The Identification and Classification of the American Sign Language Fingerspelling Alphabet Using Leap Motion

SUBMITTED BY: Emily Feng
COSUBMITTED BY: CHEN Yan, XU Jinghui
TEACHER: ZHANG Linxuan

Tsinghua University, Beijing, China
August 10, 2020

Abstract. Communication with individuals with hearing disabilities in the hearing world is often difficult without interpreters. To help these individuals to be easily understood, this project focuses on using leap motion to translate finger-spelled signs into written English. The project consists of building a gesture library of the coordinate vectors of bones on the fingers and classification of the hand transformation from an open palm position into the gestures of 26 fingerspelling letters in the American Sign Language (ASL) alphabet. Then, using Support Vector Classification (SVC), Extreme Gradient Boosting (XGBoost), and Artificial Neural Networks (ANNs) algorithms to classify and translate these signs. With a test set recorded in the gesture library different from the learning set, this gesture recognition method consisting of the coordinates of the bones has achieved an accuracy of 98% with the XGBoost classification. In the future, rather than collecting more data, coordinate vector calculation and image recognition can complement each other so that the accuracy could be further increased, and more fit for public application for the benefit of individuals with hearing disabilities.

Keywords: Leap Motion, Coordinate Vector Calculation, Support Vector Classification, XGBoost, Artificial Neural Network

Table of Contents

1 Introduction

Sign-language is a language based on the completion of a series of gestures in order to represent words or thoughts used mainly by deaf and mute people around the world. American Sign Language (ASL) is one form of sign language that is used primarily used in North America. The World Health Organization estimates that there are about 466 million people[1] in the world with hearing disabilities. Although bearing a large amount in population, people with hearing disabilities have a hard time communicating with regular people due to inconsistencies in language and other issues. However, advances in hand data acquisition and classification in gesture recognition has made the translation sign language a probability, which would facilitate the average life of people with hearing disabilities. In the year 2013, leap motion, a motion sensing device, was developed by the Ultraleap company[2]. The device collects hand gesture information and stores them on a computer device. Due to its relatively low price and high recognition rate, the leap motion device was chosen to be the base information collector of this project.

Research on gesture recognition started in the late 20th century. In 1983, Gary Grimes developed the "Digital Entry Data Glove", which consists of multiple sensors mounted on a glove, and in 1993, Fels and Hinton used VPL (Virtual Programming Language) DataGlove II with a Polhemus tracker deployed to gain access to gesture information and translate ASL[3]. Many researchers in the sign language field study sign language by breaking it up into different aspects by characteristics. In 1976, American Sign Language advocate William C. Stokoe Jr.[4] classified sign language by characteristics in motion, position, and gesture. The decomposition of sign language by language is conducive to the formation of a large number of sign language vocabulary through the arrangement of a certain number of hand shapes, movement trajectories, and other factors, and also helps to use two-dimensional images to sign language in three-dimensional space, which is also very helpful to daily sign language teaching and promotion. In 1991, Murakami and Taguchi used recurrent neural networks in hand gesture recognition[5]. They were able to train 42 gestures in the Japanese sign language alphabet and using the VPL data glove with a recognition rate of 98%. Also, an attempt at image recognition was made by Charaphayan and Marble to recognize ASL signs, and 27 out of 31[6] ASL signs were correctly recognized by their system.

From these attempts at sign language translation, many employed image recognition, hand position tracking, and a small vocabulary to work with. Therefore, with the new kinetic leap motion tracker, the position and motion of the hand could be determined. Combined with a small vocabulary from the ASL fingerspelling alphabet, this project aims to use machine learning in order to correctly recognize and translate the gestures.

The purpose of this research is to conduct and develop a system using the kinetic sensor Leap Motion in order to achieve the translation of 26 fingerspelling signs of the ASL alphabet (A-Z) into common English. This project is primarily achieved by researching the practicably of motion data recognition and collection of Leap Motion

controller and its benefits, utilizing machine learning algorithms and connecting them to Leap Motion motion-sensing and the abstraction of key frames, creating a gesture library of 26 letters of the alphabet, and training a recognition percentage of 98% with machine learning algorithms.

By the study of sign language composition and the study of Leap Motion performance, it is proposed that the first opponent language action is decomposed from the hand position, direction, and displacement, and according to the specific coordinate table of the position of the bones corresponding to each sign language action, Leap Motion could extract the above aspects of gesture information, detect and decompose the gesture movement by coordinates, and get the corresponding gesture in the table combined with a gesture library already imputed into machine learning algorithms to classify and identify the meaning of the gesture. This method utilizes Leap Motion's advantages in motion tracking detection and spatial position determination of joints and bones while using a limited identification sample to complete the translation of a large number of sign language vocabulary.

