Gesture Sequence Recognition using Inertial Motion Units(IMU)

A Project Report

submitted by

K. DILIP CHAKRAVARTHY

in partial fulfilment of requirements
for the award of the dual degree of

BACHELOR OF TECHNOLOGY AND MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS

June 2017

THESIS CERTIFICATE

This is to certify that the thesis titled **Gesture Sequence Recognition using IMUs**, submitted by **K. Dilip Chakravarthy**, to the Indian Institute of Technology, Madras, for the award of the degree of **Bachelor of Technology and Master of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. Kaushik Mitra
Research Guide
Assistant Professor
Dept. of Electrical Engineering
IIT-Madras, 600 036

Place: Chennai

Date: 10th June 2017

ACKNOWLEDGEMENTS

I would like to thank all of them that have been involved and helped me throughout my project. I am extremely grateful to my research advisor Prof. Kaushik Mitra who supported me extensively through the whole project and encouraged me towards its successful completion.

I thank Ghulam Ahmed Ansari who was there to discuss the problems as well as gave insightful suggestions during the project. I would also like to thank him for the support he extended for working on the tensorflow environment.

I would also like to thank Renju PB who has been my partner in visualizing the final product. Discussions with him led me to understand the final product to be achieved which helped me proceed in the right direction.

Finally, I thank my parents, friends and all the people that have lent me support during my 5 years of stay at IIT Madras.

ABSTRACT

KEYWORDS: Gesture Recognition ; Subgestures; Long Short Term Memory; Encoder Decoder Models.

Almost all approaches nowadays are focussed on classifying single gestures in each attempt. While they are decently successful, categorising single gestures doesn't prove to be very useful in applications that are prevalent today. In this paper, we propose a Long Short Term Memory(LSTM) based deep network on the lines of an Encoder-Decoder architecture that classifies gesture sequence accurately in one go. We also show an empirical training strategy for our architecture which can achieve good results even with limited amount of collected data. Results from the experiments performed on labelled datasets from Inertial Motion Units (IMU) proves the efficiency and usefulness of the proposed method.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

IITM	Indian Institute of Technology, Madras
RTFM	Read the Fine Manual

NOTATION

r	Radius, m
α	Angle of thesis in degrees
β	Flight path in degrees

CHAPTER 1

INTRODUCTION

In today's world we see every computing device getting seamlessly integrated in our lives day by day for which it becomes critical to have intuitive and coherent Human Computer Interactive(HCI) devices. Innovative gesture recognition algorithms play an important part in this journey through which we can develop applications like sign language recognition, touchless car assistance system, new age gaming systems, etc.

1.1 Motivation

The most popular method of gesture recognition till now was through video sequences Mitra and Acharya (2007) VI. Pavlovic (1997). However, there have then been a lot of other methods are becoming popular like IMUs , depth sensors, etc J. Galka (2016*a*) R. Aggarwal (2015). All the instruments used to gather data for the task of gesture recognition have their own challenges. In case of videos, several systems that are in place only work in good lighting conditions and such systems fail very badly due to occlusion J. Galka (2016*a*). Depth based sensors also have an issue of occlusion and the depth range. While, IMUs don't have any of the above problems, it faced the inconveniences like wiring and bulkiness till recent times.

Our work mainly deals with gesture recognition based on IMU pertaining to several advantages it posesses. Firstly, usage of IMU removes the problem of occlusion and thus we can infer data in real time without any issues. Secondly, over time, there have been lot of improvements in terms of sizing and accuracy of the IMUs being produced J. Galka (2016*a*). Also, advances in latest technologies like Bluetooth Low Energy makes it more convenient to use IMUs in recent times.

1.2 Desired Outcome

Many recognition algorithms that exist have the capability to be able to classify a single gesture and depend a lot on stop detection algorithms and gesture segmenting algorithms which doesn't help the system to be fluent. Being able to recognise multiple temporal gestures at once increases the number of functionalities of the system thus making the system smooth. For example, with a gesture database of 8 different types of gestures and a system which can recognise multiple temporal gestures upto length 3 accurately, we will ideally be able to recognise 8+8*8+8*8*8=584 gesture/functionalities. This kind of recognition systems will prove to be beneficial in systems like sign language recognition to recognise different standard phrases or in touchless car assistance systems to control the different features, etc.

In this paper we introduce a new variant of Encoder-Decoder Models(EDM) I. Sutskever (2014) in the field of gesture recognition for the first time to tackle the problem of accurate classification of multiple temporal gestures. EDMs have become very famous recently for their success in the field of natural language processing, specifically in machine translation, and other sequence to sequence problems such as video captioning P. Pan (2015).

In particular, we propose a new Selective Encoder Decoder(SED) network that has the ability to recognise multiple temporal gestures at once along with experimental results on a dataset collected from an IMU. We also show an emperical training strategy used by us and which allows the network to learn parameters with minimal training data and also demonstrate its usefulness through experimental results.

1.3 Previous Works

Most of the approaches towards gesture recognition were in the field of vision. Particularly, there was a good amount of success in the usage of HMMs. For instance, Elmezian et al. Elmezain and Al-Hamadi (2008) have proposed a Left-Right Banded HMM model for classification of the gestures.Yang et al. Z. Yang (2014) have worked on the problem using HMMs on fragmented observations. There have also been some attempts using sparse Bayesian classifier for example Wong et al. Wong and Cipolla

(2007). Recently, several successful deep learning based solutions have come up like Molchanov et al. P. Molchanov (2015) have proposed a 3D Convolutional Neural Network and achieved a reasonable accuracy on the VIVA dataset. There have been other works in the field of gesture recognition, action and emotion recognition which try to use the concept of recognizing gestures based on the recognizing the subelements Yang and Narayanan (2015) J. Galka (2016*a*) J. Galka (2016*b*) Yin (2014). Most of the above attempts being based on a recognition structure called Parallel Hidden Markov Model CD. Mitchell (1995).

Sequence to sequence models in Deep Learning is one of the newest architecture that was introduced few years ago. It has showed tremendous promise in the fields of Machine translation. Cho et al. K. Cho (2014) have first introduced the Encoder Decoder models for learning the Phrase representations. There have been many other variations of the model D. Bahdanau (2015) I. Sutskever (2014) and we derived our inspiration to apply this model for this problem.

CHAPTER 2

BACKGROUND THEORY

2.1 Overview

This chapter gives a brief information about all the concepts that have bee used for the work that has been done.

