Gesture Phase Segmentation

Amina Tabassum[*†], **Sai Malleswar Manikonda** [†] **Zhibek Kassymkanova** [†]

College of Engineering, Northeastern University
360 Huntington Avenue
Boston, Massachusetts 02115
tabassum.a@northeastern.edu, manikonda.sa@northeastern.edu , kassymkanova.z@northeastern.edu

Abstract

This paper discusses the gesture phase segmentation employing neural networks. The dataset [2] is comprised of features extracted from 7 videos people gesticulating for purpose of studying gesture phase segmentation problem. It consists of 7 raw and 7 processed files. Raw files are composed of user's position of hands,wrists , head and spine in each frame while processed files are comprised of information regarding user's hands, wrists' velocity and acceleration. The dataset is recorded using depth camera Xbox KinectTM. We preprocessed our data by standardizing and reducing dimension using Principal Component Analysis. We have tried different models : Artificial Neural Network (ANN), Vanilla Recurrent Neural Network (RNN) and Long Short Term Memory network (LSTM) and Gated Recurrent Unit (GRU) RNN. We trained our models using both Stochastic Gradient Descent and Adam optimizer. Best accuracy is achieved using Adam optimizer for all models with 66.6 % for RNN, 71.1 % for LSTM , 81.7 % for GRU RNN and 79.99% for ANN. GRU RNN performs better than all other models in terms of accuracy and processing time.

Introduction

Gesture recognition and segmentation is an important area of research in the domain of human-computer interaction. It has applications in improving interactive experiences and has been challenging for continuous large domains due to limitations such as multiple skin tones and cluttered background. Different methods have been proposed to solve and improve results in this area.
In this paper, we present the approach we took to preprocess the dataset and analyze the data. We also discuss the models we used and the performance achieved by each model. Our best-performing model achieved an accuracy of 82percent using GRU.

This paper discusses the problem of gesture phase segmentation employing neural networks.

The dataset used in this study consists of features extracted from 7 videos of people gesticulating, recorded using the depth camera Xbox KinectTM. The dataset includes both raw and processed files. Raw files contain information on the user's position of hands, wrists, head, and spine in each frame, while processed files include information regarding the user's hands' and wrists' velocity and acceleration.

We have used three different neural network models, namely Artificial Neural Network (ANN) Figure 1, Vanilla Recurrent Neural Network (RNN), and Long Short-Term Memory Network (LSTM) as shown in Figure 2, and trained them using both Stochastic Gradient Descent and Adam optimizer. Our aim was to achieve high accuracy in predicting the phase of the gesture with five possible classes: rest, preparation, stroke, hold, and retraction.

Artificial Neural Networks (ANNs) are a type of machine learning algorithm inspired by the structure and function of the human brain. They consist of layers of interconnected nodes, or neurons, that process input data and generate output. ANNs learn from examples and adjust the strengths of connections between neurons to improve their ability to make accurate predictions or classifications. They have been successfully used in a wide range of applications, including image and speech recognition, natural language processing, and predictive modeling.

Vanilla RNN (Recurrent Neural Network) is the simplest form of RNN that processes sequential data by taking an input and outputting a hidden state, which is then passed to the next time step. Recurrent Neural Networks (RNNs) are a type of neural network that is specifically designed for processing sequential data, such as time-series or natural language. Unlike traditional feedforward neural networks, RNNs have loops that allow information to be passed from one step of the sequence to the next. This allows RNNs to maintain a memory of past inputs, which is useful for tasks such as language modeling or speech recognition. RNNs are often implemented using specialized cells, such as Long Short-Term Memory (LSTM) cells or Gated Recurrent Units (GRUs), that help to overcome the problem of vanishing gradients during training.

Long Short-Term Memory (LSTM) is a type of Recurrent Neural Network (RNN) that is designed to handle the vanishing gradient problem during training. It uses specialized memory cells that can store information over long periods of time and selectively forget or retain information. LSTM cells have three gates (input, forget, and output) that control the

[*]These authors contributed equally.
[†]GitHub: https://https://github.com/AminaTabassum/Gesture-Phase-Segmentation

