 0.5 was then applied, and then a max pooling layer. LSTM layer was applied to the output with a dropout

layer. Finally, a dense layer with ReLU activation function was applied to interpret the features and an output layer with softmax activation function. Here the model was run over 100 epochs.

### 4.6.2 Evaluation and Results

This model produced a much improved log loss function (Figure 8(a)). It shows a steadier gradient descent than the two previous LSTM models. and took more epochs of training to find a stable model. The average accuracy of 10 experiments was 95.029%




(a) CNN-LSTM Loss Function

(b) CNN-LSTM Confusion Matrix

Figure 8: CNN-LSTM Evaluation and Results

## 4.7 Implementation, Evaluation, and Results of Convolutional-LSTM Model

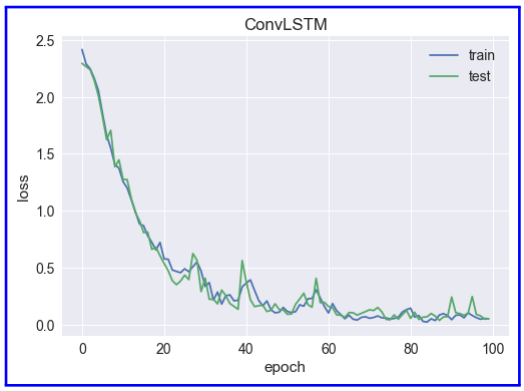
While the combination of CNN and LSTM architectures serves well at tackling the spatio-temporal problems, some correlations may not be fully captured. Employing the convolutional technique of the previous model, but as part of an LSTM layer is known as a Convolutional LSTM. Here, the input and recurrent operations on the LSTM gates are switched from matrix multiplications to convolutions. This model was proposed in 2015 with research showing it to outperform fully connected LSTM models (Shi et al., 2015).

### 4.7.1 Implementation

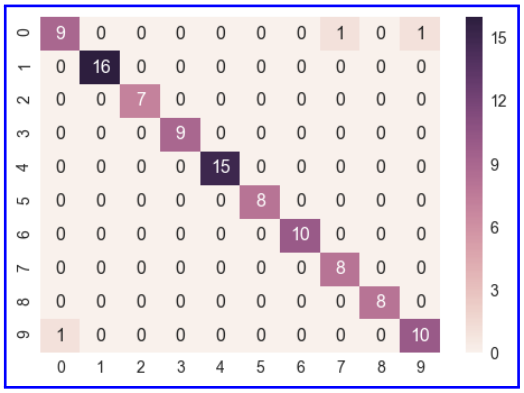
A convolutional-LSTM network consists of a ConvLSTM2D in place of the LSTM layer. This layer involves the configuration of both CNN and LSTM parameters. This means selecting the number of convolutional filters to be applied, as well as the size of the filters. A standard size of 3x3 filter was found to be most efficient.

### 4.7.2 Evaluation and Results

The average accuracy of 10 models, trained over 40 epochs, was 96.408% with a standard deviation of 3.634%. The model achieved high training and validation accuracies quickly and showed a tendency to overfit after 100 epochs.




(a) Conv-LSTM Loss Function

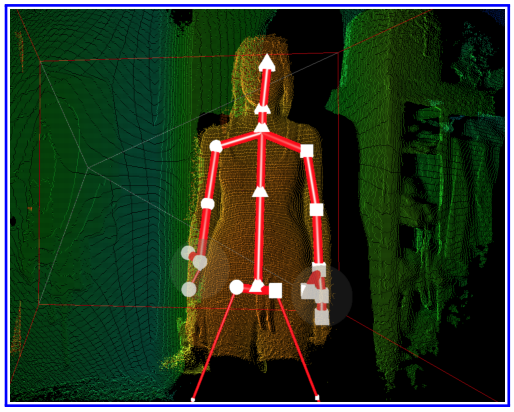



(b) Conv-LSTM Confusion Matrix

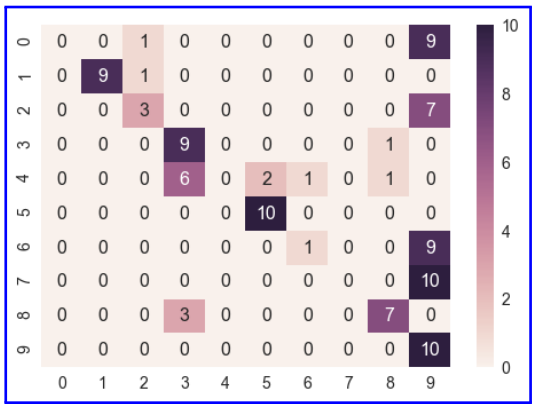
Figure 9: Conv-LSTM Evaluation and Results

