Indian Institute of Technology Kharagpur
Department of Computer Science and Engineering

American Sign Language Recognizer

Holy Walkamolies
Under the Guidance of Prof. Pabitra Mitra
and
Prof. Jhareshwar Maiti

A report
Submitted in fulfilment for the Term Project in

Machine Learning (CS60050)

Teaching Assistant : **Mr. Anirban Santara**

Submitted on 15th November 2016

Signature: ....

Abstract

American Sign Language is one of the most important tools for communication for people with hearing and speaking disabilities. However, these disabilities make it really difficult for them to perform communication tasks with people who are unable to understand their language. To this end, in our term project, we have tried to create a recognizer for this using DataGloves for reading data and have used Machine Learning to train our classification models. We present a comparison for all the common models that have been tried and used for this purpose and we present our accuracy at detection of words after training our models on character and digits as a comparison across all the implemented models.

Contents

Chapter 1

Introduction

1.1 Motivation and Objectives

To implement a Sign Language Recognizer, we have to keep in mind that a lot of research has already been done about this problem. For our specific solution, the first place to look to would be the existing literature in this and the methodologies they employ with the technology they use on which they support their method and the benefits and pitfalls of any methodology and technology they are trying to implement it by.
The final goal is to make a communication bridge for deaf/mute people to enable them to participate in conversations like everyone else using the technology around them.
The sign language gestures are to be converted to text according to the American Sign Language Standard (ASL) using *flex sensing gloves* and *position* and *angle sensors*. The project was implemented in two phases : **Number** and **Character Recognition**.

The dataset was recorded using the SDT Sensory Gloves :

- Numbers
- Character and words

Data was processed and divided into training and test sets randomly.

- For Number data set :

  Created different classification models and tested for accuracy

- For character data set:

  Created classification models for *individual characters* and used the models for predicting **recorded words** and tested for accuracy

1.2 Literature Review

Since the problem of easing communication with disabled people for people who do not understand their sign language is not a new one, there were a lot of articles, surveys and implementation designs which had already been worked on. The first task in such a case becomes to go through the reading material and figure out if any methodology is helpful to our situation, in terms of the input and testing apparatus.

This paper [1] surveys all the existing Glove based systems and their input and output patters, sensor sets and precision. It also discusses some of the most important data features for a data glove to be used for the task of Sign Language Recognition.

These papers[2][3][4] discuss implementations highly similar to our with a data glove which returns flexion values based on angle measurement or pressure values and a 3-D tracker system to detect orientation. This paper[5] reviews the differences between the most common language symbolism for Sign Language and we used this to overcome our problem of not having an orientation data in our datasets by replacing symbols with non orientation based symbols wherever possible yet keeping it closest to the American Sign Language model.

The current implementation uses the following chart as its basis for the implementation of classification and recognition.

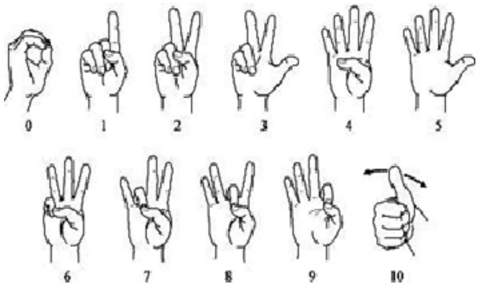

Figure 1.1: ASL Symbols for digits

The implementation of Characters and Words uses the following diagram as its basis. There are few cases where we implemented an orientation based version with tweaks to make the orientation unimportant for the learning model.