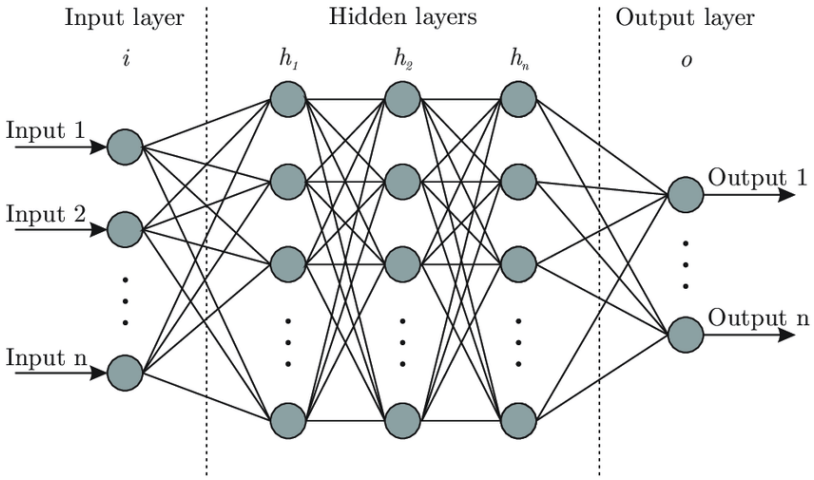

Figure 1: Artificial Neural Network

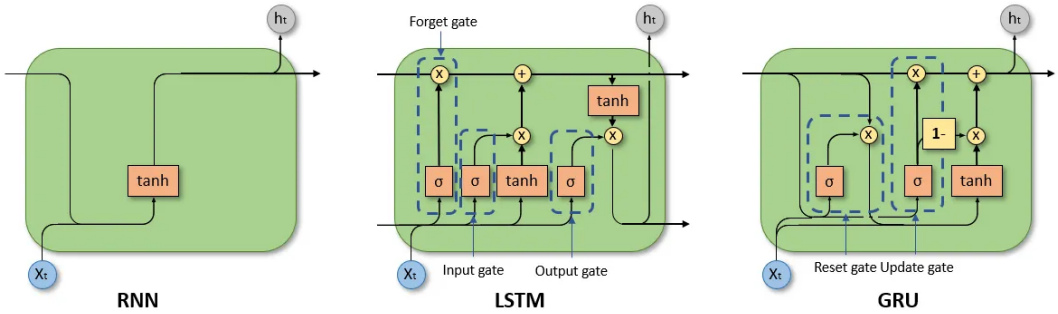

Figure 2: RNN,LSTM and GRU

flow of information and enable the network to learn long-term dependencies in sequential data. LSTMs have been successfully used in a wide range of applications, including natural language processing, speech recognition, and time-series forecasting.

Gated Recurrent Units (GRUs) are a type of Recurrent Neural Network (RNN) that are similar to Long Short-Term Memory (LSTM) networks, but with a simpler architecture. Like LSTMs, GRUs have gating mechanisms that allow them to selectively forget or retain information, but they have only two gates (reset and update) compared to three in LSTMs. GRUs have been shown to be effective in tasks such as language modeling, speech recognition, and image captioning, and they can be easier to train and computationally less expensive than LSTMs.

Background

Gesture recognition and segmentation is one of the widely solved problems in human computer interaction domain for improving interactive experiences.It is challenging for continuous large domains because of multiple skin tones , cluttered background and other limitations. Different methods have been proposed to solve and improve results. [1] discusses vision-based gesture recognition and unified spatiotemporal gesture segmentation. Three major contributions are spatiotemporal matching which can identify multiple hands even in cluttered background, classifier based pruning framework which removes poor matches of gesture models in early stages of training and subgesture reasoning which can segregate between different gestures. The performance of proposed algorithm was evaluated for American sign language signers. [3] proposes two stream faster R-CNN to segment continuous gestures into isolated gestures based on hand oriented spatiotemporal features. Initially, the extracted hand oriented features from RGB and depth are fused to train SVM classifier. The proposed method outperforms for continuous large scale gesture recognition challenges. [4] discusses gesture segmentation using support vector machines focusing on solving gesture unit segmentation problem which consists of segmenting rest position from gestures in a video frame. Point of interest, position of frame of interest, time displacement, time domain and frequency domain features are used for classifier. The best results were achieved by training SVM with simple windowed datapoint; window with 46 frames, using hands as points of interest and velocity features.

Approach

Dataset

Dataset [2] consists of 7 processed and 7 unprocessed files. Each processed file consists of 18 numerical columns and each unprocessed file consists of 31 numerical columns. Unprocessed file comprises of the user's position of their hands, wrists, head, and spine in each frame, and a processed file that contains the user's hands' and wrists' velocity and acceleration. Depth camera Xbox KinectTM is used to record dataset.