## 4.8 Implementation, Evaluation and Results of Signer Independent Testing

Considering the ultimate purpose of this ICT solution would be to be able to classify gestures from previously unseen users, a new dataset was created from the gestures of a new signer. The next objective was to evaluate the performance of the previously trained models at recognising gestures from this unseen signer. Here, an additional testing dataset is introduced and the pre-trained models tested on it. The crucial difference here is that the model was not be trained on this new dataset; purely on the first dataset. This tested the model's robustness and ability to recognise words from unseen signers.



(a) Tracking of Unseen Signer




(b) MLP Confusion Matrix

Figure 10: Signer Independent Testing

### 4.8.1 Implementation

To maintain the same train-test ratio as before (80:20), 400 samples were randomly selected from our first dataset (maintaining balance among the classes), and a new dataset was created with 100 samples. The models were trained on the original dataset and

tested on this new dataset. The testing was done by all developed models, with the MLP achieving the highest results.

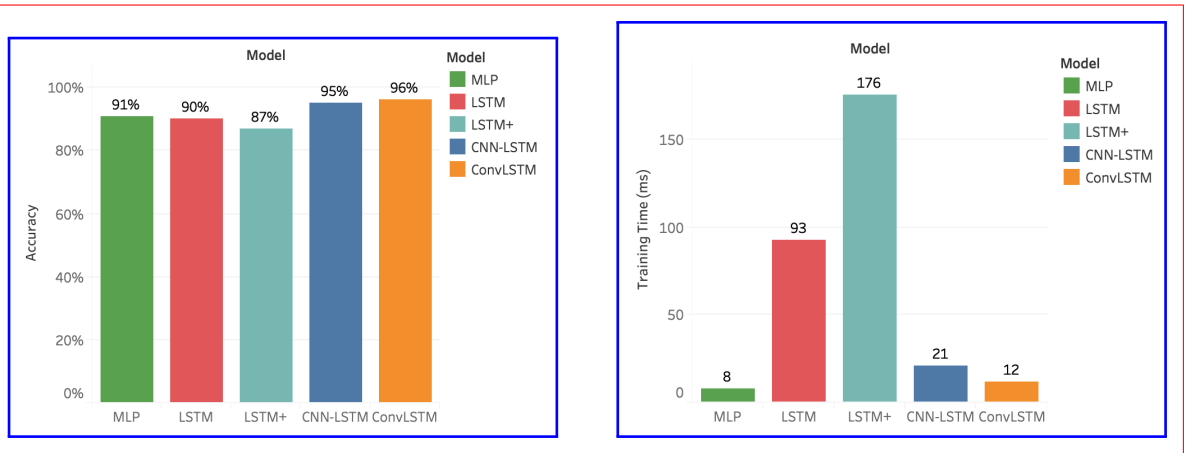
### 4.8.2 Evaluation and Results

When the model was run 10 times it achieved the list of accuracies: 18%, 26%, 35%, 43%, 39%, 10%, 47%, 54%, 32%, and 10%. As can be seen, some models reach respectable accuracy of 47% and 54%, however other models got stuck in local minima and remained at 10% throughout the entire training phase. Results from the LSTM, CNN-LSTM, and ConvLSTM models rarely exceeded 20%. Strictly speaking, this is not necessarily overfitting of the training data, but overfitting to a particular style of movement of the signer. Possibly due in part to its simpler framework, the MLP can generalise better to testing on the unseen signer.

## 4.9 Comparison of Developed Models

The accuracies of the five models analysed in this section (MLP, LSTM, Stacked-LSTM, CNN-LSTM, ConvLSTM) are presented in Figure 11(a). performed very well, with all getting average accuracies of over 90%. The simplest model, the MLP, performed quite well considering its modest architecture (outperforming the LSTM and Stacked-LSTM models). It was only with the addition of the CNN and convolutional-LSTM layers that effectively improved the predictive power.

Another important consideration when choosing a model is the time required to train it. To get a fair comparison of how long each model took to process, they were executed using similar batch sizes and optimisers. All models were trained on batch sizes of 1 and the average of 5 epochs taken. The comparison numbers are the average time per training step - so for one batch to complete one pass through the model, and the model to update the weights. The MLP was understandably the fastest to train, however, the CNN and convolutional layer decreased the standard LSTM training step from 93 ms to 21 ms and 12 ms receptively (Figure 11(b)). Other than the increased accuracy, a clear benefit to applying convolutional operations to the data before the LSTM function is the decreased training time (for a fixed number of nodes per LSTM layer).