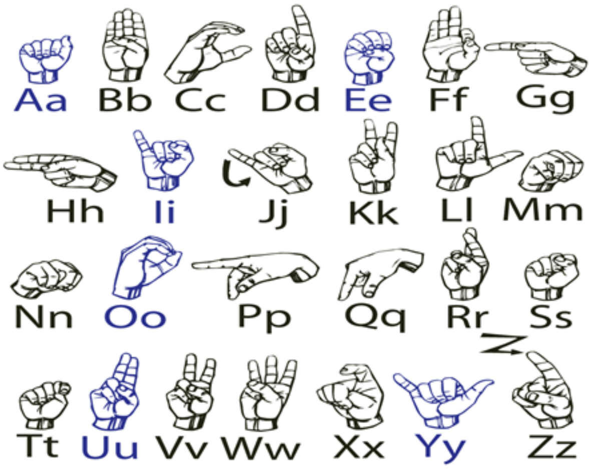

Figure 1.2: ASL Symbols for Characters A - Z

This work[6] implemented an Artificial Neural Network based model and obtained very high using that. That gave us the hint that an ANN based model would help us most with properly implemented weights.

Most papers implementing the model had only used a single implementation using HMM-GMMs or in a few newer cases, Artificial Neural Networks and in all cases, the output precision remained generally high. Their implementation methodology made our plan for implementation easy since we decided to test for the most implemented classification models and ran our dataset on it for learning and subsequent testing.

Finally, this work[7] describes an implementation of a real-time continuous stream based sign to text translation which could be very important if we decide to scale this project upwards and decide to implement a working demonstration of it. It works on using HMMs and GMM to model the transition states and symbolic states of the input to map to the output.

1.3 Technological Support

The entire technological support for our final decision to choose a SDT Sensory Glove with flexion sensors as our input was taken because of the support from the **Virtual Reality Lab** in the **Department of Industrial and Systems Engineering**. All our data collection happened using **Data**

Gloves with an **IMAX 3D Projector** for input.

Each DataGlove has 14 sensors which measure the pressure (or flexion) readings at the rate of **64 packets/s**. It auto calibrates at the beginning of a session and then at every orientation for a sign in the sign language format, both for **Digits** and **Text**, we save an output of about 8-12 seconds to have enough data to run our classification models on.

The data was collected in multiple phases by the team for:

1. Digits *(20 sets of each digit with approximately 600 - 1000 readings in each)*

2. Characters *(20 sets of each character from A to Z with approximately 800-1000 readings in each)*

3. Words *(Set of 50 most commonly used words with 2 sets as a comparison data)*

Chapter 2

Background For Prediction Models

The first step in the Sign Language Recognition, after we have the mappings of text to sign for Digits, Characters and Words is to decide how to create a system that can take this feature set as our input and then predict the symbol given a random feature vector. This is where the Machine Learning comes into play. In this section, we are going to discuss the different Machine Learning models we have run our data on and how each one works and how each differs from the others. The symbol mappings have already been discussed in the Literature Review on this problem. Hence, we need our models to be trained with the given data.

2.1 Decision Trees

Decision Trees are a predictive model used to map observations about an item to conclusions about the item's target value. The goal is to create a model that predicts the value of a target variable based on several input variables. An example is shown in the diagram at right. Each interior node corresponds to one of the input variables; there are edges to children for each of the possible values of that input variable. Each leaf represents a value of the target variable given the values of the input variables represented by the path from the root to the leaf.

A decision tree is a simple representation for classifying examples. For this section, assume that all of the features have finite discrete domains, and there is a single target feature called the classification. Each element of the domain of the classification is called a class. A decision tree or a classification tree is a tree in which each internal (non-leaf) node is labeled with an input feature. The arcs coming from a node labeled with a feature are labeled with each of the possible values of the feature. Each leaf of the tree is labeled with a class or a probability distribution over the classes.

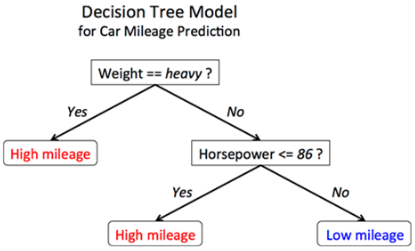

Figure 2.1: A Decison Tree Example

2.1.1 Boosted Trees

It is an optimization on the classical Decision Trees Classifier which builds the model in a stage-wise fashion and generalizes them by allowing optimization of an arbitrary differentiable loss function.