Our target variable is phase of gesture with 5 possible classes: rest, preparation, stroke, hold and retraction. Files are separated by users A, B , C and video frames. To proceed with model application we decided to concatenate all these files together: firstly, merge each processed with unprocessed file, and secondly, stack all files one-by-one. We have encountered a difficulty that each processed file had a discrepancy with corresponding unprocessed file of 4 rows. Therefore, before proceeding with data processing, we merged files by classes to detect discrepancies and deleted unique 4 records in each unprocessed file. Following that, we merged all files and our final dataset consists of 51 numerical attribute and over 9000 records.

Before applying any model, we performed exploratory data analysis on our dataset. We have observed that there are no missing values, no strong correlation among features and no outliers. In addition to that, we separated our dataset on training and testing with .30 train size and standardized our numerical features with training set parameters.We have tried using both Stochastic Gradient Descent with optimizer at learning rate of 0.01 and momentum 0.9. But,accuracy was low. For this model, we have used Adam optimizer.

Neural Network Models

Artificial Neural Network Artificial Neural Networks are basic algorithm modeled after the functioning of human brain .Artificial neurons are fundamental unit of ANNs which receive input , process it using weights and biases, produce an output which is them fed to the next layer as shown in Figure 1 Activation functions such as ReLU or sigmoid function are used to incorporate non-linear real world data.

We are using ANNs because they can learn non-linear relationships in data, scalable, can be easily generalized for new data and can perform computations in parallel.Our dataset has temporal features extracted from videos and hence ANN is suitable for it.

Initially, the dataset was pre-processed as mentioned in data subsection . Dimensionality reduction was performed using Principal Component Analysis. Our dataset is comprised of 52 attributes which were reduced to 50 by normalizing and rescaling it. These dimensions were further reduced using principal component analysis. The model summary is shown in Figure 3

We have used L2 regularizer to add a penalty term to avoid overfitting. Moreover, we have used dropout rate of 0.3 and 0.1 to avoid overfitting and improve capacity of model.

Recurrent Neural Networks Recurrent Neural Networks is a an extension of Artificial Neural Networks model, which assumes that layers are interrelated with each other as shown in Figure 2. This feature of RNN makes it a suitable algorithm for such complex data as time series data. Our dataset consists of video timestamps, which makes it suitable for RNN model. There are several variations of Recurrent Neural Networks model, which include 'Vanilla' RNN, Long Short-Term Memory and Gated Relation Neural Networks as shown in

'Vanilla' RNN consists only of layers, interconnected with each other, and the main difference of this algorithm from LSTM RNN and GRU RNN is absence of gates.

Long Short-Term Memory is a variety of RNN model that consists of gating mechanism which allows model to remember or forget information for future processing. It consists of three gates: input gate, forget gate and output gate; and they control the flow of information in the model. Input gate focuses on how much information to add to cell state, forget gate focuses on how much information to delete and output gate focuses on how much information to pass to output.

Gated Recurrent Network is also a gating mechanism as LSTM, but it consists only of two gates: reset gate and update gate. Reset gate controls the amount of information to be deleted and update gate controls the amount of new information to be added. Due to lesser amount of gates, GRU generally processes information faster than LSTM.

In our project we have implemented all three variations of RNN and made used different optimization techniques: Adaptive Moment Estimation and Stochastic Gradient Descent.

Stochastic Gradient Descent was used with learning rate of 0.1 and momentum 0.9. We carefully observed learning rate on different epochs to not let optimizer overshooting local minima.

Results

Dataset

We have discussed the preprocessing of dataset in approach section. We used Recurrent Neural Networks and Artificial Neural Network to train and evaluate our dataset. Evaluation metrics is accuracy of trained model on test data.