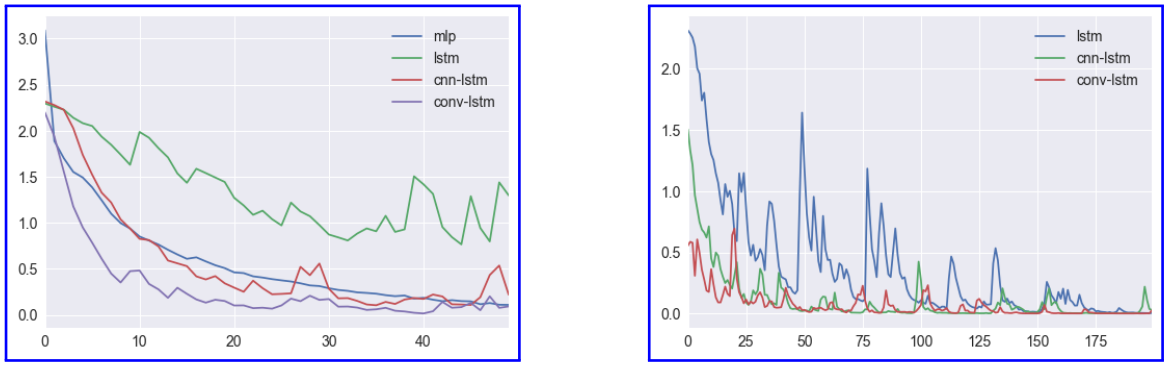

(a) Comparison of Model Accuracies

(b) Comparison of Step Times (ms)

Figure 11: Comparison of Developed Models

### 4.9.1 Performance History

The previous results compared the highest accuracy achieved by each model, and the associated time in training that model. For a complete analysis of a model's performance, it is important to visualise how the performance of the model increases with respect to each training epoch. This can show us which models reach high accuracies the quickest. The MLP can be seen to show the steadiest decline of the four models shown. This is due to its simpler construction. Of the other models, the CNN-LSTM performs marginally better, and the ConvLSTM outperforms the lot reaching a loss of $< 0.5$ after only 6 epochs (compared to 15 epochs for CNN-LSTM and 20 epochs for MLP)




(a) Comparison of Models over 40 epoch

(b) Models trained over 200 epochs

Figure 12: Loss Functions of Trained Models

This comparison completes objective 4 from Table 1 in Section 1.3.

## 4.10 Comparison of Developed Models with Existing Models

The results of the best-in-class model developed, the ConvLSTM, compare favourably to the existing state-of-the-art solutions. While the increased accuracy is a positive contribution in its own right, arguably the most important distinction is the feature set which the analysis is performed on. Most other classifiers operated on static images, and the other comparable motion-based research in the area either utilised special gloves (Kelly et al., 2009), or video analysis (Hernández, 2018) to classify gestures. Table 3 below marks the completion of objective 4 (Table 1, Section 1.2).

Table 3: Comparison with other ISL-Classification Research Projects

| Features | Classifiers | Accuracy | Author |
|---|---|---|---|
| Images (Alphabet) | SVM | 97.3% | Kelly (2010) |
| Videos (Gestures) | GTHMM | 97.7% | Kelly et al. (2009) |
| Images (Alphabet) | PCA & SVM | 99.875% | Oliveira et al. (2017) |
| Images (Alphabet) | CNN | 99.98% | Hernández (2018) |
| Videos (Gestures) | 3D CNN & LSTM | 79.66% | Hernández (2018) |
| Tracking data (Gestures) | ConvLSTM | 96.408% | Cian Vaughan (2019) |

### 4.11 Conclusion

The implementation of the models (MLP, LSTM, Stacked-LSTM, CNN-LSTM, ConvLSTM) satisfy of this satisfies Objectives 3.1 - 3.6, Table 1, Section 1.3. The MLP model proved to be quite successful and crucially more stable than early iterations of the LSTM architectures. It was not until the LSTM model was combined with CNN layers and utilising convolutional filtering that the models began to outperform the default MLP model. The most noticeable improvement was bringing stability to the shape of of the loss function. Although the not much separates the models at this stage, intuition would suggest that as the dataset grows, and the variety of different signers increases, the MLP network would struggle to learn more complex seqeunces and as such the deeper and more complex neural networks would begin to prove the more capable models.

