REAL TIME SIGN LANGUAGE GESTURE RECOGNITION FROM VIDEO SEQUENCES

A PROJECT REPORT

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

BACHELOR OF TECHNOLOGY
IN
COMPUTER ENGINEERING

Under the Supervision of
Mr. Sarfaraz Masood
Assistant Professor,
Dept. of Computer Engg.
Jamia Millia Islamia

Submitted by
Harish Chandra Thuwal
(13-BCS-0027)
Adhyan Srivastava
(13-BCS-0007)

DEPARTMENT OF COMPUTER ENGINEERING
FACULTY OF ENGINEERING AND TECHNOLOGY
JAMIA MILLIA ISLAMIA, NEW DELHI - 110017

(Year-2017)

DECLARATION

We, **Harish Chandra Thuwal** and **Adhyan Srivastava**, students of Bachelor of Technology, Computer Engineering hereby declare that the project entitled "Real Time Sign Language Gesture Recognition from Video Sequences" which is submitted by us to the Department of Computer Engineering, Faculty of Engineering and Technology, Jamia Millia Islamia, New Delhi in partial fulfilment of requirements for the award of the degree of Bachelor of Technology in Computer Engineering, has not been formed the basis for the award of any degree, diploma or other similar title or recognition.

Place: New Delhi

Date:

Harish Chandra Thuwal
(13BCS0027)

Adhyan Srivastava
(13BCS0007)

CERTIFICATE

This is to certify that the dissertation/project report(Course Code) entitled "Real Time Sign Language Gesture Recognition from Video Sequences", is an authentic work carried out by **Harish Chandra Thuwal** and **Adhyan Srivastava.**

The work is submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Engineering under my guidance. The matter embodied in this project work has not been submitted earlier for the award of any degree or diploma to the best of my knowledge and belief.

Prof. M.N Doja,
Professor and Head,
Dept. of Computer Engg.
Jamia Millia Islamia

Sarfaraz Masood,
Assistant Professor,
Dept. of Computer Engg.
Jamia Millia Islamia

ACKNOWLEDGMENT

We would like to thank our mentor **Mr. Sarfaraz Masood** (Assistant Professor, Department of Computer Engineering) for giving us the opportunity to undertake the project. We thank them for their immense guidance, and appreciate their timely engagement.

We would like to extend special gratitude to the the assistants and lab coordinators of the Department for providing us the infrastructural facilities necessary to sustain the project.

ABSTRACT

Inability to speak is considered to be true disability. People with this disability use different modes to communicate with others, there are number of methods available for their communication one such common method of communication is sign language.

Developing sign language application for deaf people can be very important, as they'll be able to communicate easily with even those who don't understand sign language. Our project aims at taking the basic step in bridging the communication gap between normal people, deaf and dumb people using sign language.

The main focus of this work is to create a vision based system to identify sign language gestures from the video sequences. The reason for choosing a system based on vision relates to the fact that it provides a simpler and more intuitive way of communication between a human and a computer. In this report, 46 different gestures have been considered.

We used the following approach for the classification of sign language gestures:-

Video sequences contain both the temporal as well as the spatial features. So we have used two different models to train both the temporal as well as the spatial features. To train the model on the spatial features of the video sequences we have used Inception model[14] which is a deep CNN(convolutional neural net). CNN was trained on the frames obtained from the video sequences of train data. We have used RNN(recurrent neural network) to train the model on the temporal features. Trained CNN model was used to make predictions for individual frames to obtain a sequence of predictions or pool layer outputs for each video. Now this sequence of prediction or pool layer outputs was given to RNN to train on the temporal features.The data set[7] used consists of Argentinian Sign Language(LSA) Gestures, with around 2300 videos belonging to 46 gestures categories. Using the predictions by CNN as input for RNN 93.3% accuracy was obtained and by using pool layer output as input for RNN an accuracy of 95.217% was obtained.

TABLE OF CONTENTS

1 Introduction **8**

1.1 Sign Language 8

---

2 Literature Survey **10**

2.1 Vision Based 10

2.1.1 Handshape recognition for Argentinian Sign Language using ProbSom 11

2.1.2 Automatic Indian Sign Language Recognition for Continuous Video sequence. 12

2.1.3 Continuous Indian Sign Language Gesture Recognition and Sentence Formation 12

2.1.4 Recognition of isolated Indian Sign Language Gesture in Real TIme 14

---

3 Algorithms **15**

3.1 Convolutional Neural Network 15

3.1.1 CNN Summarised in 6 Steps 15

3.1.1.1 Convolution 16

3.1.1.2 Subsampling 19

3.1.1.3 Pooling 19

3.1.1.4 Activation 20

3.1.1.5 Fully Connected 20

3.1.1.6 Loss (During Training) 20

3.1.2 Implementation 20

3.1.3 Inception 21

3.2 Recurrent Neural Network 22

3.2.1 Recurrent Neural Networks have Loops 23

3.2.2 How memory of previous inputs Carried Forward 24

3.2.3 Exploding and Vanishing Gradient 25

3.2.3.1 Vanishing Gradient 25

3.2.3.2 Exploding Gradient 26
3.2.4 Long Short Term Memory Units 26
3.2.5 Our RNN Model 27

4 Experimental Design **28**

4.1 Dataset Used 28
4.2 First Approach 29
4.2.1 Methodology 29
4.2.1.1 Frame Extraction and Background REmoval 30
4.2.1.2 Training CNN (Spatial Features) and Prediction 31
4.2.1.3 Training RNN (Temporal Features) 32
4.2.2 Limitation 32
4.3 Second Approach 32

5 Results **33**

5.1 Result of Approach 1 33
5.2 Result of Approach 2 33

6 Conclusion and Future Work **35**

7 References **36**

LIST OF FIGURES

Fig 1	American Sign Language	09
Fig 2	Block Diagram Vision Based Recognition System	10
Fig 3	LSA Hand Gesture Recognition System block diagram	11
Fig 4	System Overview	12
Fig 5	General Diagram of the Work	13
Fig 6	Gesture of Sentence It is Closed Today	13
Fig 7	Methodology for real time ISL classification	14
Fig 8	Convolutional Neural Network	15
Fig 9	Convolving Wally with a circle filter	16
Fig 10	Dot product of filter with single chunk of input image	17
Fig 11	Dot product or Convolve over all all possible 5x5	17
Fig 12	Input Image Convolving with layer of 6 filters	18
Fig 13	Input Image with two Conv layer having 6 & 10 filters	18
Fig 14	Subsampling Wally by 10 times	19
Fig 15	Pooling to reduce size	19
Fig 16	Max Pooling	20
Fig 17	Inception v3 model	22
Fig 18	A chunk of RNN	23
Fig 19	An Unrolled recurrent neural network	23
Fig 20	Memory of previous inputs being carried forward	24
Fig 21	Vanishing Sigmoid	25
Fig 22	Our RNN model	27
Fig 23	Approach 1 Methodology	29
Fig 24	One of the Extracted Frames	30
Fig 25	Frame after extracting hands	30
Fig 26	Train CNN and prediction	31
Fig 27	Train CNN and prediction illustration	31
Fig 28	Train RNN illustration	32
Fig 29	Results Approach 1	33
Fig 30	Results Approach 2	34

1. INTRODUCTION

Motion of any body part like face, hand is a form of gesture. Here for gesture recognition we are using image processing and computer vision. Gesture recognition enables computer to understand human actions and also acts as an interpreter between computer and human. This could provide potential to human to interact naturally with the computers without any physical contact of the mechanical devices. Gestures are performed by deaf and dumb community to perform sign language. This community used sign language for their communication when broadcasting audio is impossible, or typing and writing is difficult, but there is the vision possibility. At that time sign language is the only way for exchanging information between people. Normally sign language is used by everyone when they do not want to speak, but this is the only way of communication for deaf and dumb community. Sign language is also serving the same meaning as spoken language does. This is used by deaf and dumb community all over the world but in their regional form like ISL, ASL. Sign language can be performed by using Hand gesture either by one hand or two hands. It is of two type Isolated sign language and continuous sign language. Isolated sign language consists of single gesture having single word while continuous ISL or Continuous Sign language is a sequence of gestures that generate a meaningful sentence. In this report we performed isolated ASL gesture recognition technique.

1.1 Sign Language

Deaf people around the world communicate using sign language as distinct from spoken language in their every day a visual language that uses a system of manual, facial and body movements as the means of communication. Sign language is not an universal language, and different sign languages are used in different countries, like the many spoken languages all over the world. Some countries such as Belgium, the UK, the USA or India may have more than one sign language. Hundreds of sign languages are in used around the world, for instance, Japanese Sign Language, British Sign Language (BSL), Spanish Sign Language, Turkish Sign Language.

Sign language is a visual language and consists of 3 major components:

Fingerspelling	Word level sign vocabulary	Non-manual features
Used to spell words letter by letter .	Used for the majority of communication.	Facial expressions and tongue, mouth and body position.