2.2 Deep Learning

Deep Learning is an a separate class of machine learning algorithms where the basic objective is to find an approximation to a complex(or simple) function which can be used in a variety of tasks like recognition, prediction, feature selection, etc. It usually involves a manually decided loss function over the parameters of the cascaded network which is programmed to be minimized using various methods like gradient descent, adagrad, adam, etc.

Deep Learning has several configurations and advantages to it's usage which has made it very effective over a wide range of problems and has also become very popular in recent times. For example, for the basic unit(neuron), it could be the normal algebraic unit, convolutional unit, recurrent unit, etc. The excitation of the neuron could be a Relu, sigmoid, etc. Among the advantages it has, they can be joined easily to make complex networks which have the ability to extract informative features and the recent upsurge in the computing power makes it easy to train the networks faster over a large amount of data.

Various approaches of Deep Learning on the problem of Gesture Recognition have been showing tremendous improvements lately. Our work derives inspiration from this field and we propose a Deep Neural Network that can not only accurately classify simple gestures but also classify multiple temporal subgestures at once.

*** Mathematical Part of Deep Learning ***

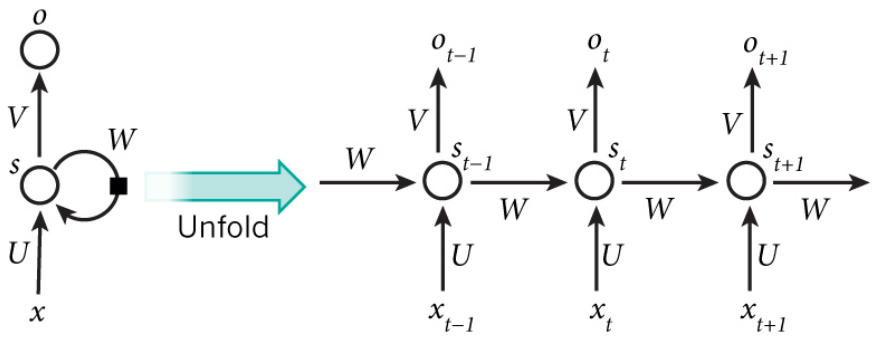

Figure 2.1: A pictorial representation of the Recurrent Neural Network which helps to understand the nature of computations

In this work we propose the usage of Recurrent Neural Networks which is a type of Deep Network which has the ability to deal with sequential data.

2.3 Recurrent Neural Networks

Recurrent Neural Network(RNN) is one kind of neural network which has the capability to operate on sequential data. In each pass of the forward propagation of the network, the state of the network gets updated on the basis of the present input as well as the previous state. This way, it has some kind of memory associated with it and can conveniently act on temporal data and provide effective results in the tasks of classification, prediction, etc.

(Elman's)RNN could mathematically be expressed as follows,

$$h_t = \sigma_h(W_h x_t + U_h h_{t-1} + b_h) \tag{2.1}$$

$$y_t = \sigma_y(W_y h_t + b_y) \tag{2.2}$$

where,

x_t : input vector

h_t : hidden layer vector

y_y : output layer vector

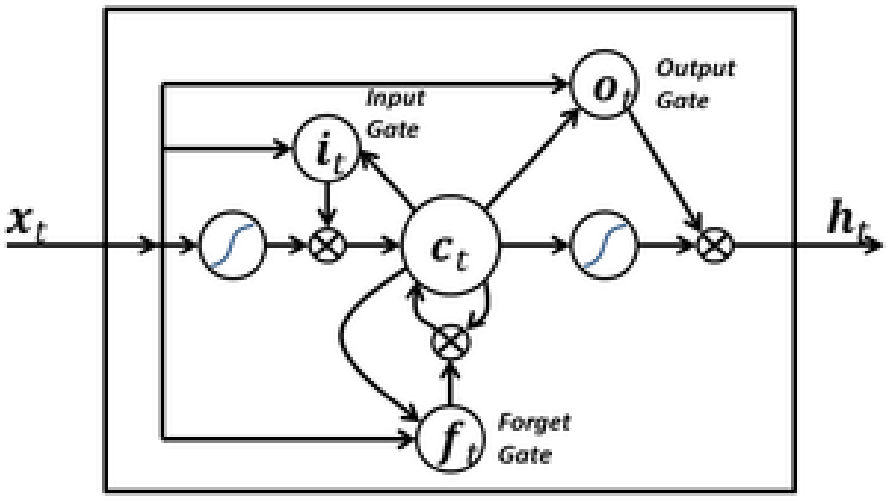

Figure 2.2: Thes standard LSTM cell

W, U, b : parameter matrices and vector

σ_h and σ_y : Activation functions

2.3.1 Long Short Term Memory RNN

Long Short Term Memory Recurrent Neural Network(LSTM) is an upgraded version of the basic RNN which basically has more functionality than a normal RNN which makes it more versatile it it's functionality as compared to the RNN.

The additional inclusion of forget, input and output gates when compared to the standard RNN improves the functionality of an LSTM. The ability to remember, forget and selectively values from the previous state that makes it more efficient in predicting the output in the present state based on the right information.

Mathematically, traditional LSTM can be expressed as follows,

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f) \tag{2.3}$$

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \tag{2.4}$$

$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o) \tag{2.5}$$

$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \tag{2.6}$$

$$h_t = o_t \circ \sigma_h(c_t) \tag{2.7}$$

where,

x_t : input vector

h_t : output vector

c_t : cell state vector

W, U and b : parameter matrices and vector f_t, i_t and o_t : gate vectors (Forget gate vector : Weight of remembering old information, Input gate vector : Weight of acquiring new information, Output gate vector : Output candidate)

σ_g : Sigmoid function

σ_c : Hyperbolic tangent

σ_h : Hyperbolic tangent

$\circ$: Hadamard product i.e entry wise product

2.3.2 Bidirectional LSTM

Bidirectional LSTM is an extended version of the LSTM where it has a set of 2 state vectors when compared to the normal LSTM which has only one. Here, one of the state vector gets updated with respect to the sequence in the forward direction while the other gets update to the same sequence that is inverted. At a time instant, both the state vectors help in predicting the output.

It usually outperforms LSTMs in respect to particular applications where there is a bit of confusion involved between multiple classes that are to be classified. Bi-directional LSTMs are incredibly useful when the context of the input is required to make the required predictions. It helps in the case that reading the sequence from both the sides can help build a better classifier. For example, in handwriting recognition, the performance can be improved by knowing the letters before and after the current letter.