The perspective this project takes on is innovative because it explores the coordinate vector calculation and classification using Support Vector Classification (SVC), Extreme Gradient Boosting (XGBoost), and Artificial Neural Networks (ANNs) with a gesture built of coordinates of individual bones on the finger instead of image classification.

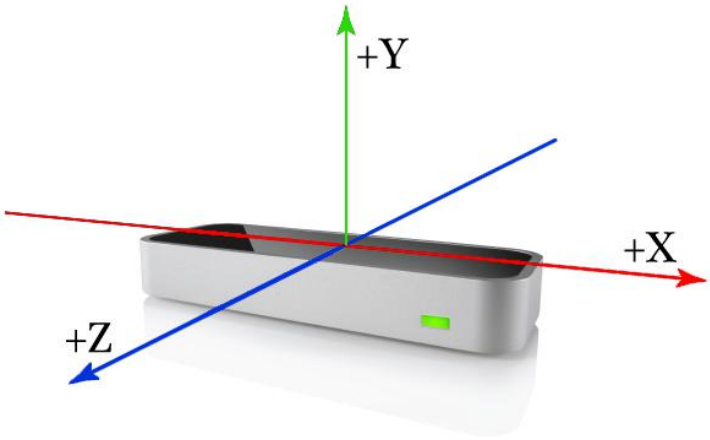
2 Algorithms

2.1 Pattern Recognition

Pattern Recognition is the analysis of the shape and order of an imagery subject and finding the relationship of the images and categorizing and recognizing each one. Pattern Recognition is split into two types: supervised classification and unsupervised classification. Supervised classification trains using supervised learning algorithms to classify different objects into groups previously inputted into the system. It is used in ways such as but not limited to face recognition and object classification[7]. Unsupervised classification refers to image segmentation and cluttering, where the classes of data are not specifically set, and images are classified based on their similarities.

In this project, the pattern recognition of gestures in sign language is classified as supervised classification, because it involves the classification of sign language specifically inline with a gesture library that contains data of the meaning of the sign.

2.2 Leap Motion



[8]

Figure 2.1 Leap Motion Controller

Figure 2.1 shows the physical appearance of the leap motion controller and its x, y, z axis[9]. The Leap Motion controller for hand tracking is a small machine, with dimensions of 13 mm by 80 mm by 30 mm, with an identification range of about 60 cm[10]. In 2013, Leap launched its sense controller, Leap Motion, for about 79.99 dollars. Leap Motion supports a variety of programming languages that allow developers to develop according to the project needs and programming habits. In this project, python is utilized and combined with Leap Motion to communicate. According to Leap Motion' s official website, the controller has a tracking accuracy of up to 0.01 mm and sends back hand data for each frame tracked with a speed of 200 frames per second. Leap Motion uses a core algorithm to store coordinate position data of hand joints after the hand is sensed.

Leap Motion appears to use the principle of light field scanning, emitting infrared light and taking pictures according to the reflection of light on different objects to the controller, based on the light field of the corresponding characteristics of specific objects in order to identify the object reflecting the light.

Leap Motion controller uses two cameras as vision similar to that of the human eye, capturing deep images in stereoscopic space. The two image sensors inside are equipped with infrared LED light is configured on each side of each camera[11]. The infrared LED lights emit infrared light at more than 100 times per second to scan a particular area above the cameras, which when hand motion is applied in that area, the infrared light would be reflected back to the sensors. Then, the sensors send the data back to the computer to track hand data. With Leap Motion's built-in CPU, the controller is able to model the joints and bones of the hand and visualize it on a computer.