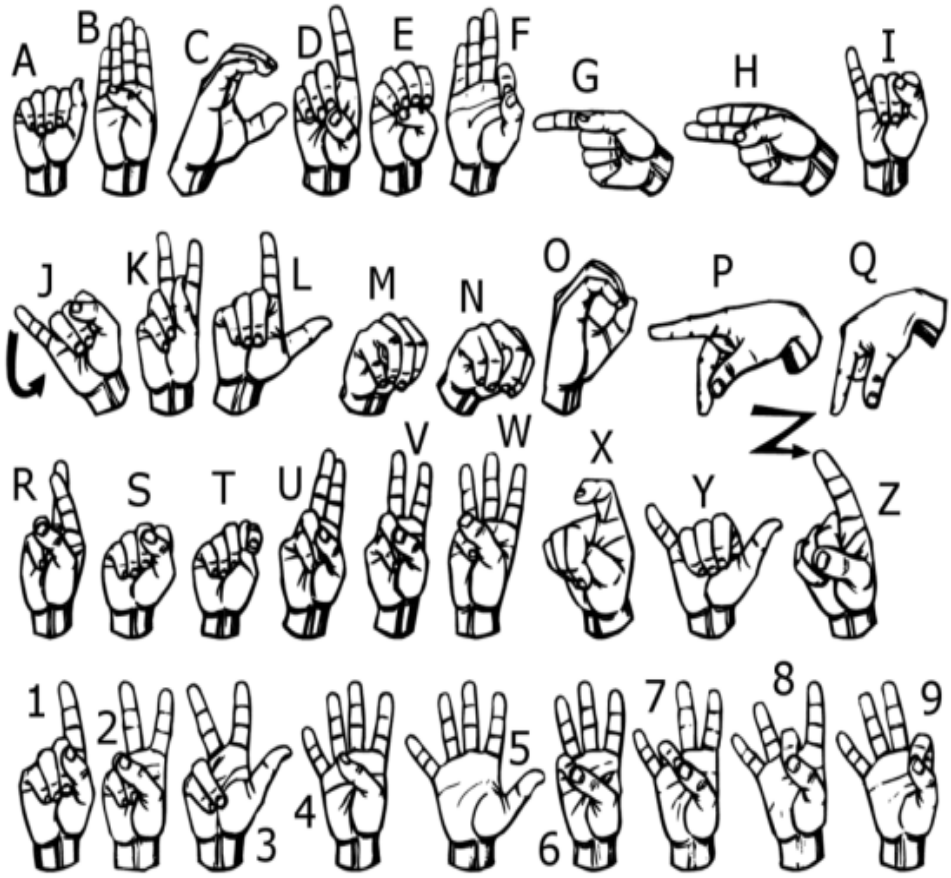

Fig 1: Finger Spelling American Sign Language[11]

2. LITERATURE SURVEY

In the recent years, there has been tremendous research on the hand sign language gesture recognition. The technology for gesture recognition is given below.

2.1 Vision-based

In vision-based methods computer camera is the input device for observing the information of hands or fingers. The Vision Based methods require only a camera, thus realizing a natural interaction between humans and computers without the use of any extra devices. These systems tend to complement biological vision by describing artificial vision systems that are implemented in software and/or hardware. This poses a challenging problem as these systems need to be background invariant, lighting insensitive, person and camera independent to achieve real time performance. Moreover, such systems must be optimized to meet the requirements, including accuracy and robustness.

The vision based hand gesture recognition system is shown in fig.--:

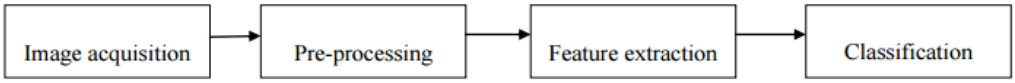

Fig 2: Block Diagram of vision based recognition system

Vision based analysis, is based on the way human beings perceive information about their surroundings, yet it is probably the most difficult to implement in a satisfactory way. Several different approaches have been tested so far.

1. One is to build a three-dimensional model of the human hand. The model is matched to images of the hand by one or more cameras, and parameters corresponding to palm orientation and joint angles are estimated. These parameters are then used to perform gesture classification.

2. Second one to capture the image using a camera then extract some feature and those features are used as input in a classification algorithm for classification.

2.1.1 Handshape recognition for Argentinian Sign Language using ProbSom [1]

In this paper, a method for hand gesture recognition of Argentinian sign language (LSA) is proposed. This paper offers two main contributions: first, the creation of a database of handshapes for the Argentinian Sign Language (LSA). Secondly, a technique for image processing, descriptor extraction and subsequent handshape classification using a supervised adaptation of self-organizing maps that is called ProbSom. This technique is compared to others in the state of the art, such as Support Vector Machines (SVM), Random Forests, and Neural Networks. The ProbSom-based neural classifier, using the proposed descriptor, achieved an accuracy rate above 90%.

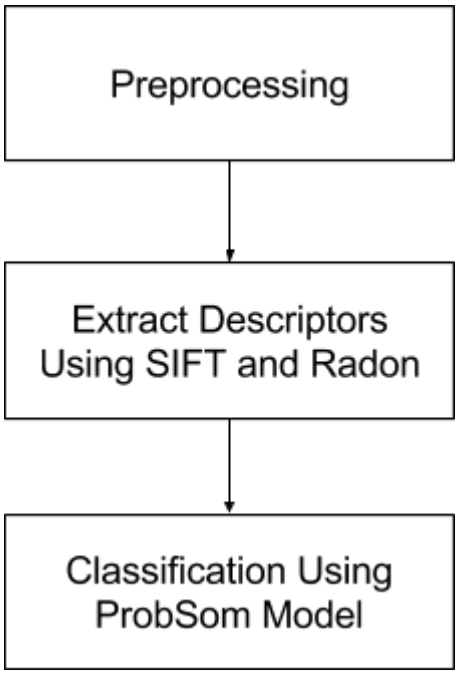

Fig 3: Block Diagram of Hand Gesture Recognition System for LSA

2.1.2 Automatic Indian Sign Language Recognition for Continuous Video Sequence [2]

The proposed system comprises of four major modules: Data Acquisition, Pre-processing, Feature Extraction and Classification. Pre-processing stage involves Skin Filtering and histogram matching after which Eigen-vector based Feature Extraction and Eigen value weighted Euclidean distance based Classification Technique was used. 24 different alphabets were considered in this paper where 96% recognition rate was obtained.

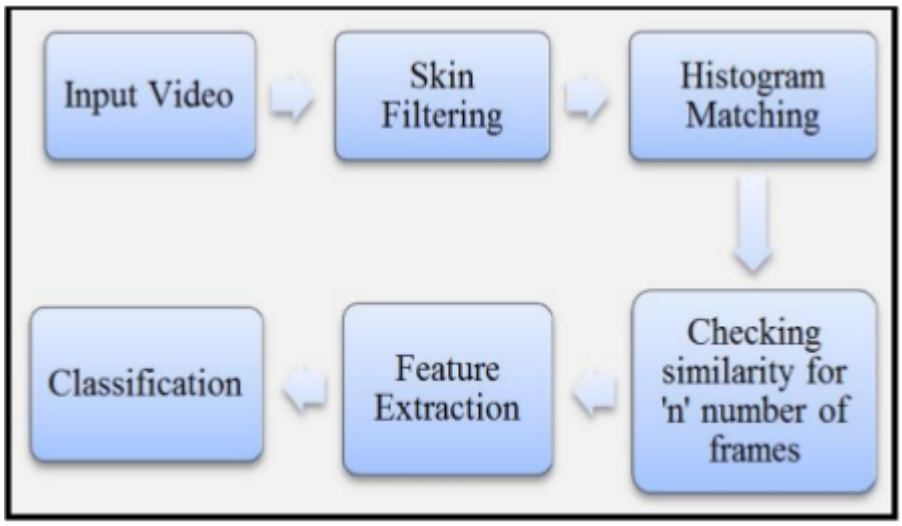

Fig 4: System Overview [2]

2.1.3 Continuous Indian Sign Language Gesture Recognition and Sentence Formation [3]

Recognizing a sign language gestures from continuous gestures is a very challenging research issue. The researchers solved this problem using gradient based key frame extraction method. These key frames were helpful in splitting continuous sign language gestures into sequence of signs as well as for removing uninformative frames. After splitting of gestures each sign has been treated as an isolated gesture. Then features of pre-processed gestures were extracted using Orientation Histogram (OH) with Principal Component Analysis (PCA) is applied for reducing dimension of features obtained after OH. Experiments were performed on their own continuous ISL dataset which was created using canon EOS camera in Robotics and Artificial Intelligence laboratory (IIIT-A). Probes were tested using various types of classifiers like

Euclidean distance, Correlation, Manhattan distance, city block distance etc. Comparative analysis of their proposed scheme was performed with various types of distance classifiers. From the above analysis they found that the results obtained from Correlation and Euclidean distance gives better accuracy than other classifiers.