## 5 Discussion

Undertaking a project that combined real-world actions, a motion-sensing input device, raw sensor data, and machine learning models was quite a challenge. Most challenging was perhaps the data preparation. Extracting the raw data from hardware that was built to be integrated with proprietary software. Then, preparing it to be presentable in a format suitable for analysis. Finally, each machine learning model was expecting a different data shape, so it has to be reshaped for each neural network architecture.

However, such a broad-ranging project resulted in learning a wide-ranging skill-set and a host of new technologies. The clearest learning for me was the huge gains made from the reduction of human actions to point-and-joint co-ordinates. While video analysis is typically reserved for the domain of supercomputers, motion-tracking could now be performed by consumer-grade computing power in a matter of minutes.

Developing a model to recognise Irish Sign language could deliver potentially life-changing impact to the affected communities, and with Microsoft recently announcing the upcoming release of the Kinect 3 (Warren, 2019) the market is primed for innovative applications to the new technology.

## 6 Conclusion and Future Work

The main research and sub research question proposed in the introduction have been fully solved, and all objectives completed. The review of the literature indicated that there is a gap for an ISL classifier. In the creation of this ICT solution, a dataset of ISL words was created using a depth-camera and skeleton tracking technology. This has been presented as a contribution to the body of knowledge. A comparison of five different neural network architectures showed a general trend towards increasing accuracy over fewer training epoch, however, it came at a cost of potentially overfitting the data. A surprising finding was the success of the most basic neural network. Although requiring slightly more epochs to train, the MLP network performed almost as well as the more complex architectures. This was likely due to its being more generalisable, in particular when it came to testing on the unseen dataset. This may change as the dataset is expanded to include multiple different signers and more complex gestures. When tested on an unseen signer the accuracy scores were much lower, but promising. Indications suggest that further training on multiple different signers would build a more robust

classifier capable of identifying gestures from unseen signers. This second dataset is also submitted as a minor contribution. The ISL-classifier methodology was designed and implemented as a minor contribution. Other minor contributions were a comparison between the developed models, and a comparison between these models and state-of-the-art models in the field.

**Future Work:** Improvements to the models can be done by expanding the dataset to include more gestures and, crucially, training it on different signers to improve robustness. The Kinect also contains a standard high-definition RGB-camera, which could combine image analysis with the gesture classifiers developed to complete the vocabulary. It could be interesting to take advantage of Kinect's other algorithms. In particular, the facial expression recogniser would contribute hugely considering the importance of facial expression in the communication of sign language.

# 7 Acknowledgement

I would specifically like to thank to Dr. Catherine Mulwa for her guidance and encouragement through out the project; to the deaf community of Ireland who assisted and advised me; to my mother and father for their continued support; and to my girlfriend Martika for being the unseen signer.

# References

Ahmed, M., Idrees, M., Abideen, Z. U., Mumtaz, R., and Khalique, S. (2016). Deaf talk using 3d animated sign language: A sign language interpreter using microsofts kinect v2. *2016 SAI Computing Conference (SAI)*.

Alp, E. C. and Keles, H. (2019). *A Comparative Study of HMMs and LSTMs on Action Classification with Limited Training Data: Proceedings of the 2018 Intelligent Systems Conference (IntelliSys) Volume 1*, pages 1102–1115. test.

Cate, H., Dalvi, F., and Hussain, Z. (2015). Sign language recognition using temporal classification. *CoRR*, abs/1701.01875.

Darwish, S. M., Madbouly, M. M., and Khorsheed, M. B. (2016). Hand gesture recognition for sign language: A new higher order fuzzy hmm approachhand gesture recognition for sign language: A new higher order fuzzy hmm approach. *International Journal of Engineering and Technology*, 8(3):157–164.

Donahue, J., Hendricks, L. A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Darrell, T., and Saenko, K. (2015a). Long-term recurrent convolutional networks for visual recognition and description. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Donahue, J., Hendricks, L. A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Darrell, T., and Saenko, K. (2015b). Long-term recurrent convolutional networks for visual recognition and description. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Dong, C., Leu, M. C., and Yin, Z. (2015). American sign language alphabet recognition using microsoft kinect. *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*.

Estiri, H. and Murphy, S. (2018). Semi-supervised encoding for outlier detection in clinical observation data. *bioRxiv*.

Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. In Gordon, G., Dunson, D., and Dudík, M., editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 315–323, Fort Lauderdale, FL, USA. PMLR.