Figure 4 displays correlation heatmap between phases. Red color demonstrates strong correlation which means if one of these variables increases, other also increases and vice versa.Black color indicates strong negative correlation and white indicates strong positive relation between two variables. It can be observed from the heatmap that almost all the variables are not correlated and hence no need to preprocess data. We decided to perform Principal Component Analysis on dataset to reduce the dimensionality of data, since the number of records on our dataset is not large (about 9000 records), while we have 50 features, which can negatively affect the processing time. The relation between variance and PCA is shown in Figure 5. We standardized this dataset and then trained ANN and RNN on processed data. RNN appeared to be performing the best on original dataset

```
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 dense_566 (Dense)            (None, 512)               3072

 dropout_201 (Dropout)        (None, 512)               0

 dense_567 (Dense)            (None, 256)               131328

 dropout_202 (Dropout)        (None, 256)               0

 dense_568 (Dense)            (None, 128)               32896

 dropout_203 (Dropout)        (None, 128)               0

 dense_569 (Dense)            (None, 256)               33024

 dense_570 (Dense)            (None, 128)               32896

 dense_571 (Dense)            (None, 64)                8256

 dropout_204 (Dropout)        (None, 64)                0

 dense_572 (Dense)            (None, 64)                4160
...
Total params: 248,325
Trainable params: 248,325
Non-trainable params: 0
_________________________________________________________________
```

Figure 3: Artificial Neural Network summary

with no reduced features and ANN was performing the best on 5 Principal Components. Therefore, RNN was applied on standardized original 50 features and ANN was applied on standardized 5 components.

Figure 6 is a boxplot demonstrating information about median, range and outliers of dataset. As can be observed from the graph, feature have different scale, therefore standardization of features should be implemented before modeling.

Experiments and performance evaluation

Artificial Neural Networks As mentioned in approach section , we have used ANN's to our dataset. We have tried different hyperparameters tuning parameters. Initially, the model performance was very low 35 % with 50 attributes. We have improved it by reducing the number of dimension (principal components) to 5.Table 1 , Table 2, Table 3 and Table 4 demonstrate the results and tuning of different hyperparameters.

Recurrent Neural Networks As we have mentioned in approach section, we decided to apply Recurrent Neural Networks model to our dataset, as the dataset consists of timestamps making it suitable for this model. Before applying RNN model, we preprocessed data by reshaping features to 3D array. Additionally, we reshaped our target variable by converting it to 5 separate columns corresponding to 5 classes with binary outcomes, to satisfy categorical cross-entropy loss. We decided to implement 'Vanilla' Recurrent Neural Networks, Long Short-Term Memory Neural Networks and Gated Relation Neural Networks. Moreover, we decided to test all these models with 2 different optimizers: Adam optimizer and SGD optimizer. Upper layer is using softmax activation function to suit for multiclass classification problem. To reduce overfitting dropout rate of 0.2 was implemented between layers and L2 regu-

Hyperparameter	**Value**
Epoch	100
Batch Size	16
L2 Regularizer	0.01
No of principal components	5
Activation function	ReLU
Dropout	0.01
Adam Optimizer	0.01

Table 1: Hyperparameters for ANN for 79.99 % Accuracy

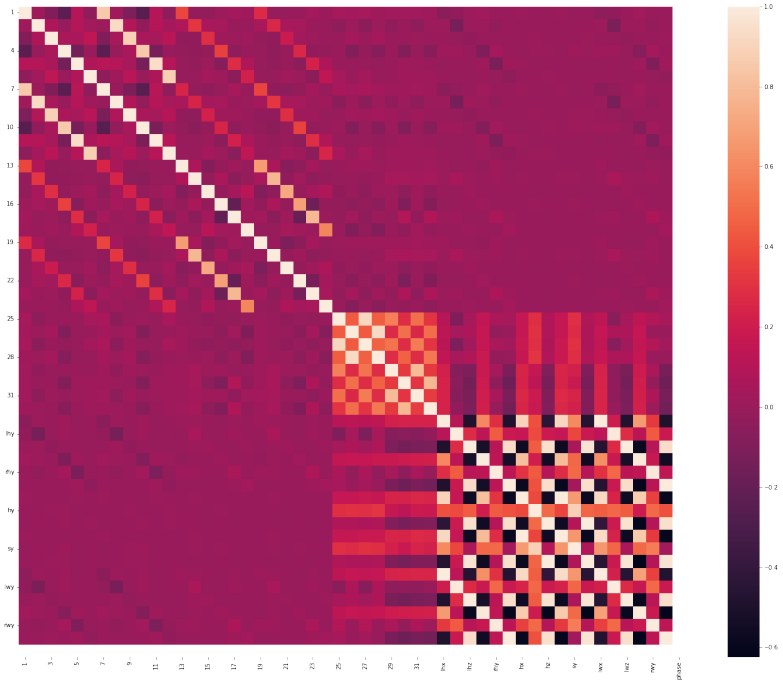

Figure 4: Correlation Heatmap

Hyperparameter	Value
Epoch	300
Batch Size	32
L2 Regularizer	0.01
No of principal components	5
Activation function	ReLU
Dropout	0.02
Adam Optimizer	0.01