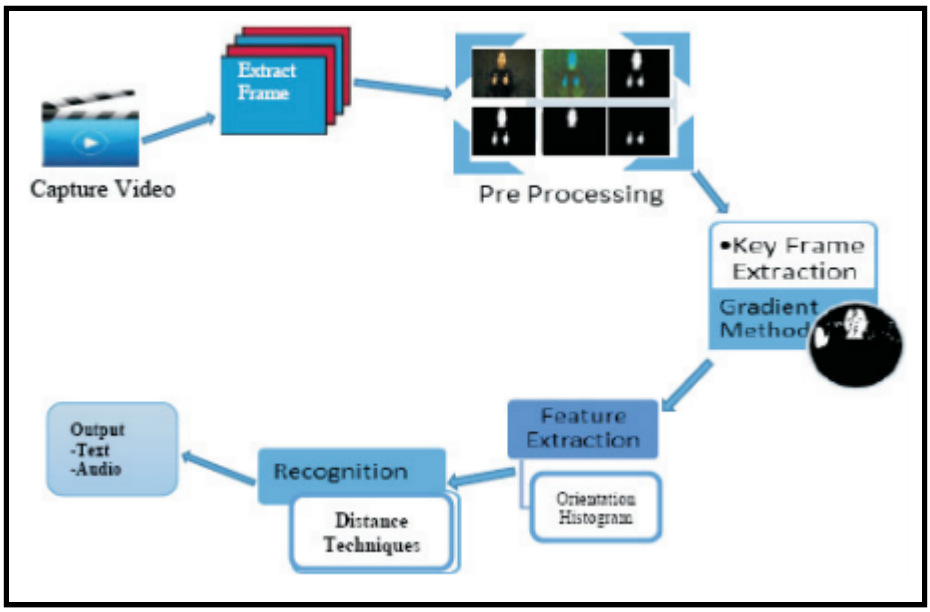

Fig 5: General Diagram of the Work [3]

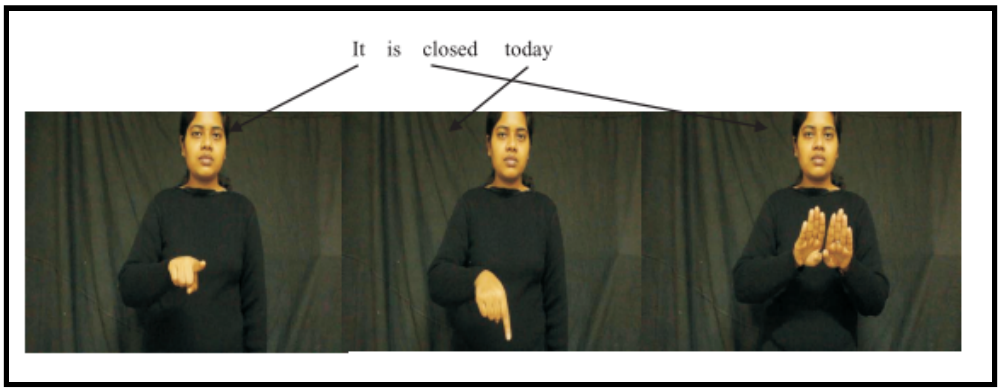

Fig 6: Gesture of Sentence It is Closed Today [3]

2.1.4 Recognition of isolated Indian Sign Language Gesture in Real Time [4]

This paper demonstrates the statistical techniques for recognition of ISL gestures in real time which comprises both the hands. A video database was created by the authors and utilized which contained several videos for large number of signs. Direction histogram is the feature used for classification due to its appeal for illumination and orientation invariance. Two different approaches utilized for recognition were Euclidean distance and K-nearest neighbor metrics.

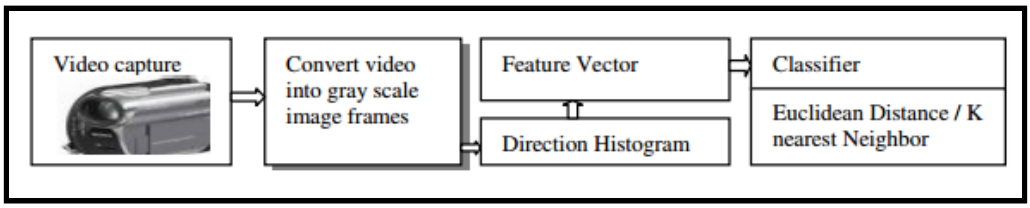

Fig 7: Methodology for real time ISL classification [4]

3.1 Convolutional Neural Network (CNN)

Neural networks, as its name suggests, is a machine learning technique which is modeled after the brain structure. It comprises of a network of learning units called neurons. These neurons learn how to convert **input signals** (e.g. picture of a cat) into corresponding **output signals** (e.g. the label "cat"), forming the basis of automated recognition.

A convolutional neural network (CNN, or ConvNet) is a type of feed-forward artificial neural network in which the connectivity pattern between its neurons is inspired by the organization of the animal visual cortex.

CNNs have repetitive blocks of neurons that are applied across space (for images) or time (for audio signals etc). For images, these blocks of neurons can be interpreted as 2D convolutional kernels, repeatedly applied over each patch of the image. For speech, they can be seen as the 1D convolutional kernels applied across time-windows. At training time, the weights for these repeated blocks are 'shared', i.e. the weight gradients learned over various image patches are averaged.

3.1.1 CNN Summarized in 4 Steps

There are four main steps in CNN: convolution, subsampling, activation and full connectedness.

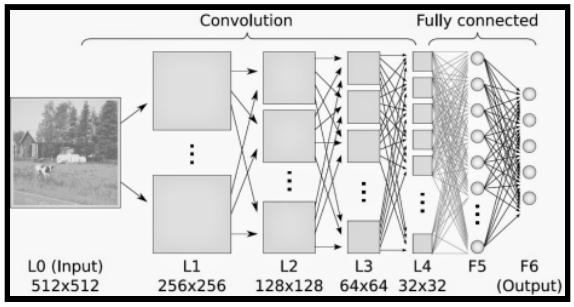

Fig 8: Convolutional neural network [13]

3.1.1.1 Convolution

The first layers that receive an input signal are called convolution filters. Convolution is a process where the network tries to label the input signal by referring to what it has learned in the past. If the input signal looks like previous cat images it has seen before, the "cat" reference signal will be mixed into, or convolved with, the input signal. The resulting output signal is then passed on to the next layer.

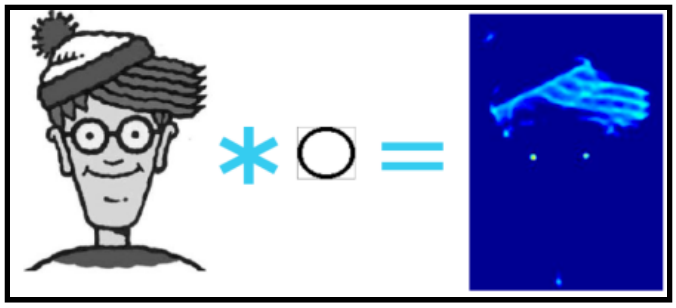

Fig 9: Convolving Wally with a circle filter. The circle filter responds strongly to the eyes.

Convolution has the nice property of being **translational invariant**. Intuitively, this means that each convolution filter represents a feature of interest (e.g whiskers, fur), and the CNN algorithm learns which features comprise the resulting reference (i.e. cat). The output signal strength is not dependent on where the features are located, but simply whether the features are present. Hence, a cat could be sitting in different positions, and the CNN algorithm would still be able to recognize it.

For e.g suppose we convolve a 32x32x3 (32x32 image with 3 channels R,G and B respectively) with a 5x5x3 filter. We take the 5*5*3 filter and slide it over the complete image and along the way take the dot product between the filter and chunks of the input image.

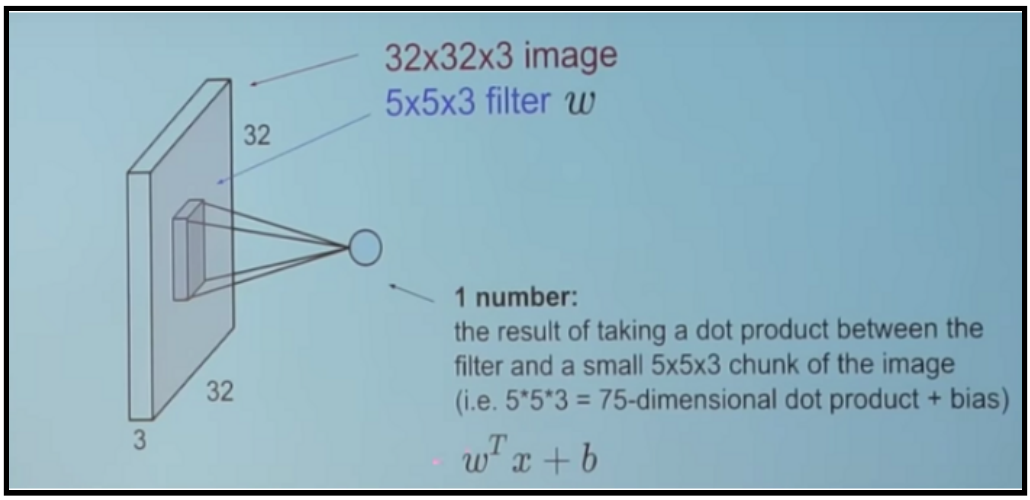

Fig 10: Dot Product of Filter with single chunk of Input Image[12]