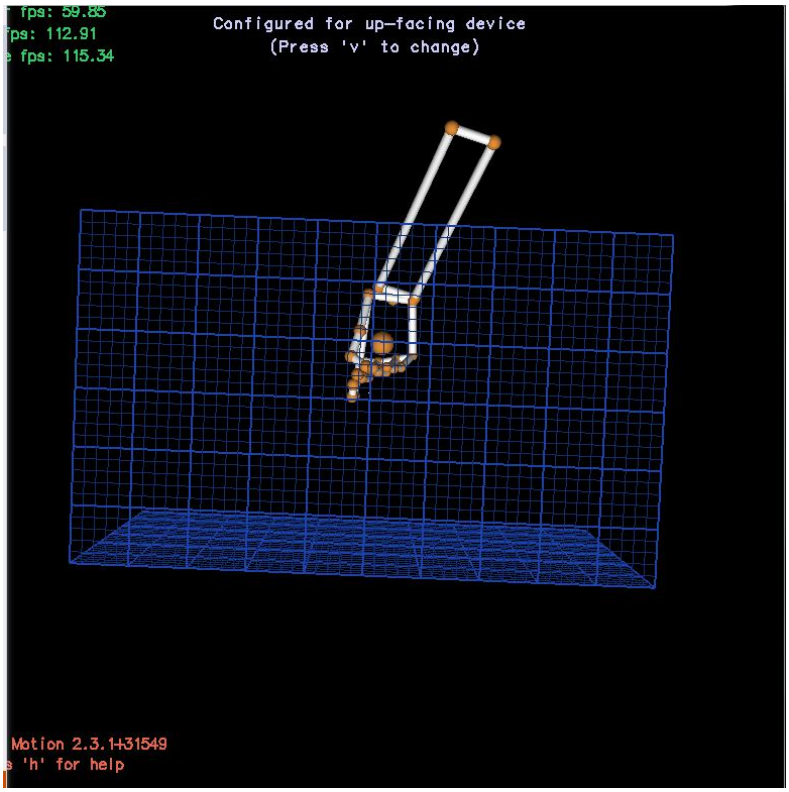


Figure 2.2 Leap Motion Visualizer With a Left Hand

Figure 2.2 shows the user interface which allows the hand motion and its joints and bones to be modeled and visualized. The Leap Motion controller assigns a unique ID number to the recognized hands to identify them. Based on the corresponding ID number, leap motion gives the coordinates of the hand live back to the computer. Leap Motion is able to provide information on the center of the palm, the speed of the palm, the orientation (direction) of the palm by vectors. The same information could be provided for the hands and arm[12]. The data provided by Leap Motion is given in millimeters, and the directions are given as vectors.

The Leap Motion controller monitors the movement and tracks the data of 29 bones and 29 joints on each hand. This project focuses on the position of fingers. On each finger, there is the metacarpal, the proximal, the intermediate, the distal, and the ring bones. Leap Motion tracks the coordinates of each of bone's start and end, with the start being the side closer to the palm, and the end being the side closer to the tip[13]. Each position of the bone is given with (x,y,z) coordinates. The sample leap python code, provided by Leap Motion SDK, called sample.py, contains a sample coding that allows leap motion to print the frame id, time stamp, the number of hands, fingers, and length of each finger. This python application provides the data on the position and direction of the bones on each finger and the arm and wrist. The length information is given in milliliters and the direction is given in vectors.

2.3 Machine Learning Algorithms

Machine Learning Algorithms are employed for supervised classification when training data already inputted into the system. The data collected by the leap motion controller are processed and matched with the inputted gesture. In this project, three

different types of machine learning algorithms are utilized in order to attempt to produce the highest accuracy and precision. The three algorithms are Support Vector Classification (SVC), Extreme Gradient Boosting (XGBoost), and Artificial Neural Networks (ANNs).

2.3.1 Support Vector Classification (SVC)

The support vector machine is an algorithm that aims to find a hyperplane in an n-dimensional space that could best classify the data points. The hyperplane with the largest margin would be the optimal hyperplane, as it increases the probability that future data points would still fall in the same category[14]. Support vectors are extreme points and vectors that help create the hyperplane[15]. In support vector classification, when the sample is linearly divisible in two-dimensional space, the sample in the space would be divided by a straight line, differentiating one side from the other. However, not all data sets could be divided linearly, therefore, SVC employs the kernel method, a method to solve a non-linear problem with linear classifiers[16]. In high-dimensional spaces, the sample is divided by a hyperplane. Using the kernel function, a two-dimensional data set could be projected onto a high-dimensional space and allow for classification, independent of the form of transformation. A "C" in SVC controls the size of the margin of the hyperplane to lower the incorrect classification rate[17]. A linear kernel is defined as:

$$f(x) = w^T * x + b \text{ [18]}$$

Where x is the data classified, w is the minimizing weight vector, and b is the linear coefficient.