Table 2: Hyperparameters for ANN for 74.43 % Accuracy

Hyperparameter	Value
Epoch	100
Batch Size	8
L2 Regularizer	0.01
No of principal components	5
Activation function	ReLU
Dropout	0.02
Adam Optimizer	0.01

Table 3: Hyperparameters for ANN for 65.55 % Accuracy

larizer was applied to the first layer. LSTM and GRU are more advanced versions of RNN, therefore they take much more time to run than Vanilla RNN. Due to this fact, a large amount of epoch could not be applied to LSTM (it is the longest model to process) and GRU. The performance could have been better if number of epochs was larger. Table 6,Table 7,Table 8,Table 9,Table 10,Table 11 demonstrate different hyper-parameter tuning of RNN's.

Discussion

In this project, we explored implementation of neural networks to segment gesture phases from videos of people performing five different gestures. Correlation analysis helps us understand relationship between two continuous variables.

Figure 7 demonstrates the comparison of performance of all models.Table 5 summarizes the model's performance for different optimizer combinations.Figure 1 demonstrates the correlation between two variables. When two variables are correlated, it means that when there is change in one variable, there is change in other variable as well. After observing correlation heatmap, we normalized data and then performed principal component analysis to reduced the number of variables, processing time and hence improve results Figure 5. We implemented different neural networks.

It can be observed from above tables that GRU RNN network outperforms all other models. Simpler ANN acheieves maximum 79.99 % accuracy as mentioned in Table 1.

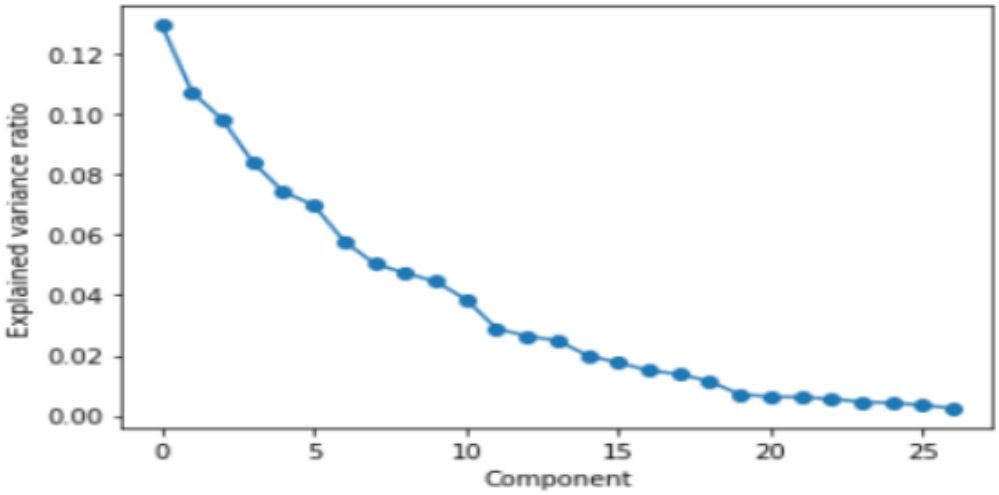

Figure 5: Variance Ratio and PCA

Hyperparameter	Value
Epoch	300
Batch Size	16
L2 Regularizer	0.01
No of principal components	5
Activation function	ReLU
Dropout	0.02
SGD Optimizer	0.01 , 0.9

Table 4: Hyperparameters for ANN for 64.33 % Accuracy

Model	Optimizer	Test Accuracy %
ANN	ADAM	79.99
ANN	SGD	65.55
'Vanilla' RNN	ADAM	66.60
'Vanilla' RNN	SGD	39.90
LSTM RNN	ADAM	70.40
LSTM RNN	SGD	71.10
GRU RNN	ADAM	75.20
GRU RNN	SGD	81.70

Table 5: Summary of models' performance

RNN's accuracy with SGD was very low. GRU RNN and LSTM perform better than others because these two models are able to learn even when signal gets smaller and smaller propogating through time. These two models are able to deal with vanishing gradient problem. GRU RNN and LSTM are similar except that GRU has two gates whereas LSTM has three gates and hence improved performance.