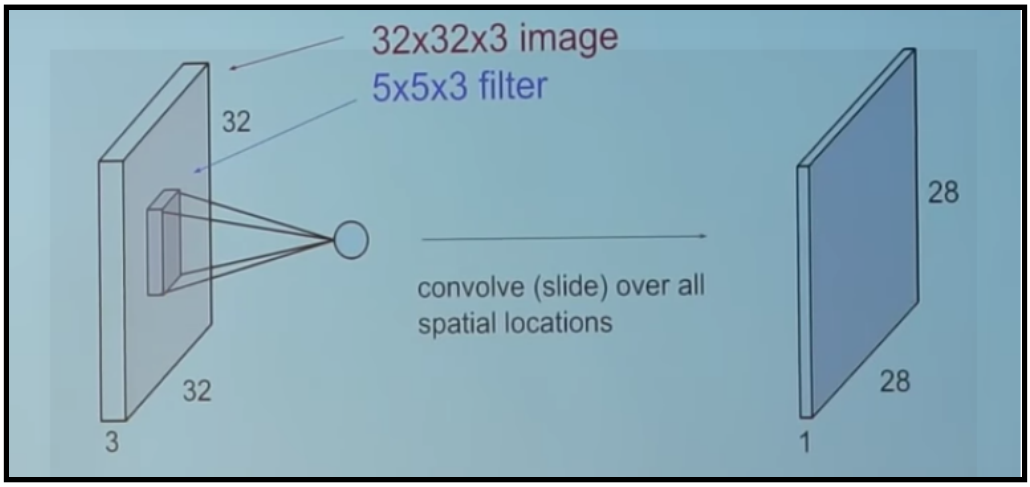

Fig 11: Dot Product or Convolve over all possible 5x5 spatial location in Input Image[12]

The convolution layer is the main building block of a convolutional neural network. The convolution layer comprises of a set of independent filters (6 in the example shown). Each filter is independently convolved with the image and we end up with 6 feature maps of shape 28*28*1.

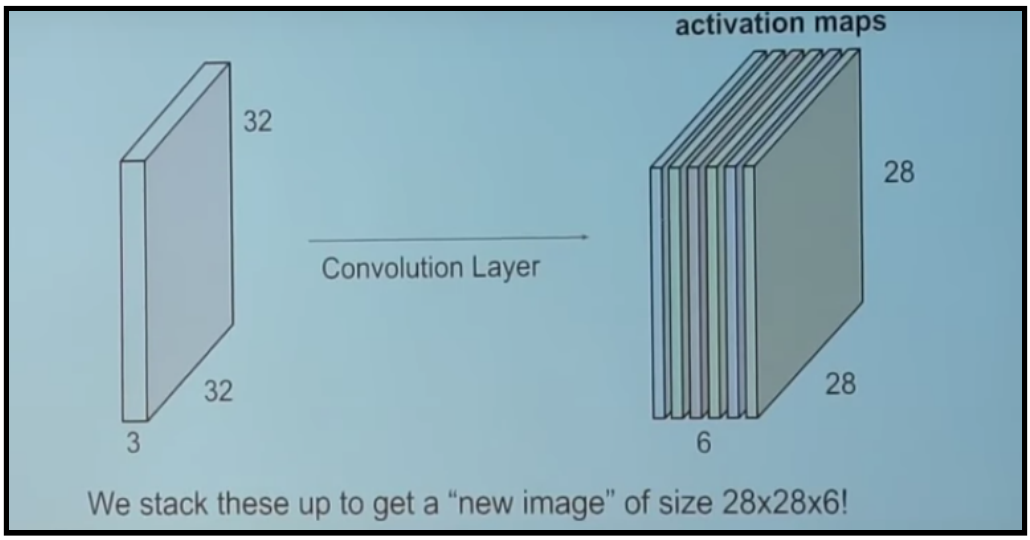

Fig 12: Input Image Convolving with a Convolutional layer of 6 independent filters[12]

The CNN may consists of several Convolutional layers each of which can have similar or different number of independent filters. For example the following diagram shows the effect of two Convolutional layers having 6 and 10 filters respectively.

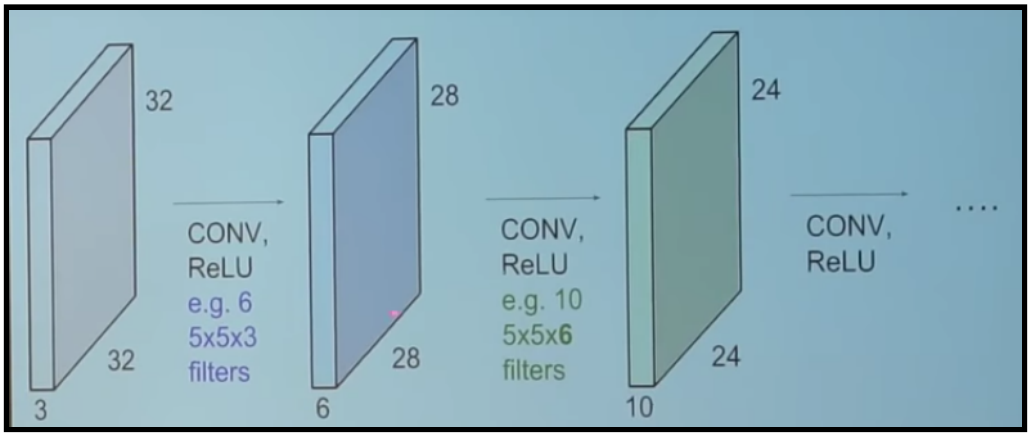

Fig 13: Input Image Convolving with two Convolutional layers having 6 and 10 filters respectively[12]

All these filters are initialized randomly and become our parameters which will be learned by the network subsequently.

3.1.1.2 Subsampling

Inputs from the convolution layer can be "smoothened" to reduce the sensitivity of the filters to noise and variations. This smoothing process is called **subsampling**, and can be achieved by taking averages or taking the maximum over a sample of the signal. Examples of subsampling methods (for image signals) include reducing the size of the image, or reducing the color contrast across red, green, blue (RGB) channels.

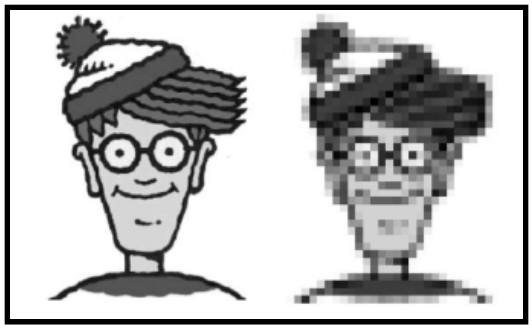

Fig 14: Sub sampling Wally by 10 times. This creates a lower resolution image.

3.1.1.3 Pooling

A pooling layer is another building block of a CNN.

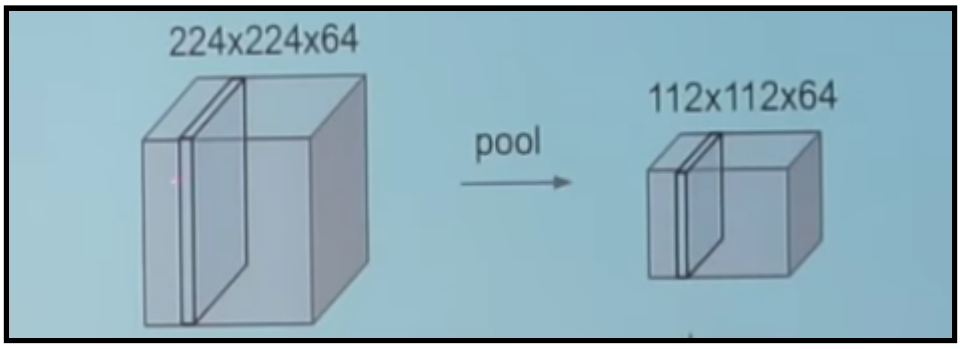

Fig 15: Pooling to reduce size from 224x224 to 112x112 [12]

Its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network. Pooling layer operates on each feature map independently.

The most common approach used in pooling is max pooling in which maximum of a region taken as its representative. For example in the following diagram a 2x2 region is replaced by the maximum value in it.