2.3.2 XGBoost

Extreme Gradient Boosting (XGBoost) is an algorithm in machine learning that involves in gradient boosted decision trees[19]. XGBoost is a form of ensemble learning. The most commonly used methods of ensemble learning are bagging and boosting. Bagging is where XGBoost uses boosting[20]. Boosting is where decision trees are built in sequence that each tree aims to reduce preceding errors. Therefore, each newly grown tree would grow from the new, updated version of the data remaining. However, boosting does not account for cost minimization[21], which is the reduction of errors in the training set, and this is where gradient boosting comes in. Gradient descent accounts for minimizing the cost function, as it attempts to find the minimum point of a function[22]. XGBoost combines gradient decent and boosting decision trees in order to achieve a highly efficient and fast machine learning algorithm.

2.3.3 Artificial Neural Networks

Artificial Neural Networks (ANNs) are an attempt to imitate brain neuron activity. In a neural network, there would always be an input layer and an output layer. The input layer is the layer that is responsible for managing the inputs from an external source. In neural networks, there is also an output layer, which is responsible for

computing the results. Between the input and output layers, there are more layers called the hidden layers, these layers are required when data needs to be separated non-linearly[23]. These layers are not visible to the external systems and they receive output from previous input layers or hidden layers and would help with calculations[24].

3 Classification

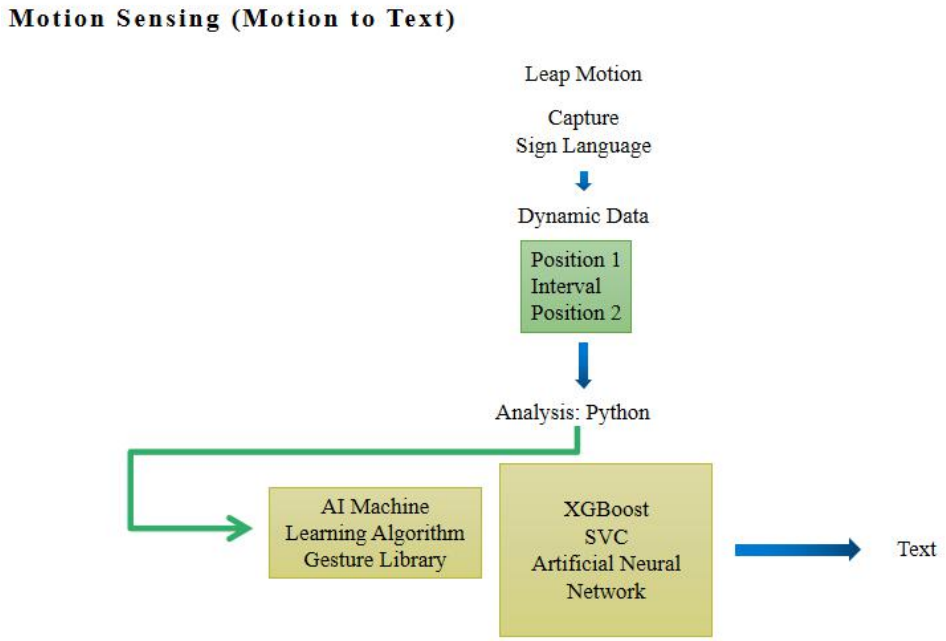


Fig. 3.1 Work Flow Chart

Figure 3.1 shows the application plan for the project. First, using the system of Leap Motion, dynamic data is collected and classified using two positions, the initial position, and the end position, with an interval between them. Using the 26 characters in the ASL alphabet as machine learning subjects, the project achieves the classification and recognition of gesture data. And finally, with the collection of an archive of ASL vocabulary, the project manages to translate the fingerspelling alphabet.

3.1 Bone Position Capturing

First, the Leap Motion controller is set up facing up on the table with minimum disrupting factors to ensure the accuracy of tracking. In the project, Visual Studio with python 2.7 was used because Leap Motion coding only supports python 2.7. The strategy employed is to calculate the distance between the initial position and the final position of each of the bones, by modifying the sample.pv document provided by leap motion SDK, only the start and end position of the bones are printed on the screen and written into a .txt file saved to the folder for individual letters.

Each hand has five fingers, the thumb, the index, the middle, the ring, and the

pinky, and each finger consists of four bones, the metacarpal, the proximal, the intermediate, and the distal. With bone.prev_joint, bone.next_joint, and bone.direction repeated for four times, the program could print the coordinates of each bone in multiple frames of all five fingers.