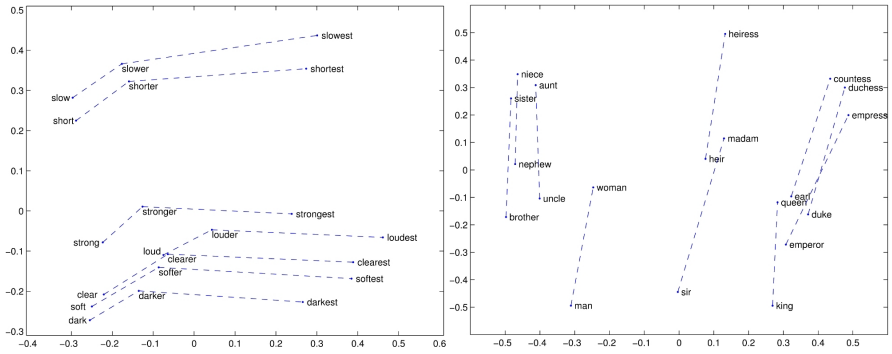

Figure 2.3: Example for pictorial representation of the embeddings on the two principal axes

2.4 Word Embedding

Word Embedding is a method for language modelling and feature learning in Natural Language Processing where words and phrases are collectively mapped to a vectors of real numbers. Conceptually it involves a mathematical embedding from a space of one dimension per word to a continuous vector space with a much lower dimension.

Such embeddings are important because they capture the similarity between various entities. Instead of having a one dimension per word, having a continuous real number vector space for various words enables us to capture the similarities between words using various similarity measurements like cosine similarity, etc.

Such similarities are easy to see when the entities are projected on a vector space map. To show an example for the similarities that is being talked about, consider the words New Delhi, Beijing, India and China. Here, we can say that New Delhi-Beijing are closely related in one way and New Delhi-India are closely related in another way, but New-Delhi is not so closely related with China as compared to other two. This kind of relations can be captured on the embeddings.

2.5 Encoder-Decoder Networks

These networks are usually deep neural networks consisting of 2 parts - encoder and the decoder. The usefulness of these networks come as suggested in the name itself - i.e when it has to encode some information in one domain and decode the same information in another domain. Encoder-Decoder networks have found a great application in a

certain type of problems called sequence to sequence problems. As the name suggests, the aim of the problem is to convert a sequence of information in one domain to that of a sequence in another domain.

Mathematically, they can be represented like this

Encoder :-

$$h_t = RNN(h_{t-1}, x_t) \tag{2.8}$$

$$s_0 = h_T \tag{2.9}$$

where T is the length of the input

Decoder :-

$$s_t = RNN(s_{t-1}, e(\hat{y}_{t-1})) \tag{2.10}$$

$$P(y_t|y_1^{t-1}, x) = softmax(Vs_t + b) \tag{2.11}$$

Parameters : U_{dec}, V, W_{dec}, U_{enc}, W_{enc}

Loss : $L(\theta)$ = $\sum_{i=1}^{T} L_t(\theta)$ = - $\sum_{i=1}^{T} log\ P(y_t = l_t|y_1^{t-1}, x)$

Algorithm : Gradient descent with backpropogation

The basic sequence to sequence model consists of two RNNs, an encoder that processes the input and the decoder that generates the output. In the basic model depicted, the sequence to be encoded is to be read and converted into a fixed-size state vector which is usually the state vector of the encoder RNN. Later, the encoded state is copied into that of the decoder and the sequence that is expected is generated.

Among one of the application that these networks have shown a great promise is in the field of Machine Translation. So far, among all the methods that exist, variants of the encoder decoder models have shown the most promising results. For example, the latest

Type	Sentence
Our model	Ulrich UNK , membre du conseil d' administration du constructeur automobile Audi , affirme qu' il s' agit d' une pratique courante depuis des années pour que les téléphones portables puissent être collectés avant les réunions du conseil d' administration afin qu' ils ne soient pas utilisés comme appareils d' écoute à distance .
Truth	Ulrich Hackenberg , membre du conseil d' administration du constructeur automobile Audi , déclare que la collecte des téléphones portables avant les réunions du conseil , afin qu' ils ne puissent pas être utilisés comme appareils d' écoute à distance , est une pratique courante depuis des années .
Our model	" Les téléphones cellulaires , qui sont vraiment une question , non seulement parce qu' ils pourraient potentiellement causer des interférences avec les appareils de navigation , mais nous savons , selon la FCC , qu' ils pourraient interférer avec les tours de téléphone cellulaire lorsqu' ils sont dans l' air " , dit UNK .
Truth	" Les téléphones portables sont véritablement un problème , non seulement parce qu' ils pourraient éventuellement créer des interférences avec les instruments de navigation , mais parce que nous savons , d' après la FCC , qu' ils pourraient perturber les antennes-relais de téléphonie mobile s' ils sont utilisés à bord " , a déclaré Rosenker .
Our model	Avec la crémation , il y a un " sentiment de violence contre le corps d' un être cher " , qui sera " réduit à une pile de cendres " en très peu de temps au lieu d' un processus de décomposition " qui accompagnera les étapes du deuil " .
Truth	Il y a , avec la crémation , " une violence faite au corps aimé " , qui va être " réduit à un tas de cendres " en très peu de temps , et non après un processus de décomposition , qui " accompagnerait les phases du deuil " .

Figure 2.4: Results from the famous Sutskever et al. paper on Machine Translation

work done has recorded a BLEU score of 36.5 for the translation task of English-French on the whole of WMT-14 dataset.

CHAPTER 3

Selective Encoder Decoder(SED) Architecture

In this section we give the description of the problem to be solved, the dataset and the Selective Encoder Decoder Architecture used to solve it.

3.1 Description of the problem

The aim of the work is to capture the gestures performed by the hand, break them down into a standard set of subgestures and then classify them to a certain gesture according to the temporal sequence of subgestures. For this, we used the data captured from an IMU(Inertial Motion Unit) tied to the finger of a hand, which has the capability to provide close to accurate information about the acceleration and orientation of it in the real world(total of 6 dimensions).

3.1.1 Advantages of this method

The primary advantage that this kind of classification gives is that it increases the scope of number of gestures that can be recognized. For example, suppose we consider 8 number of standard sub gestures in our solution and gesture recognition based on segregation into subgestures upto length 3, we can see that if a proper robust solution can be designed for this, then we can accommodate 8 + 8*8+ 8*8*8 = 584 different gestures using one single classifier.