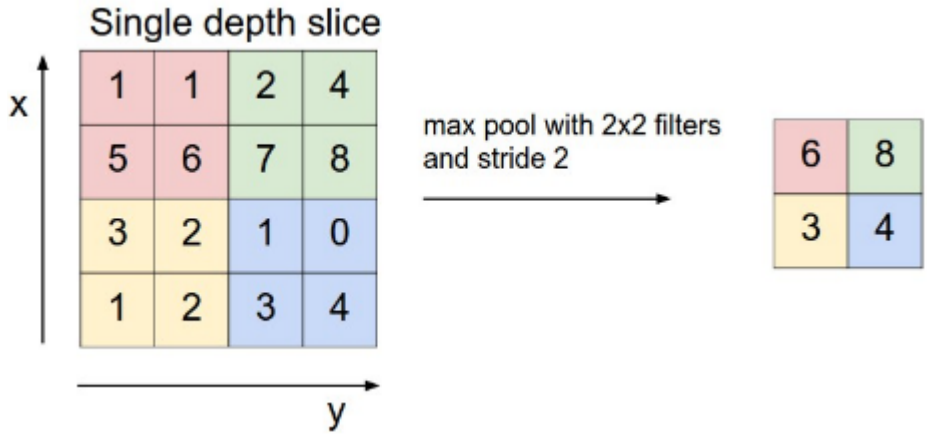

Fig 16: Max Pooling [12]

3.1.1.4 Activation

The activation layer controls how the signal flows from one layer to the next, emulating how neurons are fired in our brain. Output signals which are strongly associated with past references would activate more neurons, enabling signals to be propagated more efficiently for identification.

CNN is compatible with a wide variety of complex activation functions to model signal propagation, the most common function being the Rectified Linear Unit (ReLU), which is favored for its faster training speed.

3.1.1.5 Fully Connected

The last layers in the network are fully connected, meaning that neurons of preceding layers are connected to every neuron in subsequent layers. This mimics high level reasoning where all possible pathways from the input to output are considered.

3.1.1.6 (During Training) Loss

When training the neural network, there is additional layer called the loss layer. This layer provides feedback to the neural network on whether it identified inputs correctly, and if not, how far off its guesses were. This helps to guide the neural network to reinforce the right concepts as it trains. This is always the last layer during training.

3.1.2 Implementation

Algorithms used in training CNN are analogous to studying for exams using flash cards. First, you draw several flashcards and check if you have mastered the concepts on each card. For cards with concepts that you already know,

discard them. For those cards with concepts that you are unsure of, put them back into the pile. Repeat this process until you are fairly certain that you know enough concepts to do well in the exam. This method allows you to focus on less familiar concepts by revisiting them often. Formally, these algorithms are called gradient descent algorithms for forward pass learning. Modern deep learning algorithm uses a variation called stochastic gradient descent, where instead of drawing the flashcards sequentially, you draw them at random. If similar topics are drawn in sequence, the learners might over-estimate how well they know the topic. The random approach helps to minimize any form of bias in the learning of topics.

Learning algorithms require feedback. This is done using a **validation set** where the CNN would make predictions and compare them with the true labels or ground truth. The predictions which errors are made are then fed backwards to the CNN to refine the weights learned, in a so called backwards pass. Formally, this algorithm is called **backpropagation of errors**, and it requires functions in the CNN to be differentiable (almost).

CNNs are too complex to implement from scratch. Today, machine learning practitioners often utilize toolboxes developed such as Caffe, Torch, MatConvNet and Tensor flow for their work.

3.1.3 Inception [14]

We've used the Inception v3 model of the Tensor Flow library. Inception is a huge image classification model with millions of parameters that can differentiate a large number of kinds of images. We only trained the final layer of that network, so training will end in a reasonable amount of time. Inception-v3 is trained for the ImageNet Large Visual Recognition Challenge using the data from 2012 where it reached a top-5 error rate of as low as 3.46%.

Fig 17: Inception v3 model Architecture

We performed transfer learning on Inception model that is we downloaded the pre-trained Inception v3 model (trained on ImageNet Dataset consisting of 1000 classes) , added a new final layer corresponding to the number of categories and then trained the final layer on the dataset.

The kinds of information that make it possible for the model to differentiate among 1,000 classes are also useful for distinguishing other objects. By using this pre-trained network, we are using that information as input to the final classification layer that distinguishes our dataset.

3.2 Recurrent Neural Network (RNN)

Humans don't start their thinking from scratch every second. We don't throw everything away and start thinking from scratch again. Our thoughts have persistence. Traditional neural networks can't do this but Recurrent Neural Networks can. There is information in the sequence itself, and recurrent nets use it to perform tasks that feedforward networks can't. Recurrent networks are distinguished from feedforward networks by the fact that they have feedback loop, ingesting their own outputs moment after moment as input. They're especially useful with sequential data because each neuron or unit can use its internal memory to maintain information about the previous input.

For example in case of a network that is suppose to classify what kind of event is happening at every point in a movie. It requires the network to use its reasoning about previous events in the film to inform later ones. Another example in case of language, "I had washed my house" is much more different than "I had my house washed". This allows the network to gain a deeper understanding of the statement. This is important to note because reading

through a sentence even as a human, you're picking up the context of each word from the words before it.

3.2.1 Recurrent Neural Networks have Loops

A RNN has loops in them that allow information to be carried across neurons while reading in input. In the following diagram, a chunk of Recurrent neural network, **A**, looks at some input x_t and outputs a value h_t. The loop allows information to be passed from one step of the network to the next. The decision a recurrent net reached at time step t-1 affects the decision it will reach one moment later at time step t. So recurrent networks have two sources of input, the present and the recent past, which combine to determine how they respond to new data.

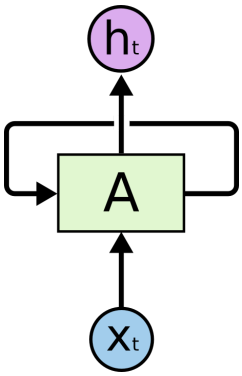

Fig 18: A chunk of Recurrent Neural Network

A recurrent neural network can be thought of as multiple copies of the same network, each passing a message to a successor. Consider what happens if we unroll the loop:

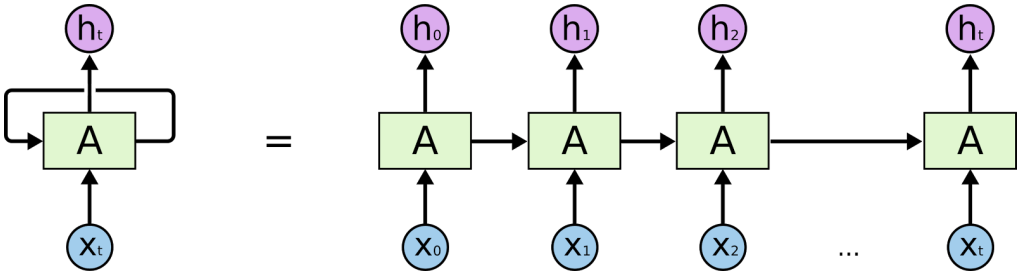

Fig 19: An Unrolled recurrent neural network

This chain-like nature reveals that recurrent neural networks are intimately related to sequences and lists. They're the natural architecture of neural network to use for such data. The sequential information is preserved in the recurrent network's hidden state, which manages to span many time steps as it cascades forward to affect the processing of each new example

3.2.2 How Memory of previous inputs Carried forward

$$\mathbf{h}_t = \phi\left(W\mathbf{x}_t + U\mathbf{h}_{t-1}\right),$$

The hidden state at time step t is h_t. It is a function of the input at the same time step x_t, modified by a weight matrix W, added to the hidden state of the previous time step h_t-1 multiplied by its own hidden-state-to-hidden-state matrix U called transition matrix. The weight matrices are filters that determine how much importance to accord to both the present input and the past hidden state. The error they generate can be used to adjust their weights using Backpropagation through Time (BPTT). The sum of the weight input and hidden state is squashed by the function φ – either a logistic sigmoid function or tanh.

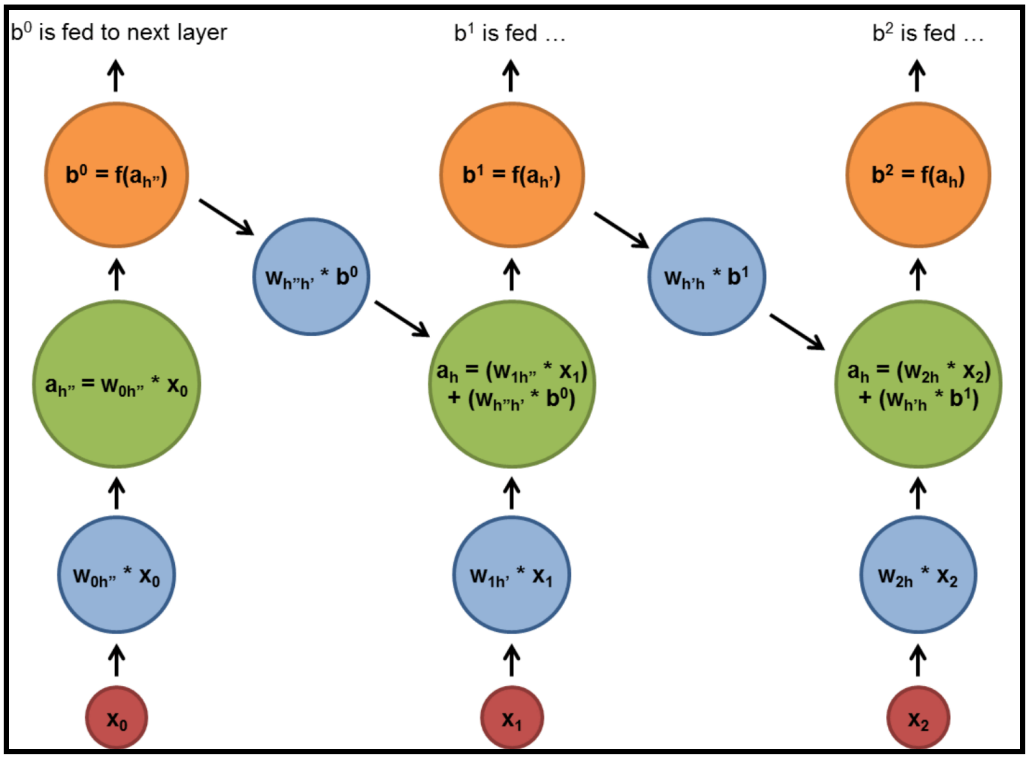

Fig 20: Memory of previous inputs being carried forward

Because this feedback loop occurs at every time step in the series, each hidden state contains traces not only of the previous hidden state, but also of all those that preceded h_t-1 for as long as memory can persist.