The actual data collection consists of letting leap motion recognize an initial palm position at the start of every letter and slowly shifting the fingers to become the letters without moving the center palm, running the program 20 times per letter. After writing all the gestures into .txt files, each of the 26 test letters has a total of 20 input samples, sorted into their individual folders.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
18	19	20	21	22	23	24	25	26								
R	S	T	U	V	W	X	Y	Z								

Fig. 3.3 The Letters and Their Assigned Number

The key frame extraction approach taken on was to take the first and last frames of the gesture captured. This set of 26 letters, each assigned to an individual number, are qualified to participate in the sample group.

The coordinates of bones of a hand are displayed with the following order. The first set of numbers is the frame id of the frame captured gotten with print "Frame id: %d" % (frame.id), each frame has a unique frame id. The second number is the finger id of the thumb. Below each bone name, the start position of the bone, the end position of the bone, and the orientation of the bone are displayed. This information makes up the gesture library that corresponds to each translation.

The data stored in the folders are not processed and are just individual data with many unwanted intermediate frames. Only the first and last frames are needed for displacement calculations, therefore, with firstframe = lines[0:87] and lastframe = lines[-86:] written into a file would allow the data of the first and last frames, each with 87 lines, to be generated into separate files. and converted into .json files with the essential information separately stated only. After conversion, the data is separated into a train_set.csv and a test_set.csv, where 14 sets of data of each letter are used to train and power the machine learning algorithms, and 6 sets of each letter are used to test for accuracy. The training set is provided to the three types of algorithms, (SVC, XGBoost, Neural Network) and tested.

3.2 Gesture Classification

In SVC, C was set to 1 and gamma to 0.005. The linear kernel is chosen to be the best fit.

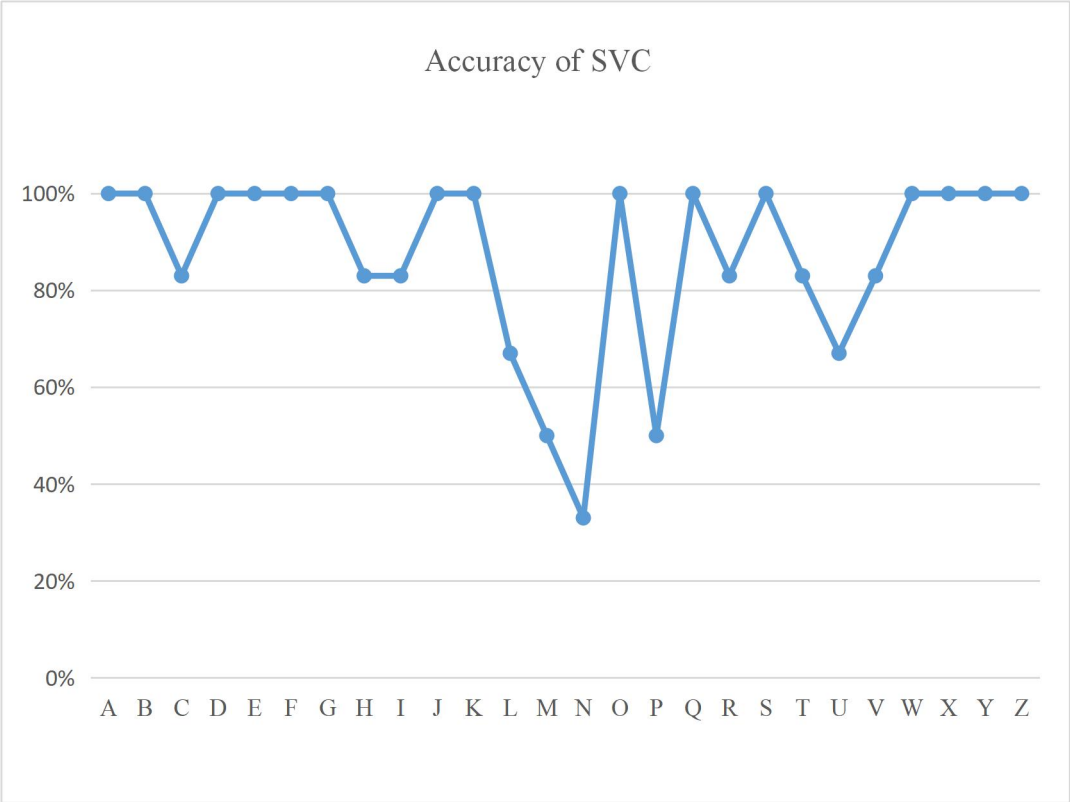


Fig. 3.4 Individual Accuracy of Different Letters with SVC

With the Support Vector Classification of data, an accuracy of about 88% was achieved as shown in the graph. Many of the letters have an individual accuracy of 100% with some minor inconsistencies. However, letters like “m”, “n’, and “p” are less accurately determined.