Accordingly, such a classifier can has applications in various domains where it is required to have a large number of gestures that can be classified such as Sign Language Recognition, Human Computer Interaction, Car feature control, etc. For example, in the case of sign language recognition, it could be developed into a proper device which can recognize

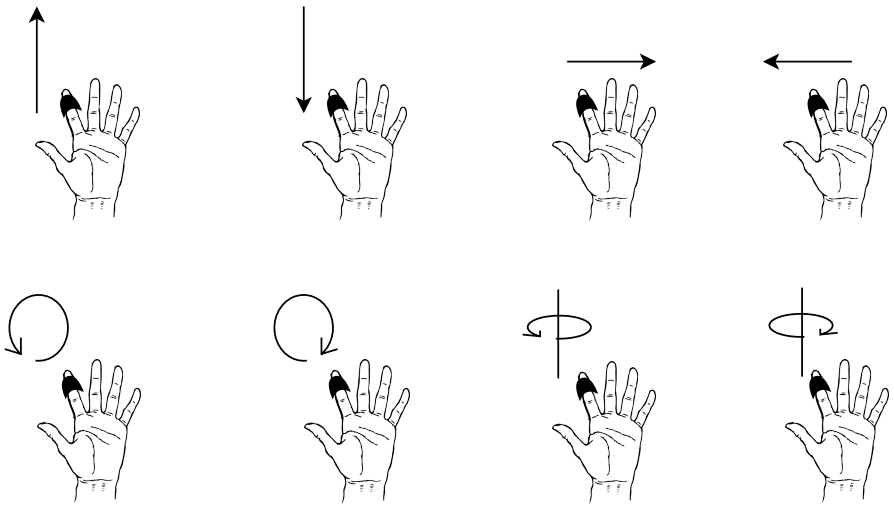

Figure 3.1: Pictorial description of the gestures that have been considered in this paper. The black highlightening on the index finger indicates the position of the IMU that has been placed.

3.2 Formulation

Let **x** = (x_1, x_2, x_3 .... x_n),where each x_i is a vector of dimension D_1 be a sequence of data in one domain named A. The objective is to convert the same sequence from domain A into a sequence **y** = (y_1, y_2, y_3 .... y_m) in another domain B where each y_i is a vector of dimension D_2. The aim of the experiment being to maximize the accuracy of the sequence in domain B.

In this case, the problem we have at hand is to convert a sequence of temporal data collected from the IMU which will be the domain A into that of sequence of subgestures which is the domain B. For example, if swipe down and swipe right were a part of the subgestures domain, then suppose a person drew an 'L' in the air, then the model should be able to predict the gesture as [swipe down, swipe right].

3.3 Dataset

We have generated our own dataset for this. We used the BNO055 Bosch IMU to capture the data from the moving hand. The IMU has a triaxial accelerometer and a triaxial gyroscope along with a triaxial geomagnetic sensor, which is capable of giving the real time IMU frame acceleration and orientation of the IMU in the form of Euler

angles. As a part of preprocessing, we converted the IMU frame acceleration into world frame acceleration using Euler geometry as the following:

$$a_I = R_B^I * a_m + \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \tag{3.1}$$

Where a_I is the Inertial world frame acceleration of the sensor, R_B^I is the rotation matrix from the body frame to the inertial frame of the sensor and a_m is the 3-axis acceleration vector from the accelerometer. Thus, at every time instant we get a 6 dimensional feature vector which contains a 3 dimensional world frame acceleration and 3 dimensional orientation.

Our gesture database consists of the following basic gestures : swipe right, swipe up, swipe left, swipe down, clockwise circle, anti-clockwise circle, clockwise twist and anti clockwise twist. Each of the gesture was captured about 200 times in different different settings with various people making it a total of 1600 gestures.

*** Specs of the IMU *** Frequency, max length of the gesture, average length, ranges of angles and the acceleration, etc.

3.4 Proposed Method

In this section, we give the information about the SED architecture that has been used to achieve the best results as well as the training strategy that has been employed to take advantage of the architecture to be able to recognize multiple subgestures by training on very limited data.

3.4.1 SED Architecture

The following is the architecture developed to use for the problem.

Encoder : In the encoder, a bi-directional LSTM Schuster and Paliwal (1997) has been used with a state size **L**. The sequence of inputs is the stream of data arising from the IMU which is a 6-dimensional stream of data. After the forward propagation of the

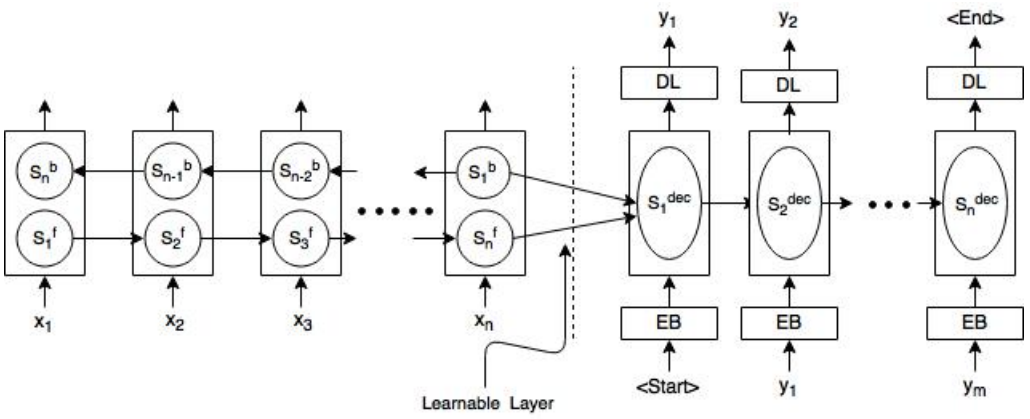

Figure 3.2: Selective Encoder Decoder Architecture.

sequence, whatever state remains is the encoded information of the input data.

The Encoder needs to read the whole sequence and condense the information into some tangible form which is the final state here. The bi-directional LSTM has been added because it can capture the information in a more abstract manner. For example suppose there are 2 gestures, a right swipe and a clockwise circle, here both of the gestures initially have a same trajectory and thus a standard LSTM might not be able to classify well. Reading the data from both directions(i.e usage of bi-directional LSTM) can and has improved the accuracy based on the experiments.

