Project Phase IV Report

Data Mining

CSE 572

Submitted to:

Professor Ayan Banerjee

Ira A. Fulton School of Engineering

Arizona State University

Submitted By:

Abhijith Shreesh (ashreesh@asu.edu)

Abishek Ravichandran (aravic14@asu.edu)

Aditya Chayapathy (achayapa@asu.edu)

Chandni Shrivastava (cshrivas@asu.edu)

Kevin Thomas (kthoma46@asu.edu)

Recap of Previous Phases

This section gives us a brief recap as to what had been carried out in the previous two phases, to give a better understanding of what is being carried out in this phase.

Phase I

A set of common gestures from the American Sign Language were enacted several times in front of a Kinect and the corresponding movements were recorded from the wristband sensors – Accelerometer, Gyroscope, Electromyography (EMG) and Orientation – and stored in a comma separated value file.

Phase II

Sensor data collected from Phase I was used to implement feature extraction and feature selection methods to better represent the data of different gesture classes i.e. the ASL signs for about, and, can, cop, deaf, decide, father, go out, find and hearing, in a reduced feature space.

Feature Extraction

Based on the hand movements over time and our intuition, we came up with the following set of features and respective transformations that would best represent each action:

Sensor/Feature	Transformation
Electromyograph (EMG0R)	Root Mean Square (RMS)
Electromyograph (EMG2R)	Root Mean Square (RMS)
Electromyograph (EMG3R)	Root Mean Square (RMS)
Electromyograph (EMG4R)	Root Mean Square (RMS)
Accelerometer (ALY)	Discrete Wavelet Transform (DWT)
Accelerometer (ARX)	Discrete Wavelet Transform (DWT)
Accelerometer (ARZ)	Standard Deviation (STD)
Orientation (OPR)	Discrete Wavelet Transform (DWT)
Orientation (ORR)	Standard Deviation (STD)
Orientation (OPL)	Standard Deviation (STD)

Feature Selection

From the extracted list of features, our aim was to determine the features that could yield the most variance on the data set. This was accomplished by performing Principal Component Analysis (PCA).

The extracted features for all the actions were combined to form a feature matrix to be fed into PCA. The execution of PCA resulted in the top three principal components capturing 90.62% of the variance in the features. The major contributing features to these principal components were from the sensors EMG and Orientation.

Phase III

A set of Decision Tree, Support Vector Machine and a Neural Network model were implemented on each of ten users' data to predict the ten selected ASL signs, after performing PCA on the features extracted during Phase II.

Based on our preprocessing analysis of each group's data, we concluded that the data collected by the groups - DM05, DM11, DM16, DM20, DM22, DM24, DM26, DM28, DM32 and DM36 best suited our requirements under the given constraints. For each of the groups' sensor data, we combined all the data for all gestures and constructed gesture specific input by labelling a given gesture as 1 and remaining nine as 0. We then randomly selected 60% (random sampling) of the data for training and the remaining 40% was used for testing to report the accuracy metrics such as Precision, Recall and F1 score for each of the three models, which are Decision Trees (fitctree), SVM (fitcsvm) and Neural Network (Neural Network Toolbox).

Phase IV

Goal

The goal of this phase is to develop models to classify the ten selected ASL signs using Decision Tree, Support Vector Machine and a Neural Network model on the features extracted from PCA. The input data in this phase is user independent as opposed to Phase III.

Definitions

This section contains the set of definitions for the concepts we have used to carry out this phase of our project. The definitions are given below:

Decision Tree

A decision support tool which uses a tree-like model of decisions and their possible outcomes. [1] In this phase, we have utilized the ***fitctree*** provided by Matlab. As per Matlab, ***fitctree*** is a fit binary classification decision tree for multiclass classification. [2]

Support Vector Machine

Support Vector Machines (SVM) are supervised learning models with association learning algorithms that analyze data used for classification and regression analysis. [3] In this phase, we have utilized the ***fitcsvm*** provided by Matlab. As per Matlab, ***fitcsvm*** trains or cross-validates a support vector machine (SVM) model for two-class (binary) classification on a low-to-moderate dimensional predictor data set. ***fitcsvm*** supports mapping the predictor data using kernel functions, and supports SMO, ISDA, or L1 soft-margin minimization via quadratic programming for objective-function minimization. [4] We have two different types of SVMs, one class is linear SVM, which separates the points in the space into two categories which are as wide from each other as possible. The other class is the non-linear SVM, which maps the inputs into high dimensional feature space and classifies the points.

Artificial Neural Network

Artificial Neural Networks are computing systems inspired by the biological neural networks like brain networks. It is a collection of artificial neurons, where each neuron can transmit the signal to other

connected neurons, the receiving neurons can process the signal and send signals further. In most cases, the signal from a neuron is mostly a number and the output of each neuron either increases or decreases the strength of that real number. [5] In this phase, we have utilized the Neural Network Toolbox functions provided by Matlab. It helps us create, train, and simulate shallow and deep learning neural networks.

True Positives

The True Positives (TP), refer to the positive tuples that were correctly labeled by the classifier. [7]

True Negatives

The True Negatives (TN), refer to the negative tuples that were correctly labeled by the classifier. [7]

False Positives

The False Positives (FP), refer to the positive tuples that were incorrectly labeled by the classifier. [7]

False Negatives

The False Negatives (FN), refer to the negative tuples that were incorrectly labeled by the classifier.