In XGBoost, 100 boosting trees are employed with a learning rate of 0.05 and a depth of 3.

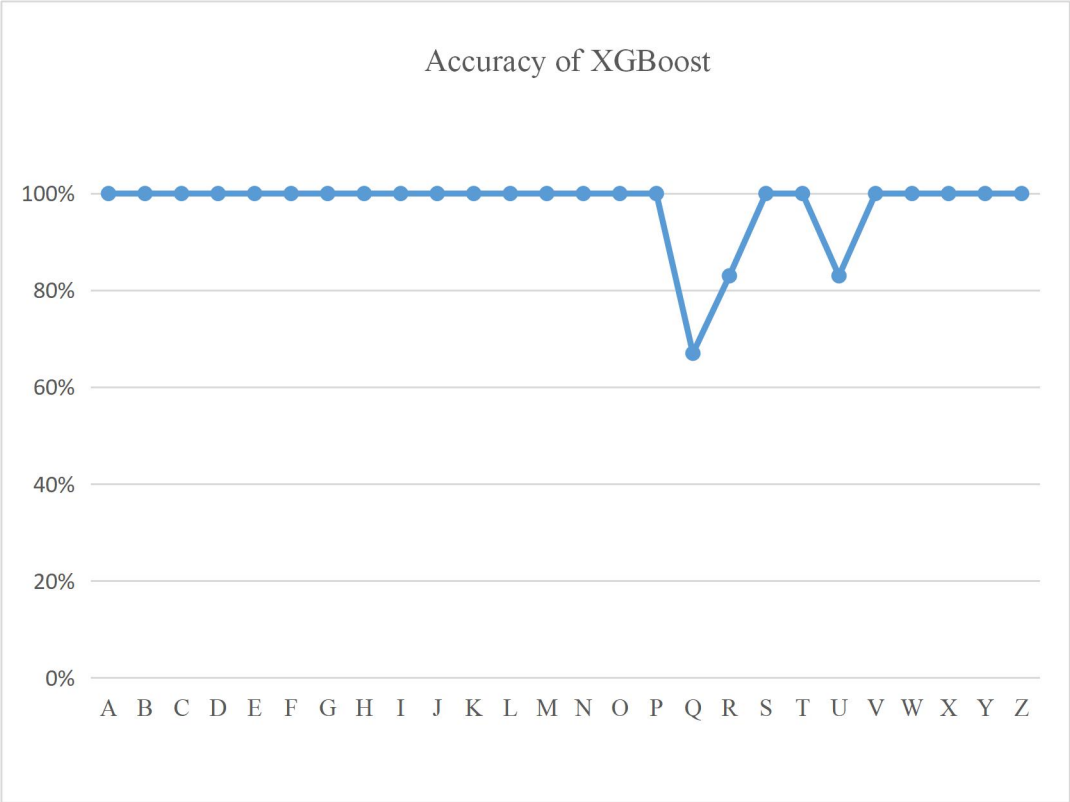


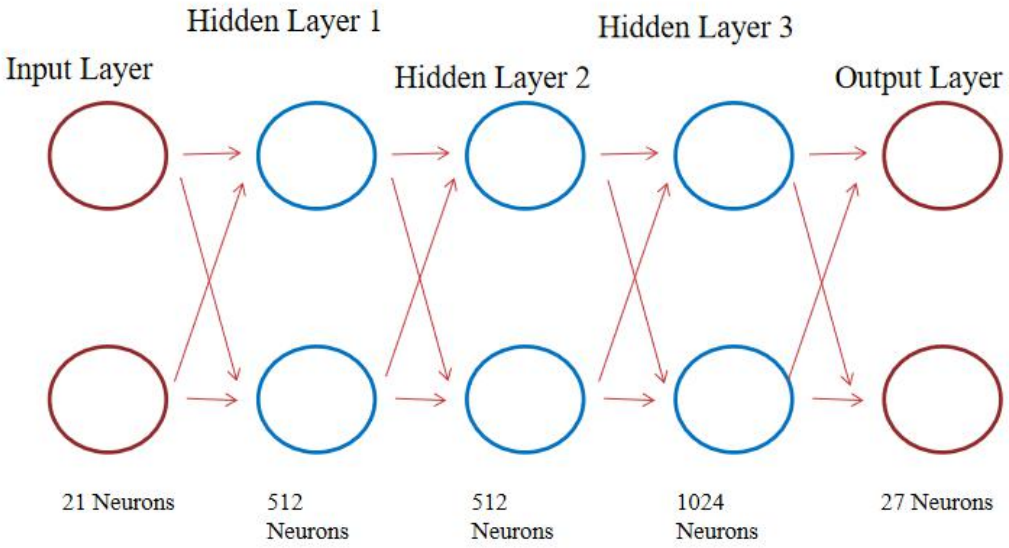
Fig. 3.5 Individual Accuracy of Different Letters with XGBoost