3.2.3 Exploding and Vanishing Gradient Problem

In theory, RNNs are absolutely capable of handling long-term dependencies.Sadly, in practice, RNNs don't seem to be able to learn them as explained in [5]. The gradient expresses the change in all weights with regard to the change in error.Since the layers and time steps of deep neural networks relate to each other through multiplication, gradient is susceptible to vanishing or exploding.

3.2.3.1 Vanishing Gradient

The gradients of the network's output with respect to the parameters in the early layers become extremely small. In other words even a large change in the value of parameters for the early layers doesn't have a big effect on the output. Hence the network can't learn the parameter effectively.

This happens because the activation functions (sigmoid or tanh) squash their input into a very small output range in a very nonlinear fashion. For example, sigmoid maps the real number line onto a "small" range of [0, 1]. As a result, there are large regions of the input space which are mapped to an extremely small range. In these regions of the input space, even a large change in the input will produce a small change in the output - hence the gradient is small.

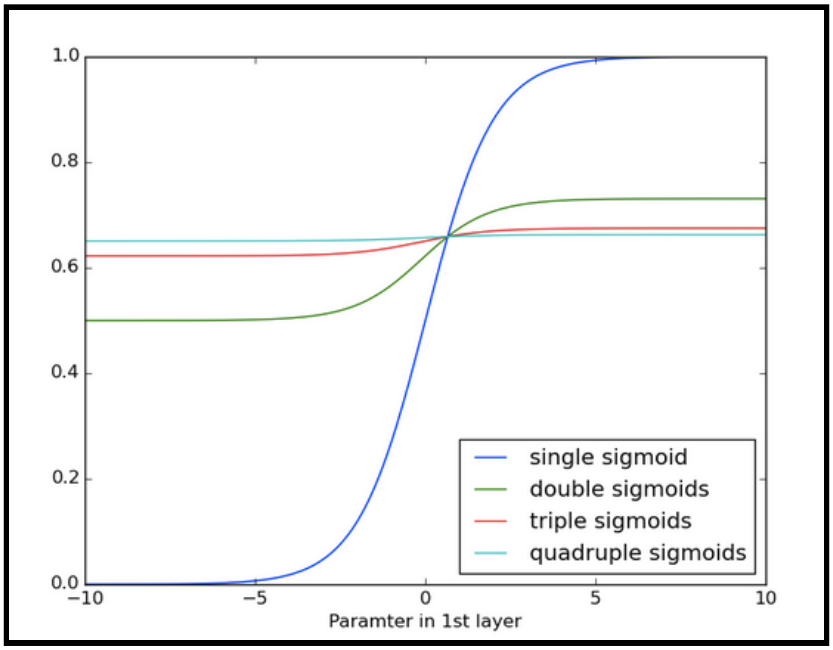

Fig 21: Vanishing Sigmoid (Analogous to Vanishing Gradient)

This becomes much worse when we stack multiple layers of such non-linearities on top of each other. For instance, first layer will map a large input region to a smaller output region, which will be mapped to an even smaller region by the second layer, which will be mapped to an even smaller region by the third layer and so on. As a result, even a large change in the parameters of the first layer doesn't change the output much.

In the Fig _ we can see the effects of applying a sigmoid function over and over again. The data is flattened until, for large stretches, it has no detectable slope. This is analogous to a gradient vanishing as it passes through many layers.

3.2.3.2 Exploding Gradient

Exploding gradients treat every weight as though it were the proverbial butterfly whose flapping wings cause a distant hurricane. Those weights' gradients become saturated on the high end; i.e. they are presumed to be too powerful.

Exploding gradients can be solved relatively easily, because they can be truncated or squashed. Vanishing gradients can become too small for computers to work with or for networks to learn – a harder problem to solve.

3.2.4 Long Short-Term Memory Units (LSTMs)

A variation of of recurrent net with "Long Short-Term Memory Units" LSTMs, was proposed by the German researchers Sepp Hochreiter and Juergen Schmidhuber [6] as a solution to the vanishing gradient problem.

LSTMs help preserve the error that can be backpropagated through time and layers. By maintaining a more constant error, they allow recurrent nets to continue to learn over many time steps (over 1000), thereby opening a channel to link causes and effects remotely.

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn!

3.2.5 Our RNN Model

We have create a RNN model based on LSTMs. The first layer is an input layer used to feed input to the upcoming layers. Its size is determined by the size of the input being fed. Our Model is a **wide network** consisting of single layer of 256 LSTM units. This Layer is followed by a fully connected layer with softmax activation. In Fully Connected every neuron is connected to every neuron of previous layer. The fully connected layer consists of as many neurons as there are categories/classes. Finally a regression layer to apply a regression (linear or logistic) to the provided input. We used adam[8] (Adaptive Moment Estimation) which is a stochastic optimizer, as a gradient descent optimizer to minimize the provided loss function "categorical_crossentropy" (which calculate the errors).

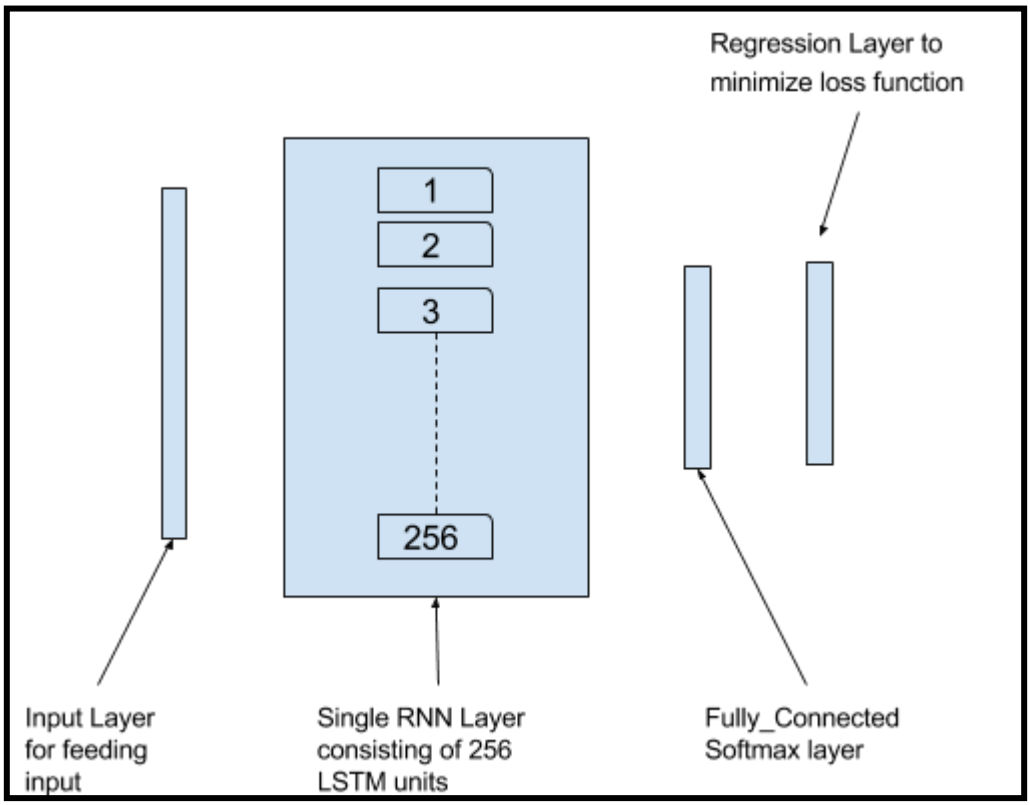

Fig 22: Our RNN Model

We also tried a wider RNN network with 512 LSTM units and another deep RNN network with three layers of 64 LSTM units each. We tested these on a sample of the dataset and found that wide model with 256 LSTM units performed the best and therefore only the wide model was used for training and testing on complete dataset.

4. EXPERIMENTAL DESIGN

We have used two approaches to train the model on the temporal and the spatial features. Both approaches differ by the inputs given to RNN to train it on the temporal features.