Hernández, I. (2018). Automatic irish sign language recognition. Master's thesis, University of Dublin, Trinity College.

Huang, J., Zhou, W., Li, H., and Li, W. (2015). Sign language recognition using 3d convolutional neural networks. *2015 IEEE International Conference on Multimedia and Expo (ICME)*.

Karpathy, A. (2019). http://cs231n.github.io/neural-networks-3/.

Kelly, D. (2010). *Computational Models for the Automatic Learning and Recognition of Irish Sign Language*. PhD thesis, National University of Ireland, Maynooth.

Kelly, D., Donald, J. M., and Markham, C. (2009). Evaluation of threshold model hmms and conditional random fields for recognition of spatiotemporal gestures in sign language. *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*.

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. (2016). On large-batch training for deep learning: Generalization gap and sharp minima. *CoRR*, abs/1609.04836.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization.

Leeson, L. and Saeed, J. I. (2012). *Irish sign language: a cognitive linguistic account*. Edinburgh University Press.

LeMaster, B. (1998). *Sex differences in Irish Sign Language*, page 67–86. Mouton De Gruyter.

Liu, T., Zhou, W., and Li, H. (2016). Sign language recognition with long short-term memory. *2016 IEEE International Conference on Image Processing (ICIP)*.

Muhammad, A., Addenan, M., Latiff, M., Haris, B., Surip, S., and Mohamed, A. S. A. (2016). Interactive sign language interpreter using skeleton tracking. *test*, 8:137–140.

Oliveira, M., Chatbri, H., Little, S., Ferstl, Y., Oconnor, N. E., and Sutherland, A. (2017). Irish sign language recognition using principal component analysis and convolutional neural networks. *2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*.

Pahlevanzadeh, M., Vafadoost, M., and Shahnazi, M. (2007). Sign language recognition. *2007 9th International Symposium on Signal Processing and Its Applications*.

Panzner, M. and Cimiano, P. (2016). Comparing hidden markov models and long short term memory neural networks for learning action representations. *Lecture Notes in Computer Science Machine Learning, Optimization, and Big Data*, page 94–105.

Parcheta, Z. and Hinarejos, C. D. M. (2018). Sign language gesture classification using neural networks. *IberSPEECH 2018*.

Pascanu, R., Gulcehre, C., Cho, K., and Bengio, Y. (2014). How to construct deep recurrent neural networks. In *Proceedings of the Second International Conference on Learning Representations (ICLR 2014)*.

Potkin, O. and Philippovich, A. (2018). Static gestures classification using convolutional neural networks on the example of the russian sign language. *AIST*.

Rahagiyanto, A., Basuki, A., Sigit, R., Anwar, A., and Zikky, M. (2017). Hand gesture classification for sign language using artificial neural network. *2017 21st International Computer Science and Engineering Conference (ICSEC)*.

Sainath, T. N., Vinyals, O., Senior, A., and Sak, H. (2015). Convolutional, long short-term memory, fully connected deep neural networks. *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Shi, X., Chen, Z., Wang, H., Yeung, D., Wong, W., and Woo, W. (2015). Convolutional LSTM network: A machine learning approach for precipitation nowcasting. *CoRR*, abs/1506.04214.

Sigit, R., Setiawardhana, and Kartika, D. R. (2016). 3d sign language translator using optical flow. *2016 International Electronics Symposium (IES)*.

Stein, D., Dreuw, P., and Ney, H. (2017). Hand in hand: Automatic sign language to speech translation. *The 11th Conference on Theoretical and Methodological Issues in Machine Translation*, 5:30–43.

Stokoe, W. C. (1965). *A dictionary of American sign language on linguistic principles*. Gallaudet College Press.

Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., and Paluri, M. (2017). A closer look at spatiotemporal convolutions for action recognition. *CoRR*, abs/1711.11248.

Warren, T. (2019). Microsoft shrinks kinect into a 399 cloud-powered pc peripheral.

Xiaozhuchacha (2017). xiaozhuchacha/kinect2toolbox.

Yang, S. and Zhu, Q. (2017). Continuous chinese sign language recognition with cnn-lstm. *Ninth International Conference on Digital Image Processing (ICDIP 2017)*.

Zaki, M. M. and Shaheen, S. I. (2011). Sign language recognition using a combination of new vision based features. *Pattern Recognition Letters*, 32(4):572–577.

Zhao, R., Ali, H., and van der Smagt, P. (2017). Two-stream RNN/CNN for action recognition in 3d videos. *CoRR*, abs/1703.09783.