Using XGBoost, an average accuracy of 98% was achieved. Most of the letters have an individual accuracy of 100% with only some letters in the 90% and 80% area.

Three hidden layers are employed in the classification of gestures in this project with tensorflow to provide the best results. The Rectified Linear Unit (ReLU) activation function is set for each hidden layer, which is a linear function, where

$$R(z) = \max(0, z)$$ [25]

The training set has 335 data points and 21 input features, and the neural network model is modeled to look like this:



Where 21 neurons are employed in the input layer, 512 and 1024 neurons are employed in hidden layers, and 27 output neurons that decide the outcome.

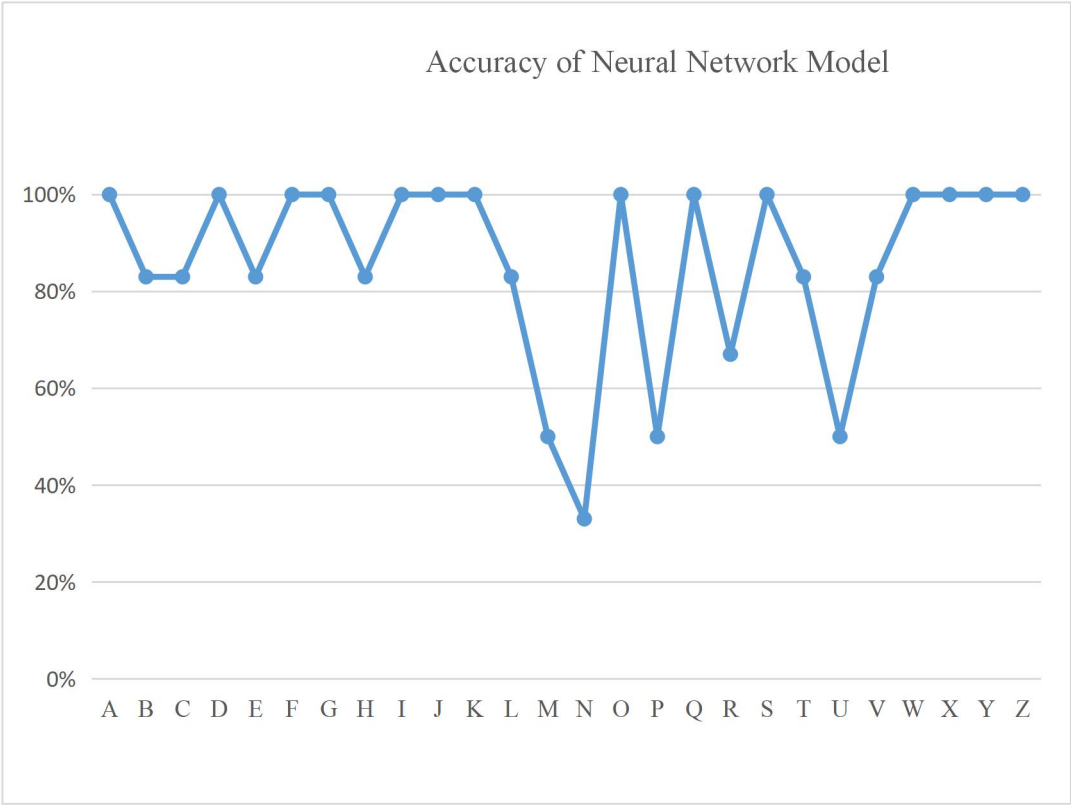


Fig. 3.6 Individual Accuracy of Different Letters with Neural Network

Classification with the Neural Network Model achieves an accuracy of about 86%, where letters like “m”, “n”, and “u” with lower accuracy. However, the accuracy percentage of many letters are less than the ones tested with other algorithms.