In between Encoder and Decoder : Now that the information about the input sequence is contained in the final state of the Encoder, we add a learnable layer of weights and biases between it and the initial state vector of the Decoder. The dimensions of this layer are $[2L, 2L]$(2L is the state size of Decoder and the Encoder has 2 LSTMs each with state size L)

The idea of having a learnable layer between the Encoder and the Decoder is so that the Decoder can selectively extract the information it requires from the encoded data similar to the functionality of selective read for state in LSTMs. This modification has proved to be impactful and improved the performance validated from the experiments performed.

Decoder : For the decoder, we use a standard LSTM with the initial state that has been obtained by multiplying the values in the final encoder states with the layer of weights. Here, the vocabulary of the target domain is the set of basic gestures that have been collected in addition to two manually added elements $< Start >$ and $<$

$End >$. At the input side, before the input is fed to the LSTM layer, it passes through an embedding layer T. Mikolov (2013). At the output side, connected to the outputs of the LSTM is a dense layer of the size of the vocabulary on top of which exists the softmax layer to predict the right output which is then fed in as the input at next time step. The first input is the $< Start >$ symbol and the prediction continues till the $< End >$ symbol is reached.

On the decoder side, while the functionality is common from the original decoder network, the embedding layer helps to learn similarities between various simple gestures from the vocabulary set which has helped to improve performance.

Mathematically, for an input sequence **x** = (x_1, x_2, x_3 .... x_n), the state h_t at every time instant t in the encoder depends as

$$h_t = f(x_t, x_{n-t}, h_{t-1}) \tag{3.2}$$

and, the final state c remains as

$$h_t = f(x_t, x_{n-t}, h_{t-1} \tag{3.3}$$

The selective layer in between the encoder and decoder converts the final state c in to c' based on the dense layer parameter M as follows

$$c' = M * c \tag{3.4}$$

which is when the decoder converts this information into the sequence accordingly maximizing the following probability

$$p(y) = \prod_{t=1}^{m} p(y_t/(y_1, ..., yt-1)), c') \tag{3.5}$$

And since an LSTM is being used on the decoder side,

$$p(y_t/(y_1, ..., yt-1)), c') = g(y_{t-1}, s_t, c') \tag{3.6}$$

where s_t is the state of the LSTM at time T and g is the update function of LSTM.

3.4.2 Training Strategy

When it comes to recognizing the subgestures, we have formulated a new strategy which would help in identifying multiple subgestures in sequence through a simple method without using much training data. In this case, we use training data only from simple gestures that have been recorded to identify even multiple subgestures in a sequence.

Here, to identify multiple subgestures in a sequence we create the relavant dataset simply by taking the permutations of the subgestures data that has been collected and taking a random subset of the permutations that have been taken. In this way, just with very little amount of data that is collected, it is possible to create a dataset for the outcome that we would like to see.

This especially is very efficient and works mainly because of the nature of the architecture. Because of the combination of the linearities and non linearities that exist in the computation of states and outputs, and the scalability in terms of number of smallest units that can be accomodated especially in the encoder part, it makes possible to be able to pack information very compactly. Also the fact that LSTMs are a sequentially processing units, this infrastructure helps with recognizing the patterns in the sequence and then the decoder being able to tag them to the relevant subgestures.

In this method, particularly for this architecture, we can observe an analogy with that of the problem of machine translation. In many cases during translation, many words in one language gets translated into just a single word in the other language and vice versa. Here in this case can be mapped to the problem of the first part only i.e many words to one. Thus, in a way it's just a subset of the more successful implementation of machine translation.

CHAPTER 4

Experiments

In this chapter, we give information about all the methods and experiments that have been performed and their corresponding results as well as the ideas that lie behind the formulation of the successful method about which has been mentioned in the subsection "Architecture".

We compare and contrast the approaches that have been tried in order to achieve the goal of gesture sequence recognition. In particular, we shall describe about the methods like Parallel Hidden Markov Models, LSTMs and Encoder Decoder Models that have been used to achieve the goal of gesture sequence recognition. We give intrinsic details about the intuition behind the application of the methods as well as qualitative results including the accuracy rates, F1 scores and the Confusion Matrices.

4.1 Recognition using Parallel HMM

4.1.1 Introduction

Parallel HMM is an arrangement of multiple left-right HMMs that are arranged in the manner shown in the picture. As seen in the picture, the start of each HMM is connected to the start state as well as the end of each HMM is connected to the end state. Here, in this case the start and the end states are not entitled to contribute to the emission of symbols which make it the case that the emission probabilities of all the symbols at these states are equal.

Here, the idea is that each of the left-right HMM corresponds to that of a certain subgesture and when a certain complex gesture is performed, the path taken over the complete structure which maximizes the probability for the gesture performed is the one which includes the each left-right HMM corresponding to each subgesture in the right sequence. This kind of structure has also been useful in the case of human emotion recognition based on body language.

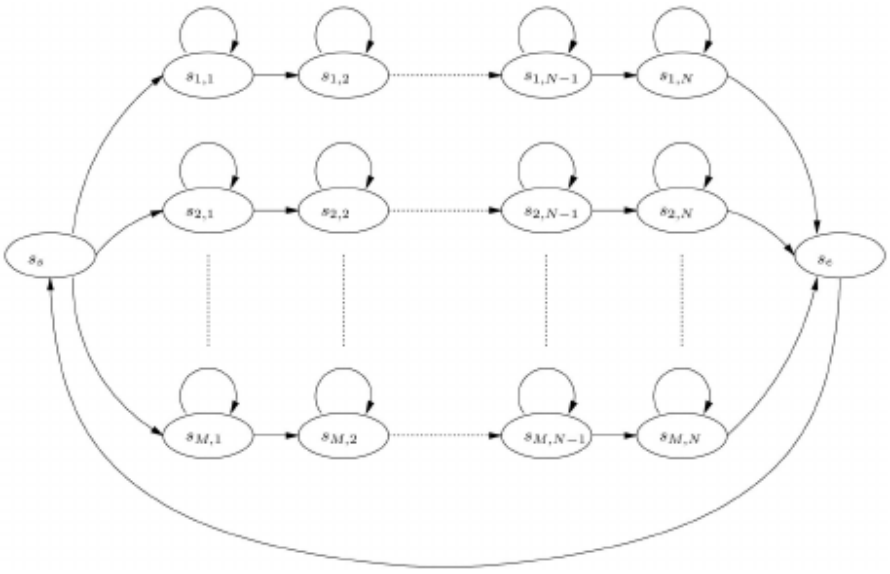

Figure 4.1: Example for pictorial representation of the embeddings on the two principal axes

4.1.2 Training

For training this structure, preprocessing was initially done on the procured data. For training the HMM in general using the Baum Welsch Algorithm, the outputs that need to be trained on should be whole numbers. For this purpose, we converted every data sample in the gestures to a whole number by usage of clustering methods like K-means and Gaussian Mixture Models and attaching the cluster number for each data point. This method has been successful previously in the context of speech recognition where specifically the subgestures here correspond to syllables there.