2.1.2 Random Forests

It is another optimization of the Classical Decision Trees where we construct a multitude of decision trees at training time and outputs the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.

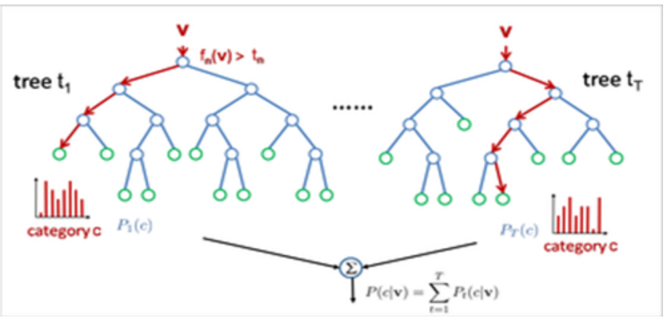

Figure 2.2: Random Trees Visualisation

2.2 Support Vector Machines

Support Vector Machines (SVMs) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.

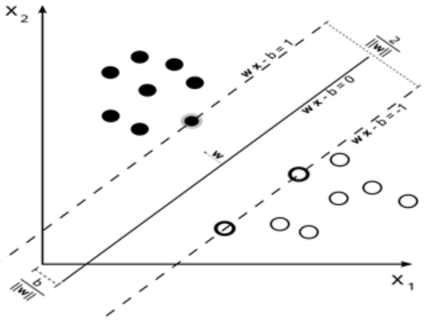

Figure 2.3: An SVM Example in 2 dimensions

In a Support Vector Machines, a data point is viewed as a p-dimensional vector (a list of p numbers), and we want to know whether we can separate such points with a $(p-1)$ dimensional hyperplane. This is called a linear classifier. There are many hyperplanes that might classify the data. One reasonable choice as the best hyperplane is the one that represents the largest separation, or margin, between the two classes. So we choose the hyperplane so that the distance from it to the nearest data point on each side is maximized.

2.3 Artificial Neural Networks

Neural Networks are a computational approach which is based on a large collection of neural units loosely modeling the way the brain solves problems with large clusters of biological neurons connected by axons. Each neural unit is connected with many others, and links can be enforcing or inhibitory in their effect on the activation state of connected neural units. Each individual neural unit may have a summation function which combines the values of all its inputs together. There may be a threshold function or limiting function on each connection and on the unit itself such that it must surpass it before it can propagate to other neurons. These systems are self-learning and trained rather than explicitly programmed and excel in areas where the solution or feature detection is difficult to express in a traditional computer program.

The word *network* in the term 'artificial neural network' refers to the interconnections between the neurons in the different layers of each system. An example system has three layers. The first layer has input neurons which send data via synapses to the second layer of neurons, and then via more synapses to the third layer of output neurons. More complex systems will have more layers of neurons, some having increased layers of input neurons and output neurons. The synapses store parameters called "weights" that manipulate the data in the calculations.

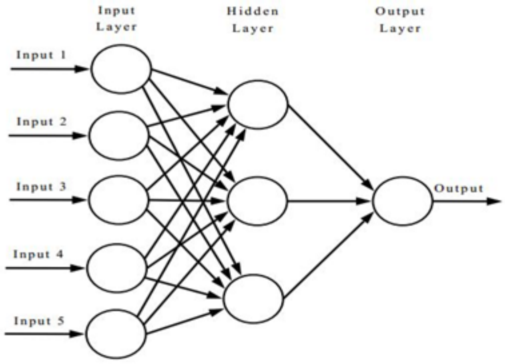

Figure 2.4: AN ASR Example

An ANN is typically defined by three types of parameters:

- The interconnection pattern between the different layers of neurons
- The learning process for updating the weights of the interconnections
- The activation function that converts a neuron's weighted input to its output activation.

Chapter 3

Implementation Work

3.1 Data Collection and Cleaning Methodoloy

SDT Sensory Gloves provide flexion data from fourteen sensors. For every usage run, they need to calibrated against a minimum and maximum flexion so that they don't give random errors in the readings.