4.1 Data Set Used

The data set[7] used for both the approaches consists of Argentinian Sign Language(LSA) Gestures, with around 2300 videos belonging to 46 gestures categories. 10 non-expert subjects executed the 5 repetitions of every gesture thereby producing 50 videos per category or gesture.

Id	Name	Id	Name	Id	Name	Id	Name
1	Son	**13**	Enemy	**25**	Country	**37**	To-Land
2	Food	**14**	Dance	**26**	Red	**38**	Yellow
3	Trap	**15**	Green	**27**	Call	**39**	Give
4	Accept	**16**	Coin	**28**	Run	**40**	Away
5	Opaque	**17**	Where	**29**	Bitter	**41**	Copy
6	Water	**18**	Breakfast	**30**	Map	**42**	Skimmer
7	Colors	**19**	Catch	**31**	Milk	**43**	Sweet-Milk
8	Perfume	**20**	Name	**32**	Uruguay	**44**	Chewing gum
9	Born	**21**	Yogurt	**33**	Barbeque	**45**	Photo
10	Help	**22**	Man	**34**	Spagheti	**46**	Thanks
11	None	**23**	Drawer	**35**	Patience		
12	Deaf	**24**	Bathe	**36**	Rice		

Out of the 50 gestures per category, 75% i.e. 40 were used for training and 25% i.e. 10 were used for testing

4.2 First Approach

In this approach we extracted spatial features for individual frames using inception model (CNN) and temporal features using RNN. Each video (a sequence of frames) was then represented by a sequence of predictions made by CNN for each of the individual frames. This sequence of predictions was given as input to the RNN.

4.2.1 Methodology

- First, we will extract the frames from the multiple video sequences of each gesture.
- After the first step, noise from the frames i.e background, body parts other than hand are removed to extract more relevant features from the frame.
- Frames of the train data are given to the CNN model for training on the spatial features. We have used inception model for this purpose which is a deep neural net.
- Store the train and test frame predictions. We'll use the model obtained in the above step for the prediction of frames.
- The predictions of the train data are now given to the RNN model for training on the temporal features. We have used LSTM model for this purpose.

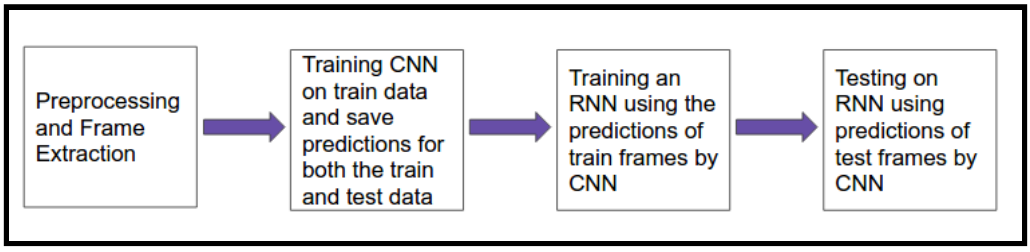

Fig 23

In further subsections of this section, each step of the methodology has been shown diagrammatically for better understanding of that step.

4.2.1.1 Frame Extraction and Background Removal

Each video gesture video is broken down into a sequence of frames. Frames are then processed to remove all the noise from the image that is everything except hands.

The final image consists of grey scale image of hands to avoid color specific learning of the model

Fig 24: One of the Extracted Frames

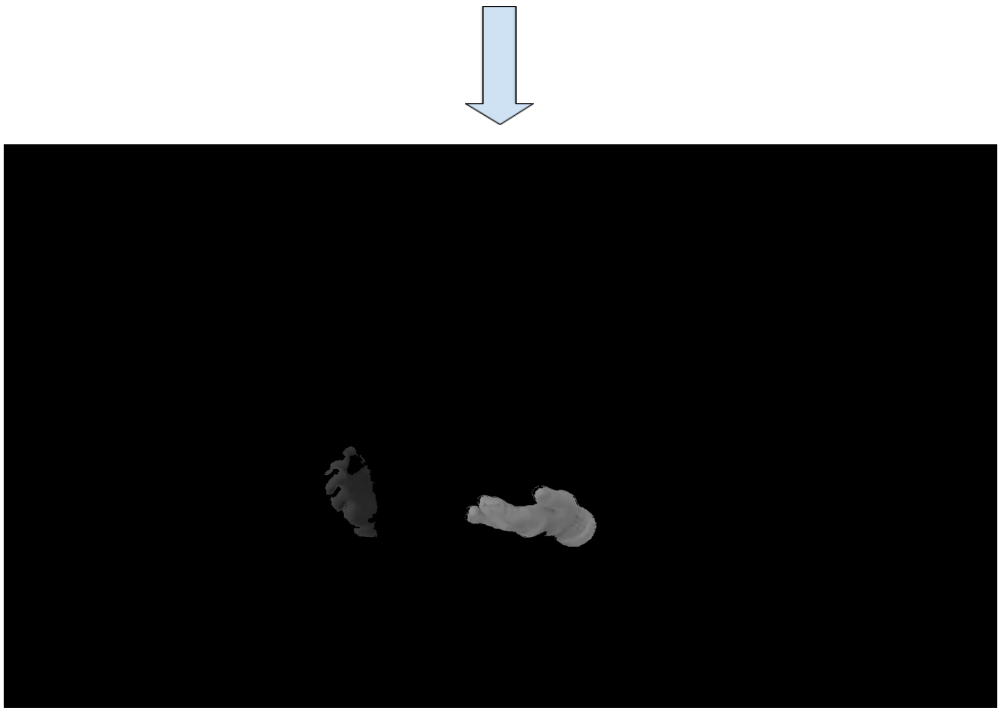

Fig 25: Frame after extracting hands (Background Removal)

4.2.1.2 Train CNN(Spatial Features) and Prediction

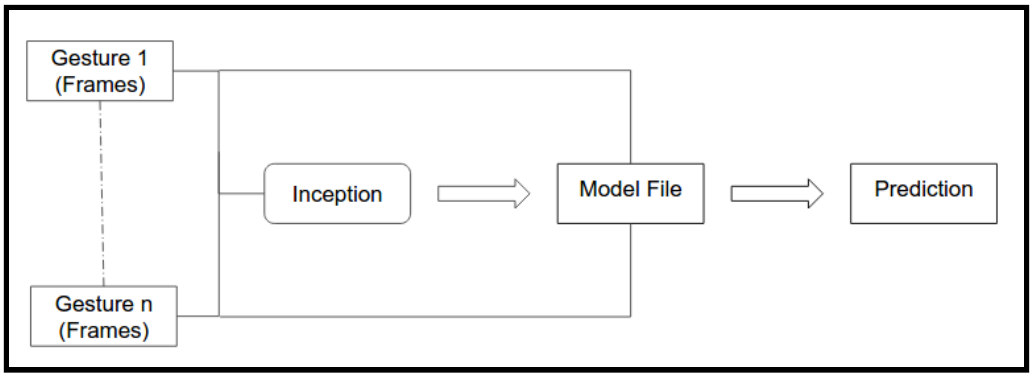

Fig 26

The first row in the below illustration is the video of a gesture Elephant. The second row shows the set of frames extracted from it. The third row shows the sequence of predictions for each frame by CNN after training it.