Conclusion

Having observed the performance of Artificial Neural Networks and Recurrent Neural Networks, we came to a conclusion that RNN performs slightly better than ANN and, out of variety of RNN models, GRU works the best. We consider this as expected discovery, since our dataset consists of timesteps and RNN is focused on fitting time-series data. Additionally, we have observed how different optimizers and hyperparameters may influence the performance of models and the difference in the time efficiency between models. Even though GRU is the best performing model, it takes approximately the same time to process twice more epochs than LSTM, which makes GRU more efficient model to implement for larger datasets. Considering that, LSTM might have performed better for larger number of epochs, but it will not be time efficient.

Hyperparameter	Value
Epoch	300
Batch Size	32
L2 Regularizer	0.01
No of features	50
Activation function	ReLU, Softmax
Dropout	0.02
Adam Optimizer	0.01

Table 6: Hyperparameters for 'Vanilla' RNN for 66.6 % Accuracy

Hyperparameter	Value
Epoch	100
Batch Size	32
L2 Regularizer	0.01
No of features	50
Activation function	ReLU, Softmax
Dropout	0.02
SGD Optimizer	0.01, 0.09

Table 7: Hyperparameters for 'Vanilla' RNN for 39.9 % Accuracy

Hyperparameter	Value
Epoch	50
Batch Size	32
L2 Regularizer	0.01
No of features	50
Activation function	ReLU, Softmax
Dropout	0.02
Adam Optimizer	0.01

Table 8: Hyperparameters for LSTM RNN for 70.4 % Accuracy

Hyperparameter	Value
Epoch	50
Batch Size	132
L2 Regularizer	0.01
No of features	50
Activation function	ReLU, Softmax
Dropout	0.02
SGD Optimizer	0.01 , 0.9

Table 9: Hyperparameters for LSTM RNN for 71.1 % Accuracy

Hyperparameter	Value
Epoch	100
Batch Size	32
L2 Regularizer	0.01
No of features	50
Activation function	ReLU, Softmax
Dropout	0.2
Adam Optimizer	0.01

Table 10: Hyperparameters for GRU RNN for 75.2 % Accuracy

Hyperparameter	Value
Epoch	200
Batch Size	16
L2 Regularizer	0.01
No of features	50
Activation function	ReLU, Softmax
Dropout	0.02
SGD Optimizer	0.01 , 0.9