Normalization of the readings :

The values for every orientation is according to a basic calibration on wearing the gloves for the first time; i.e, max and min sensor values were recorded for each sensor separately and the recorded values were normalized between 0 and 1 using the calibration results:

$Value = \frac{(Value - min\_calibration)}{(max\_calibration)}$

The Data was recorded by every member of the team for each character/number and the readings were received in a .csv format file :

Input features :

- Fourteen flexion sensor readings, hence X [dimension = 14]

Output :

- Y : Classification between 11 classes; i.e, digit values from 0 - 10 *(For the digit classification)*

3.2 Digit Recognition

From Figure 1.1, we can see clearly that for every unique number, we would be getting a unique vector as our features and that is what we have trained our models on which have all been described and explained in the Background section. For Support Vector Machines, we used two kernels to check which one could map the best to the input.

3.2.1 Model Comparison

Classifier Model	**Precision**
Artificial Neural Networks	99.18
Boosted Tree Classifier	98.95
Random Forest Classifier	97.02
Decision Tree Classifier	95.51
SVM(kernel = linear)	87.69
SVM(kernel = rbf)	85.15

This implementation seems to have a very high accuracy but it is also the case that since this is static input, mapping of input states to output is really easy here. From the table and the corresponding histogram chart, we are able to gather a few relevant points:

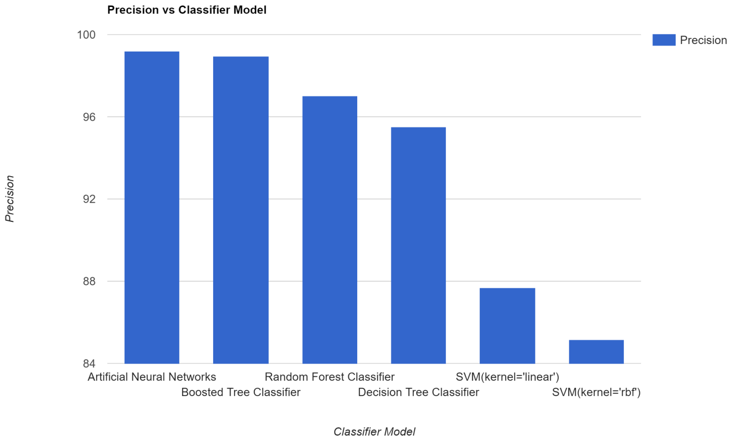

Figure 3.1: Precision vs Classifier Models for Digit Recognition

- The topology of the manifold of gestures is not suitable for RBF kernel
- The linear kernel is more suited than the RBF kernel for this manifold

- Decision trees work better than SVM-Kernel machines because of their ability to model nonlinear decision boundaries efficiently through a piece-wise linear approximation
- A random forest (ensemble of decision trees) generalizes better than a single decision tree
- A deep learning artificial neural network performed the best through data-driven representation learning from a huge amount of data

3.3 Character & Word Recognition

The Character and Word recognition model are based on symbols from Figure 1.2. Every character can be its upper case or lower case version depending on where it occurs in the word. Along with the characters, we also took the data for the most common 50 words used in the vocabulary. This was used as our test set for the model generation. We created a model for character recognition, similar to our model for digit recognition and then, in order to test the algorithm, we followed four steps:

1. Implemented the models on a dataset of sentences which were also recorded manually
2. A continuous stream of characters was obtained
3. The unique characters were separated from the string
4. Then with the help of a dictionary, we modeled the string distribution to obtain the individual words

3.3.1 Model Comparison

Classifier Model	**Precision**
Artificial Neural Networks	99.95
Boosted Tree Classifier	99.92
Random Forest Classifier	99.54
Decision Tree Classifier	98.13
SVM(kernel = rbf)	98.32

Our model for word recognition seems to give a very high accuracy for all cases, even more so than word which was an 11 element output vector compared to our Character set, which is a 37 element vector (11 digits + 26 characters) and it worked on all the models very well.