Subsequently, each of the HMM branch in the structure is trained separately using the training samples of that particular subgesture. Later, all the transition and emission probabilities of each state in each branch are combined into the master matrices for the whole structure. In these matrices, as mentioned above about the start and stop states, transition probabilities from all end states of each branch to the end state is equal and the transition probabilities from the start state to all the other start states in each branch is the same. Similarly, the emission probabilities at the end state and the start state for all the symbols are hardcoded to be the same.

4.1.3 Testing

Now that the HMM model is formed. When a new gesture is recorded for finding the subgestures in the right sequence, each sample of the gesture is first fitted to the clustering model that was done before during training. Now the state sequence is figured out by using the forward propagation algorithm using the transition and emission matrices that were computed. From this state sequence, we can find out the HMM branches that were involved and thus the subgestures in sequence.

4.1.4 Disadvantages

While conceptually the model looks sound, this approach has a lot of disadvantages which are the following.

1. Fixed number of states for each branch : It is not possible to have multiple number of states for each branch as there would be problems while normalizing the probabilities to find the path of maximum probability. Thus, there is no flexibility over the nature of the subgestures that are used.

2. Tag for start and end : In this method there is no tag for the start and end of the subgestures that can be captured which makes it difficult for the structure to segregate the gesture into multiple subgestures. This mainly creates confusion at the end and start states which creates the confusion.

3. Cannot accomodate variability : Since, every sample of the gesture is usually clustered to give a certain tag, the space over which this model can be used is very limited and depends quite a lot on the number of clusters that are used which in turn increases the computational resources to be used.

Thus, this model wasn't quite successful when used in this problem.

4.2 Basic Sequence to Sequence Model

4.2.1 Introduction

Sequence to Sequence models were first used for the problem of language translation which has shown successful results on the famous WMT dataset of English to French translations. It is a basic version of the architecture that has been described in the section above "Architecture Used".

In this particular architecture, the encoder and decoder are identical elements where each of them is a basic RNN with an embedding layer before the inputs. The working mechanism of the structure is quite similar to the one mentioned before except for the fact that the state here in this case is simply copied manually from the encoder to that of the decoder rather than having a learnable layer.

At the visually optimal tradeoff between performance, training time and space consumed the network used a state_size of 128 and the number of layers to be 3 both in the encoder and decoder side. Also, the most optimal performance was obtained when the cluster size was 80.

4.2.2 Training

Similar to the case of Parallel HMM, initially here all the sample points are first clustered and which forms the training input. Similarly, for the training output, each of the subgesture is given a separate tag and is provided as a sequence. Now we have the input seqeuence and output sequence which together make the training data.

It was also an important factor for proper ratios of training samples of different permutation length of the subgestures. For us, the best ratio that worked well was 20:30:50 of subgestures of permutation length of 1,2,3 respectively. Training was thus done using the Adam optimizer using the sum of categorical crossentropy as the loss function using a batch size of 64.

4.3 Updated version

4.3.1 Improvements

Improvement 1: Separately thinking about the encoder and the decoder, we tried to improve the performance of each of them separately. Thus thinking about the encoder, we tried classifying the subgestures separately using the encoder only so as to decide the best encoder.

For this we performed experiments using different units. Firstly, we used the basic RNN to classify the subgestures and got an abysmal performance. Later, we performed the classification using the LSTM which improved the performance but still was not at the best. Later, we used the Bi-directional LSTM which was able to classify really well and that too was able to learn the right parameters very quickly.

Thus for this purpose we made the encoder unit to be the bidirectional LSTM with the state size of 128*2 and thus the basic unit of the decoder as well to the normal LSTM of size 256.

Improvement 2: Inspired by the naively good results that were obtained in the previous method, we looked at the cause of such a weak behaviour and found out that the performance was limited with maxima at cluster size of 80. The performances kind of followed a bell shaped curve with respect to the cluster size.

Thus, originated the idea to remove dependence on the cluster size by simply removing the embedding layer on the side of the encoder because of the fact that the training samples have continuous smooth data of whose advantage can be tapped into by the neurons of the encoder which in turn can automatically have the effect of clustering while in action. As it turned out to be, we have observed a good rise in the performance of the network.

Improvement 3: With decently good results at this juncture, the encoder and decoder were separately performing well and thus didn't require much of changes. Thus, we made an observation that something could be done at the interface that arised in the following idea.

Inspired by the selective read function in the state updation of the LSTMs, we then

introduced a learnable layer in between the Encoder and the Decoder. This was done with the intuition that the decoder should be able to pick the right information it requires in a linear fashion which will make it easier for being able to decode the information. This improvement has increased the performance majorly and brought the present level of excellent performance.

4.4 Results

In this section, we give information about the recognition models that have been used to compare the results, training methods used to achieve the results and results obtained by using all of the above recognition models.

For training the different models mentioned below, we used the Adam optimizer D. Kingma (2014) to minimize the loss function of summation of the categorical cross entropy of the predicted gestures. This was performed on a training batch size of 64 with 10 epochs where epochs 1-5 have a learning rate of 0.1, epochs 6-8 have a learning rate of 0.05 and epochs 9-10 have a learning rate of 0.02.

4.4.1 Recognition Models

For showing the superiority of the architecture proposed, we show the comparison of performances of the following recognition architectures.

Normal Classification(gestures of sequence length 1) :-

- Standard LSTM
- Bi-directional LSTM

Sequence of gestures(gestures upto sequence length 3) :-

- The Selective Encoder Decoder architecture with a copy function instead of a learnable layer in between the encoder and the decoder. (SED-C)
- Selective Encoder Decoder (SED)

Below, we show in order the performance of the recognition models for only gestures with sequence length 1, gestures with sequence length 2/3 and gestures with sequence length 1/2/3 combined.