Table 11: Hyperparameters for GRU RNN for 81.8% Accuracy

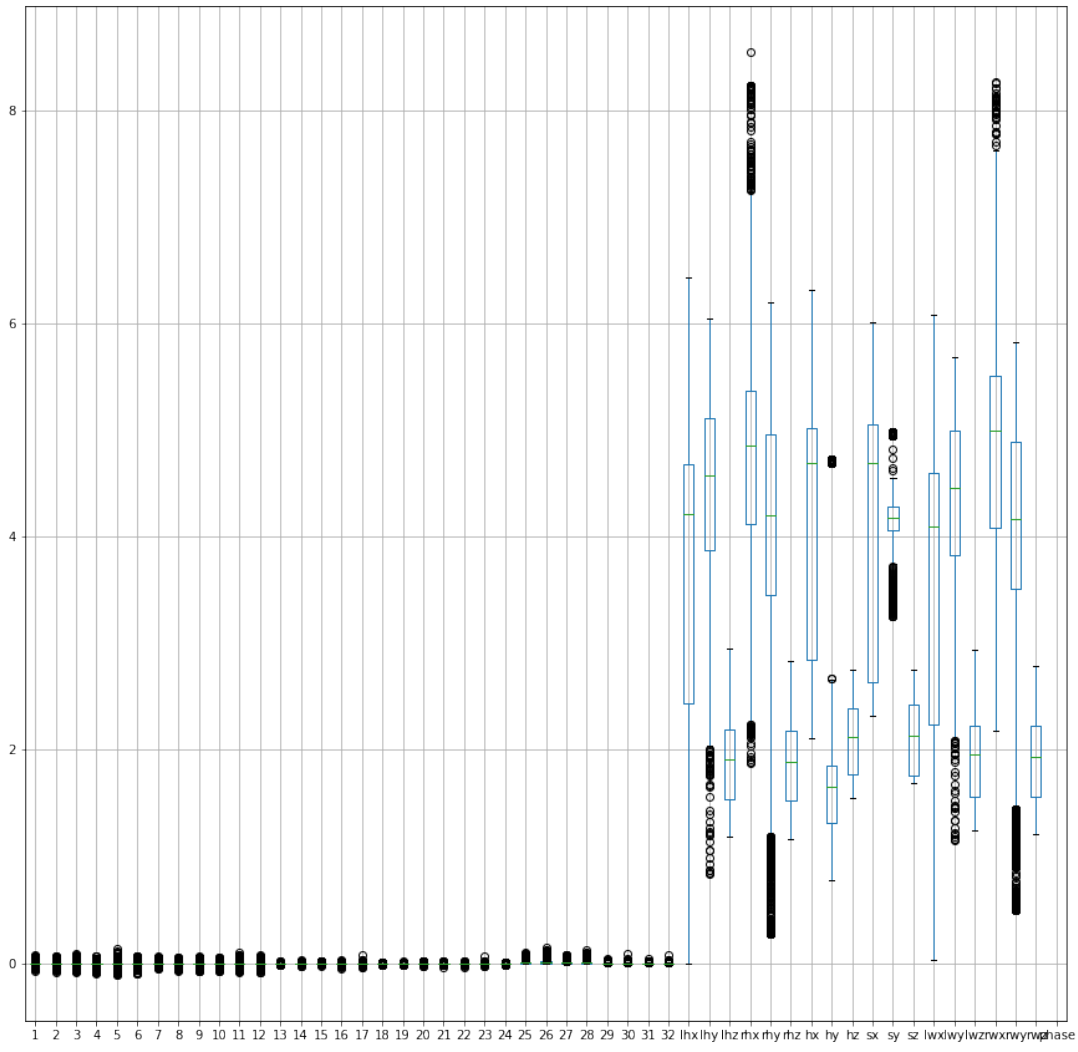

Figure 6: Boxplot

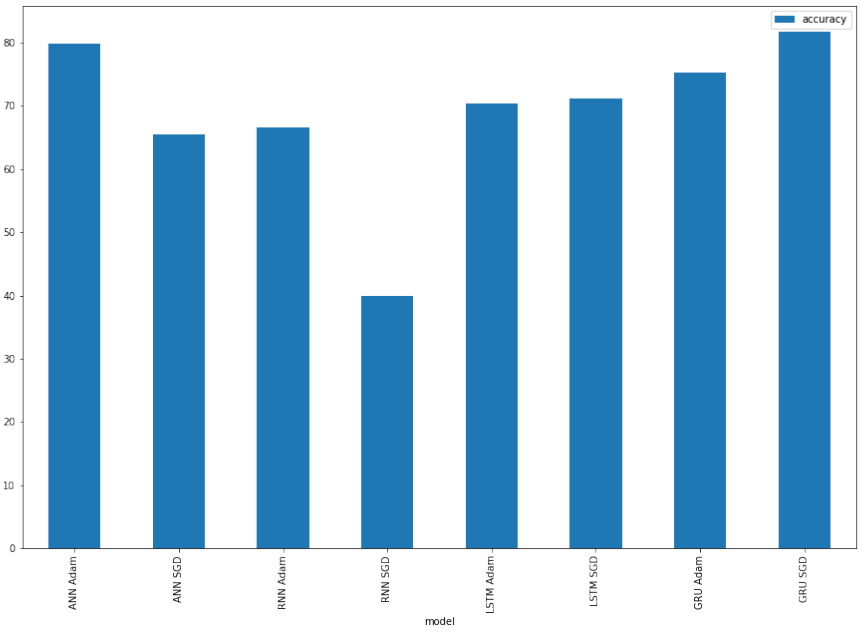

Figure 7: Performance Comparison of Models

References

[1] Alon, J.; Athitsos, V.; Yuan, Q.; and Sclaroff, S. 2009. A Unified Framework for Gesture Recognition and Spatiotemporal Gesture Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(9): 1685–1699.

[2] Dua, D.; and Graff, C. 2017. UCI Machine Learning Repository.

[3] Liu, Z.; Chai, X.; Liu, Z.; and Chen, X. 2017. Continuous Gesture Recognition With Hand-Oriented Spatiotemporal Feature. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*.

[4] Madeo, R. C. B.; Lima, C. A. M.; and Peres, S. M. 2013. Gesture unit segmentation using support vector machines: segmenting gestures from rest positions. In *ACM Symposium on Applied Computing*.