Precision

The precision (P) is the fraction of records that turn out to be positive in the group the classifier has declared as positive class. Higher precision implies a lower number of false positive errors committed by the classifier. [8] Mathematically it can be represented as: [9]

$$P = \frac{T_p}{T_p + F_p}$$

Recall

Recall computes the fraction of positive examples correctly predicted by the classifier. Higher recall implies very few positive examples are misclassified as the negative class. Recall is equivalent to the true positive rate (TPR). [8] Mathematically it can be represented as: [9]

$$R = \frac{T_p}{T_p + F_n}$$

F1 Score

Precision and Recall can be summarized into another metric, called the F1 Score, which is the harmonic mean of precision and recall. The harmonic mean of two numbers tends to be closer to the smaller of the two numbers. Therefore, a high value of the F1 measure ensures that both precision and recall are high. [8] Mathematically it can be represented as: [9]

$$F1 = 2\frac{P \times R}{P + R}$$

Approach Adopted for Implementation

In this section, we explain the different methods we used in Matlab, to implement the three different models.

Decision Tree (fitctree)

For this phase the default function provided by Matlab was used. Matlab by default sets *PredictorSelection* as *allsplits* by default, and we use this version of decision tree to classify the data. Once the training is complete, the test data is loaded, and the model returns the predicted labels, which are used for the computation of the accuracy metrics.

Support Vector Machine (fitcsvm)

We experimented with three types of kernels: linear, polynomial kernels and RBF kernel. Linear and polynomial kernels didn't perform well for any accuracy measures used, i.e., Precision, Recall and F1 Score. Clearly the gestures are not separable. We found out that the RBF kernel gave us the best results in terms of all the measures we calculate. Hence, we used the RBF kernel.

Neural Network (Neural Network Toolbox)

We used the feedforwardnet implementation provided by Matlab. The network is created using the feedforwardnet() function and the number of hidden layers is set to 15. To train the data, we use the randomly sampled input and the training data labels. We used the net() function and the test data to obtain the predicted results. If the predicted score is less than or equal to 0.5, it is classified as the negative class(0) and for values greater than 0.5, the test sample would be classified as the positive class(1).

Implementation - User Independent Analysis

The data from all the 37 users was agglomerated and grouped based on gestures. We then constructed gesture specific input by labelling a given gesture as 1 and remaining nine as 0. We then randomly selected 60% (random sampling) of the data for training and the remaining 40% was used for testing to report the accuracy metrics such as Precision, Recall and F1 score for each of the three models, which are Decision Trees (fitctree), SVM (fitcsvm) and Neural Network (Neural Network Toolbox).

Results

The following table shows our results for Precision, Recall and F1 score for all the gestures across all users' data:

Gesture	Classifier	Precision	Recall	F1 Score
About	Decision Tree	0.46939	0.37705	0.41818
	Support Vector Machine	0.80315	0.41803	0.54987
	Neural Network	0.65896	0.46721	0.54676
And	Decision Tree	0.44776	0.43062	0.43902
	Support Vector Machine	0.7619	0.076555	0.13913
	Neural Network	0.59722	0.20574	0.30605
Can	Decision Tree	0.5027	0.5082	0.50543
	Support Vector Machine	0.74725	0.37158	0.49635
	Neural Network	0.61538	0.61202	0.6137
Cop	Decision Tree	0.38037	0.30542	0.3388
	Support Vector Machine	0.71429	0.073892	0.13393
	Neural Network	0.65778	0.35845	0.46403
Deaf	Decision Tree	0.44878	0.4381	0.44337

	Support Vector Machine	0.85714	0.25714	0.3956
	Neural Network	0.7	0.23333	0.35
Decide	Decision Tree	0.35714	0.33473	0.34557
	Support Vector Machine	0.90196	0.19247	0.31724
	Neural Network	0.72093	0.25941	0.38154
Father	Decision Tree	0.52153	0.545	0.53301
	Support Vector Machine	0.77907	0.335	0.46853
	Neural Network	0.6087	0.49	0.54294
Go Out	Decision Tree	0.46281	0.35443	0.40143
	Support Vector Machine	1	0.006329	0.012579
	Neural Network	0.53333	0.10127	0.17021
Find	Decision Tree	0.50962	0.52736	0.51834
	Support Vector Machine	0.78	0.19403	0.31076
	Neural Network	0.79452	0.28856	0.42336
Hearing	Decision Tree	0.50505	0.42017	0.45872
	Support Vector Machine	0.88	0.27731	0.42173
	Neural Network	0.72222	0.10924	0.18978

References

- Decision tree Wiki: https://en.wikipedia.org/wiki/Decision_tree
- fitctree Matlab: https://www.mathworks.com/help/stats/fitctree.html
- Support Vector Machine Wiki: https://en.wikipedia.org/wiki/Support_vector_machine
- fitcsvm Matlab: https://www.mathworks.com/help/stats/fitcsvm.html
- Artificial Neural Network Wiki: https://en.wikipedia.org/wiki/Artificial_neural_network
- Neural Network Toolbox: https://www.mathworks.com/products/neuralnetwork.html
- Han, Jiawei, Jian Pei, and Micheline Kamber. Data mining: concepts and techniques. Elsevier, 2011.
- Tan, Pang-Ning, Michael Steinbach, and Vipin Kumar. "Data mining cluster analysis: basic concepts and algorithms." Introduction to data mining (2013).
- http://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html