4.4.2 Gestures with sequence length 1

The following table shows the comparison of different algorithms of the gestures captured with the composition of sequence with 1 gesture only.

Method/Accuracies	Length 1
LSTM	66.4
Bi-directional LSTM	**97.7**
SED - C	95.4
SED	97.3

Table 4.1: This table shows the accuracies of various algorithms on gestures of sequence length 1

From the table, we can see that the bi-directional LSTM, SED-C and SED perform really well. This can be attributed mainly to the proficiency of bi-directional LSTM which also happens to be the major part of SED-C and SED.

4.4.3 Gestures with sequence length 2 or 3

To establish the superiority of SED architectures while predicting gestures with sequence length of more than 1, the respective recognition models have been trained separately with gestures with sequence length 2 in experiment 1 and gestures with sequence length 3 in experiment 2. Also, the recognition models have been hardcoded to predict 2 or 3 labels respectively for the above mentioned experiments i.e prediction stops after the architecture has predicted 2 or 3 labels respectively. The following table shows the performance of the recognition models on the data.

Method/Accuracies	Length 2	Length 3
SED - C	91.2	88.4
SED	**94.6**	**90.2**

Table 4.2: This table shows the accuracies of the architectures on gestures of sequence length 2 or 3 separately.

The above table cements the effectiveness of the SED class architectures in problems related to subgesture sequence identification given the gesture. The main reason that can be attributed to their effectiveness is this architectures' ability to be able to recognize the patterns of data to be able to predict the subgestures in sequence.

4.4.4 Gestures with sequence length 1,2 and 3 combined

In this experiment, we train the architecture on gestures with sequence lengths 1,2 and 3 at once and let the model predict the sequence of gestures. In this experiment, the prediction stops when the model predicts a "Stop". The following table shows the results for this experiment.

Method/Accuracies	Length 1	Length 2	Length 3
SED - C	84.3	71.6	66.4
SED	**90.2**	**86.7**	**77.5**

Table 4.3: This table shows the accuracies of the architectures used when trained with gestures of length 1,2 and 3 combined

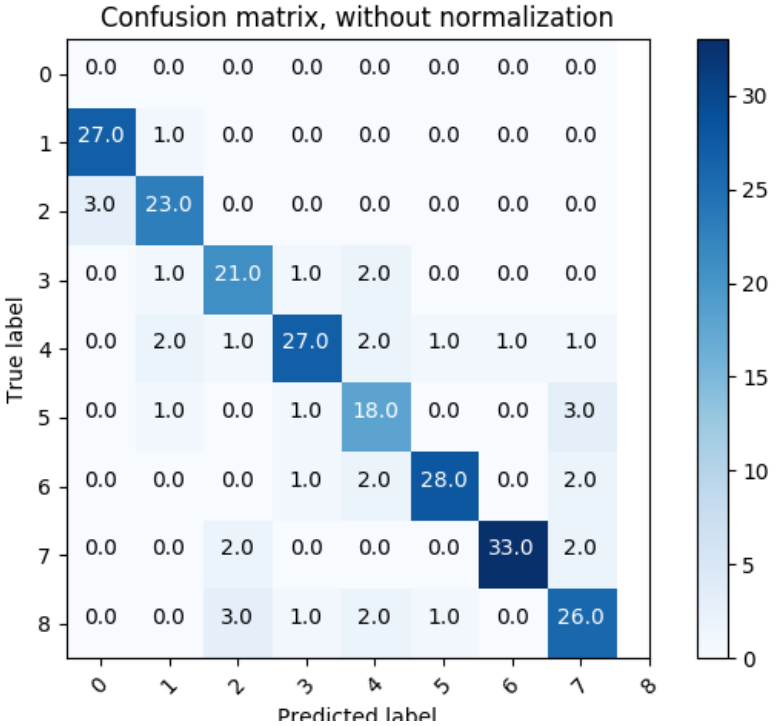

Figure 4.2: Confusion Matrix of gestures of sequence length 1 on SED architecture

Here, we can see the competance of the SED architecture. In this case of predicting the sequence of gestures of variable sequence lengths just using the same architecture at one go after training on all variants of sequence lengths, the SED architecture clearly shows a quantum jump in terms of accuracies. This can be attributed to the learnable layer in between the Encoder and Decoder which performs the function of selective read similar to the one that exists in the standard LSTM.

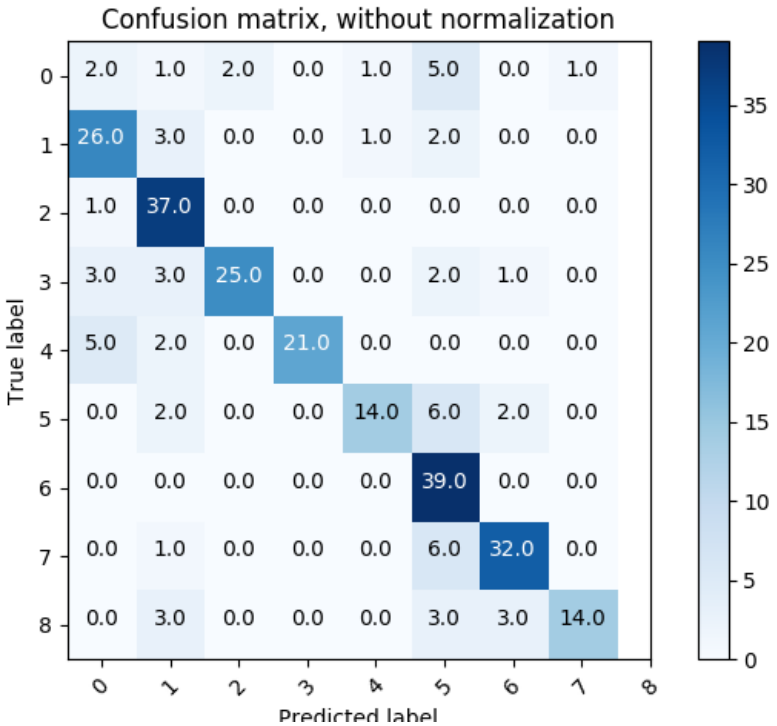

Figure 4.3: Confusion Matrix of gestures of sequence length 2 on SED architecture

Also, using the SED architecture, the F1 scores obtained for gestures with 1,2 or 3 sequence lengths respectively are **0.89**, **0.82** and **0.78**. The confusion matrix for the above experiment performed using the SED architecture is shown in Figure 3.