Form the table for this and the corresponding histogram, we can draw a few conclusions:

- The decision tree classifier seems to have a problem when trying to classify between so many classes of output.
- The deep learning ANN implementation still performed the best because of its data-driven representation learning especially given the high number of output classes.
- A Boosted Decision Tree still gives results comparable to the ANN Classifier because of the same piece-wise linear approximation
- Also, the rbf kernel for the SVM seems to perform better when mapped to character data which was not the case in the digits model. However, it still lags behind in accuracy to ANN and Boosted Trees

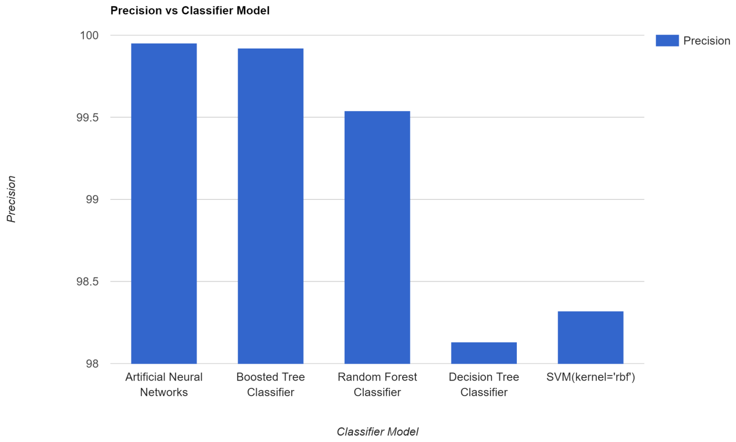

Figure 3.2: Precision vs Classifier Models for Word Prediction

3.4 Future Work

The work implemented in this project gives us a model for American Sign Language recognition for individual words and simple sentences given the flexion data from sensors. Future work on this project would focus on three parts:

1. Extending the model to take care of non static input with continuous variation and marking symbol states and transition states for the input to provide a workable input to the classifier.

2. Extending the model to check for correct sentence formation resulting in detection of even complicated sentences correctly by creating an LSTM based classifier for that.
3. Extending the model to work directly with a stream of input and provide real-time translation of the symbols made with the gloves to the text
4. Finally, creating an open source application encompassing the above tasks with a proper GUI interface

Bibliography

[1] Laura Dipietro, Angelo M Sabatini, and Paolo Dario. A survey of glove-based systems and their applications. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38(4):461–482, 2008.

[2] Neelesh Sarawate, Ming Chan Leu, and CEMİL ÖZ. A real-time american sign language word recognition system based on neural networks and a probabilistic model. *Turkish Journal of Electrical Engineering & Computer Sciences*, 23(Sup. 1):2017–2123, 2015.

[3] J Bukhari, Maryam Rehman, Saman Ishtiaq Malik, Awais M Kamboh, and Ahmad Salman. American sign language translation through sensory glove; signspeak. *Int. J. u-and e-Service, Science and Technology*, 8, 2015.

[4] Wu jiangqin, Gao wen, Song yibo, Liu wei, and Pang bo. A simple sign language recognition system based on data glove. In *Signal Processing Proceedings, 1998. ICSP '98. 1998 Fourth International Conference on*, volume 2, pages 1257–1260 vol.2, 1998.

[5] Neelam K Gilorkar and Manisha M Ingle. A review on feature extraction for indian and american sign language. *IJCSIT) International journal of computer Science and information Technologies*, 5(1):314–318, 2014.

[6] Olga Katzenelson and Solange Karsenty. A sign-to-speech glove. In *workshop IUI2014 on Interacting with Smart Objects, Germany*, 2014.

[7] Kehuang Li, Zhengyu Zhou, and Chin-Hui Lee. Sign transition modeling and a scalable solution to continuous sign language recognition for real-world applications. *ACM Trans. Access. Comput.*, 8(2):7:1–7:23, January 2016.