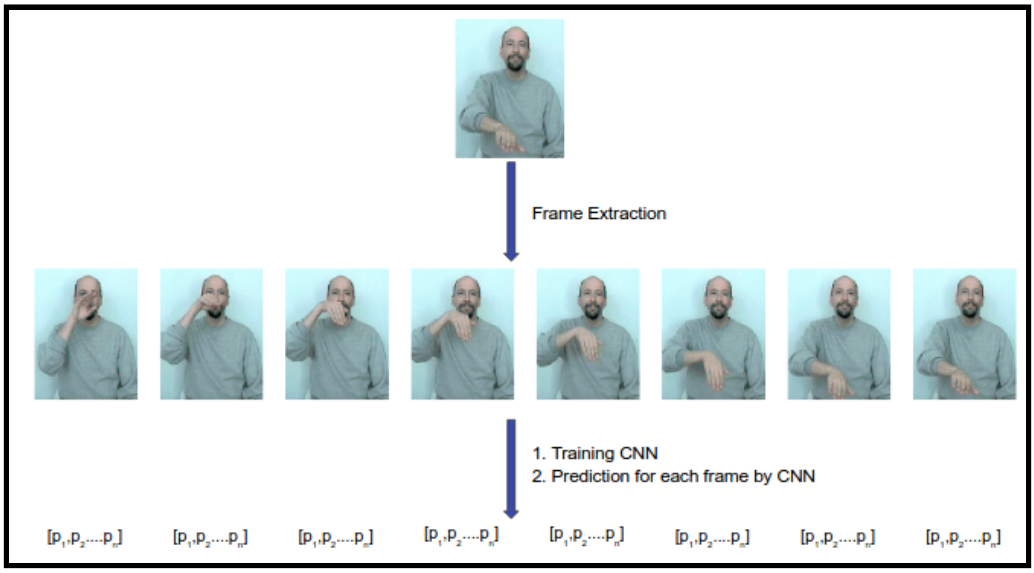

Fig 27

4.2.1.3 Training RNN (Temporal Features)

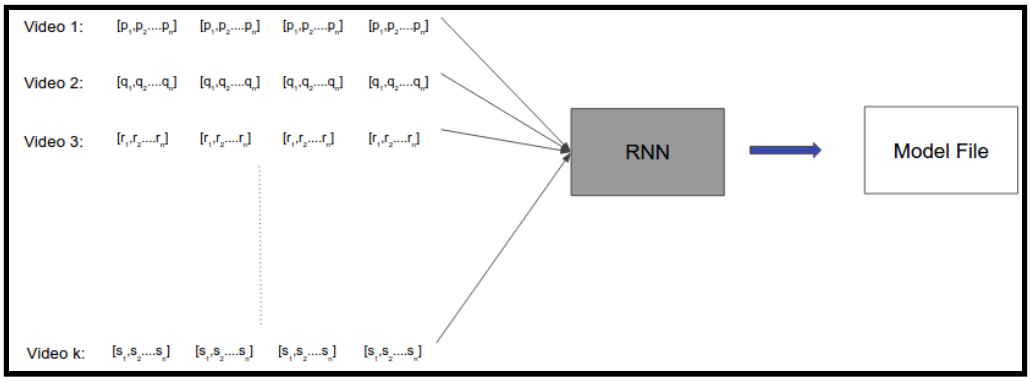

Fig 28

4.2.2 Limitations

The length of a probabilistic prediction by CNN in the sequences of predictions of frames is equal to the number of classes to be classified. In our case it is equal to 46 because we have 46 classes to classify. Therefore the length of feature vector of each frame for the RNN is dependent upon the number of classes to be classified. Less are the number of classes, less would be the length of feature vector for each frame.

4.3 Second Approach

In this approach we have used CNN to train the model on the spatial features and have given the output of the pool layer, before it's made into a prediction, to the RNN. The pool layer gives us a 2048 dimensional vector that represents the convoluted features of the image, but not a class prediction.

Rest of the steps of this approach are same as that of first approach. Both approaches only differ by inputs given to RNN.

5.1 Result of Approach 1

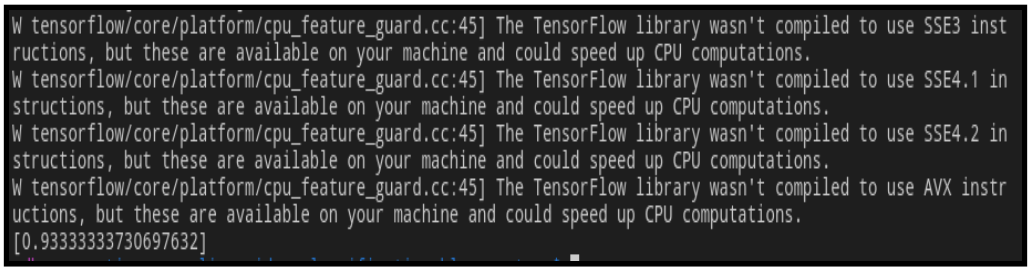

Fig 29

Average accuracy obtained using this approach is 93.3333%.

5.2 Result of Approach 2

Out of the 460 Gestures (10 Per category) used for testing 438 were recognized correctly giving an average accuracy of 95.217%.

Category Wise Accuracy is tabulated and is given on the next page.

ID	Gesture	Accuracy	ID	Gesture	Accuracy
1	Name	100	24	Spagheti	100
2	Yogurt	100	25	Patience	100
3	Accept	90	26	Deaf	90
4	Man	100	27	Eenemy	90
5	Drawer	100	28	Dance	90
6	Bathe	100	29	Rice	100
7	Opaque	90	30	To-Land	100
8	Country	100	31	Yellow	100
9	Water	90	32	Green	90
10	Red	100	33	Give	100
11	Call	100	34	Food	80
12	Colors	90	35	Away	100
13	Run	100	36	Copy	100
14	Bitter	100	37	Coin	90
15	Perfume	90	38	Where	90
16	Map	100	39	Skimmer	100
17	Born	90	40	Trap	80
18	Help	90	41	Sweet-Milk	100
19	Milk	100	42	Breakfast	90
20	None	90	43	Chewing-Gum	100
21	Urugway	100	44	Photo	100
22	Son	80	45	Thanks	100
23	Barbeque	100	46	Catch	90

Fig 30: Accuracy

The second approach provided a better accuracy than the first approach because of the fact that in the first approach the input to the RNN was a sequence of 46 dimensional prediction while in the second approach the RNN was being given a sequence of 2048 dimensional pool layer output. This gave RNN more number of feature points to distinguish among different videos.

7. CONCLUSION AND FUTURE WORK

Hand gestures are a powerful way for human communication, with lots of potential applications in the area of human computer interaction. Vision-based hand gesture recognition techniques have many proven advantages compared with traditional devices. However, hand gesture recognition is a difficult problem and the current work is only a small contribution towards achieving the results needed in the field of sign language gesture recognition. This report presented a vision-based system able to interpret isolated hand gestures from the Argentinian Sign Language(LSA).

Videos are difficult to classify because they contain both the temporal as well as the spatial features. We have used two different models to classify on the spatial and temporal features. CNN was used to classify on the spatial features whereas RNN was used to classify on the temporal features. We obtained an accuracy of 95.217 %. This shows that CNN along with RNN can be successfully used to learn spatial and temporal features and classify Sign Language Gestures.

We have used two approaches to solve our problem and both of the approaches only differ by the inputs given to the RNN as explained in the methodologies above.

We wish to extend our work further in recognising continuous sign language gestures with better accuracy. This method for individual gestures can also be extended for sentence level sign language. Also the current process uses two different models, training inception (CNN) followed by training RNN. For future work one can focus on combining the two models into a single model.

8. REFERENCES

[1] Ronchetti, Franco, Facundo Quiroga, César Armando Estrebou, and Laura Cristina Lanzarini. "Handshape recognition for argentinian sign language using probsom." Journal of Computer Science & Technology 16 (2016).

[2] Singha, Joyeeta, and Karen Das. "Automatic Indian Sign Language Recognition for Continuous Video Sequence." ADBU Journal of Engineering Technology 2, no. 1 (2015).

[3] Tripathi, Kumud, and Neha Baranwal GC Nandi. "Continuous Indian Sign Language Gesture Recognition and Sentence Formation." Procedia Computer Science 54 (2015): 523-531.

[4] Nandy, Anup, Jay Shankar Prasad, Soumik Mondal, Pavan Chakraborty, and Gora Chand Nandi. "Recognition of isolated indian sign language gesture in real time." Information Processing and Management (2010): 102-107.

[5] Bengio, Yoshua, Patrice Simard, and Paolo Frasconi. "Learning long-term dependencies with gradient descent is difficult." IEEE transactions on neural networks 5, no. 2 (1994): 157-166.

[6] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9, no. 8 (1997): 1735-1780.

[7] Ronchetti, Franco, Facundo Quiroga, César Armando Estrebou, Laura Cristina Lanzarini, and Alejandro Rosete. "LSA64: An Argentinian Sign Language Dataset." In XXII Congreso Argentino de Ciencias de la Computación (CACIC 2016). 2016.

[8] Kingma, Diederik, and Jimmy Ba. "Adam: A method for stochastic optimization." arXiv preprint arXiv:1412.6980 (2014).

[9] Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors." Cognitive modeling 5, no. 3 (1988): 1

[10] Hahnloser, Richard HR, Rahul Sarpeshkar, Misha A. Mahowald, Rodney J. Douglas, and H. Sebastian Seung. "Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit." Nature 405, no. 6789 (2000): 947-951.12 Bottou, Léon. "Large-scale machine learning with stochastic gradient descent." In Proceedings of COMPSTAT'2010, pp. 177-186. Physica-Verlag HD, 2010.

[11] Copyright © William Vicars, Sign Language resources at LifePrint.com, http://lifeprint.com/asl101/topics/wallpaper1.htm

[12] https://medium.com/technologymadeeasy/the-best-explanation-of-convolutional-neural-networks-on-the-internet-fbb8b1ad5df8

[13] https://www.quora.com/What-is-an-intuitive-explanation-of-Convolutional-Neural-Networks

[14] Abadi, Martín, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado et al. "Tensorflow: Large-scale machine learning on heterogeneous distributed systems." arXiv preprint arXiv:1603.04467 (2016).

[15] Cooper, Helen, Brian Holt, and Richard Bowden. "Sign language recognition." In Visual Analysis of Humans, pp. 539-562. Springer London, 2011.

[16] Zhang, Chenyang, Xiaodong Yang, and YingLi Tian. "Histogram of 3D facets: A characteristic descriptor for hand gesture recognition." In Automatic Face and Gesture Recognition (FG), 2013 10th IEEE International Conference and Workshops on, pp. 1-8. IEEE, 2013.

[17] Cooper, Helen, Eng-Jon Ong, Nicolas Pugeault, and Richard Bowden. "Sign language recognition using sub-units." Journal of Machine Learning Research 13, no. Jul (2012): 2205-2231.