4.4.5 Training Strategy

For the normal classification problem, the maximum sequence length of measurements among all the gesture is computed and all the other gestures have been padded with zeros till the max length. Each modified sequence of gesture measurements were then trained with each model along with their correct class.

For the sub-gesture classification problem, it was not necessary to create a separate dataset pertaining to the sub-gestures classification. We took the permutations of all the recorded gestures, padded them according to the maximum length of such gestures and trained it on the model along with the sequence of subgestures they belonged to.

This method of training was effective because the model only had to learn the patterns of the each of the gestures which would make it sufficient be able to recognize the sequence of subgestures. We show that this method is effective from the results we

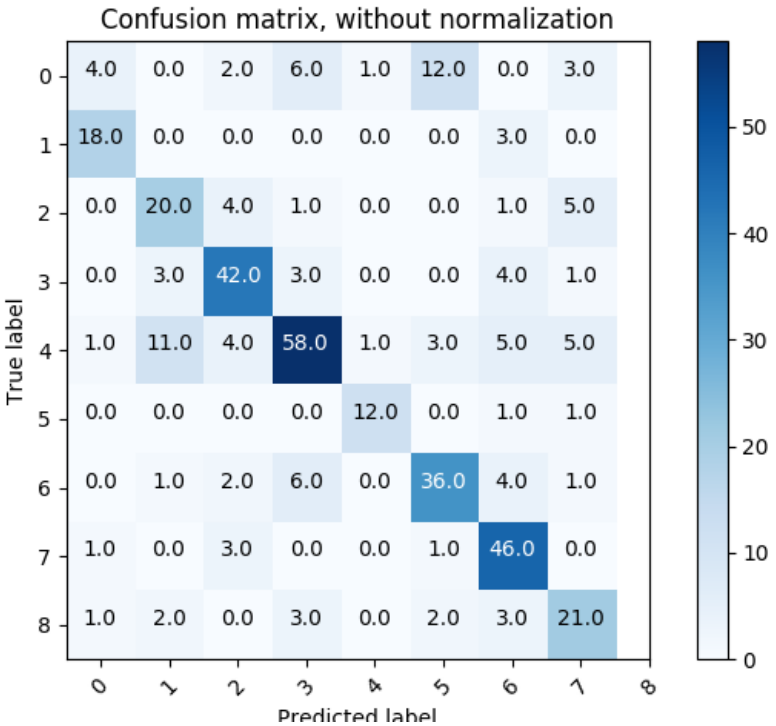

Figure 4.4: Confusion Matrix of gestures of sequence length 3 on SED architecture

obtained.

CHAPTER 5

Conclusion

Through this work, we have shown a method that has been successful in recognizing the gestures in a sequence using the IMU. We believe that it is a big step in the direction of ubiquitous control in the field of Human Computer Interaction basically by increasing the field of possibilities just by using one instrument - here namely the hand. We have also shown that exquisiteness of the architecture involved by it's performance capabilities just by utilizing a fraction of real collected data which otherwise in other cases have never been possible. In the future, such capabilities are going to become quite common and a new field of research that would be complementing this area and also be very interesting to look into is the real time gesture segmentation. Such a combination of capabilities in future will be the way to work towards bringing the revolution of ubiquitous usage of Human Computer Interactive devices.

APPENDIX A

A SAMPLE APPENDIX

Just put in text as you would into any chapter with sections and whatnot. Thats the end of it.

REFERENCES

1. **CD. Mitchell, R. H. e. a.** (1995). A parallel implementation of a hidden markov model with duration modelling for speech recognition.

2. **D. Bahdanau, K. C. e. a.** (2015). Neural machine translation by jointly learning to align and translate. *ICLR*.

3. **D. Kingma, J. A.** (2014). A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

4. **Elmezain, M.** and **A. Al-Hamadi** (2008). A hidden markov model-based isolated and meaningful hand gesture recognition. *World Academy of Science, Engineering and Technology*, **31**, 2070–3740.

5. **I. Sutskever, O. V. e. a.** (2014). Sequence to sequence learning with neural networks. *NIPS*.

6. **J. Galka, M. M. e. a.** (2016*a*). Inertial motion sensing glove for sign language gesture acquisition and recognition. *IEEE Sensors Journal*, **16**(16).

7. **J. Galka, M. M. e. a.** (2016*b*). Inertial motion sensing glove for sign language gesture acquisition and recognition. *IEEE Sensors Journal*, **16**(16).

8. **K. Cho, B. M. e. a.** (2014). Learning phrase representations using rnn encoderâĂŞdecoder for statistical machine translation. *EMNLP 2014*.

9. **Mitra, S.** and **T. Acharya** (2007). Gesture recognition: A survey. *IEEE Transactions on Systems, Man and Cybernetics*, **37**(311).

10. **P. Molchanov, S. G. e. a.** (2015). Hand gesture recognition with 3d convolutional neural networks. *CVPR*.

11. **P. Pan, Z. X. e. a.** (2015). Hierarchical recurrent neural encoder for video representation with application to captioning. *arXiv:1511.03476*.

12. **R. Aggarwal, S. e. a.** (2015). Online handwriting recognition using depth sensors. *Document Analysis and Recognition (ICDAR)*.

13. **Schuster, M.** and **K. Paliwal** (1997). Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, **45**(11).

14. **T. Mikolov, I. S. e. a.** (2013). Distributed representations of words and phrases and their compositionality. *NIPS*.

15. **VI. Pavlovic, R. S. e. a.** (1997). Visual interpretation of hand gestures for human-computer interaction: A review. *PAMI*, **19**(677).

16. **Wong, S.** and **R. Cipolla** (2007). Continuous gesture recognition using sparse bayesian classifier. *ICPR*.

17. **Yang, Z.** and **S. Narayanan** (2015). Modelling dynamics of expressive gestures in dyadic interactions. *Journal of Affective Computing*, **10**(10).

18. **Yin, Y.** (2014). Real time continuous gesture recognition for natural multimodal interaction. *PhD Thesis, MIT*.

19. **Z. Yang, A. O. e. a.** (2014). Gesture dynamics modelling for attitude analysis using graph based transform. *ICIP*.

LIST OF PAPERS BASED ON THESIS

1. Authors.... Title... *Journal*, Volume, Page, (year).