4 Conclusion

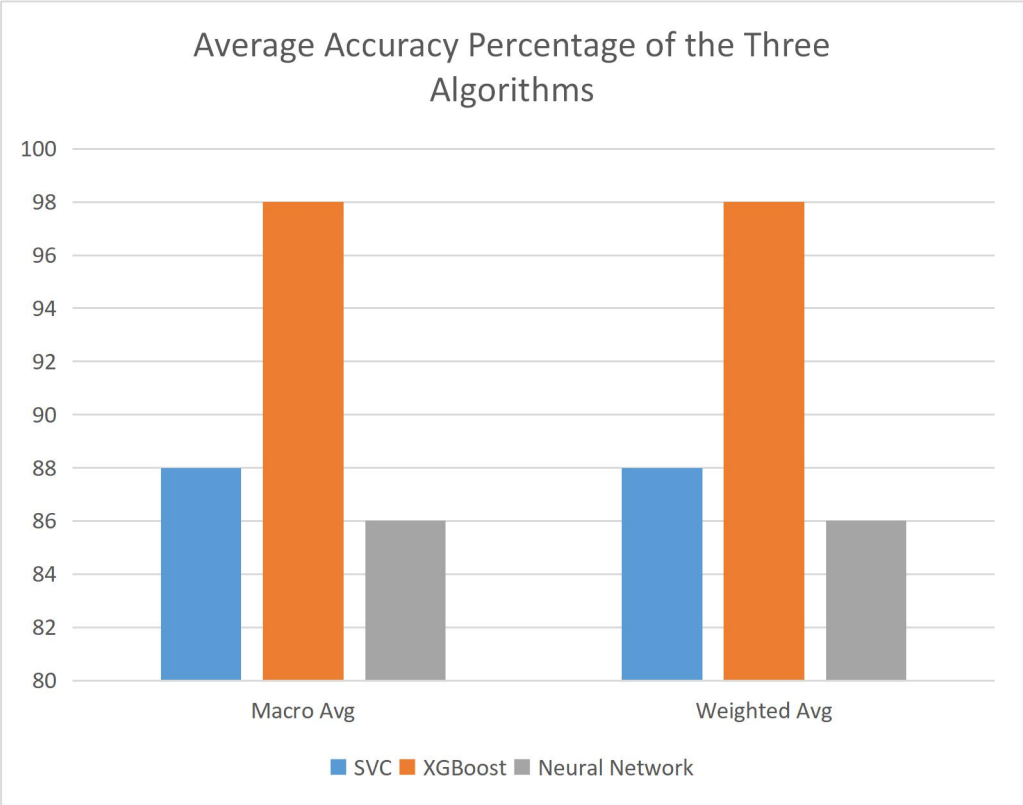


Fig. 4.1 Average Accuracy Percentage of the Three Algorithms

Tested with the three types of algorithms, the accuracy of XGBoost provided the highest percentage of 98% as compared with the 88% of SVC, and the 86% of Neural Network Model. This is probably due to the amount of data being provided. The Neural Network Model performs better with large amounts of data, which is possibly a cause for the result to be a lower percentage.

4.1 Errors and Limitations

Leap Motion has a small identification range, and the main data collected is the displacement of the joints of the fingers, therefore, the individual's hand must be in the range of identification, which means many gestures of ASL cannot be recognized.

Deaf people who have been signing their whole life tend to sign faster than leap motion recognizes, and the project depends on the running speed and accuracy of the leap motion controller.

The leap motion controller heats up very quickly, with a span of about 30 minutes, and therefore, is incompatible with long term work spans, and must be unplugged from the device to allow for cool-down.

Test subject hand position from the beginning to the end might slightly vary, due to finger motions being small, only moving in the millimeters, would cause precision variations of about 3-4 mm. Also, the extent of the stretch of each finger at the initial position might also vary from gesture to gesture.

Leap Motion does not perform very well with the recognition of the movement of

bent fingers or fingers blocking between other fingers, which is essential for the recognition and differentiation of the letters, especially “t”, “m”, “n”, and “s”, which the only difference is the position of the thumb in between fingers.

4.2 Future Considerations

This project researches the identification and classification of static hand motions that are determined by their fingers and displacement between an open palm position and their separate gestures, therefore, possible future development includes the recognition of movement in between gestures from gesture A to B. Also, translation needs to support the different sizes of different hands and their differentiation in millimeters of movement.

This study confirms that gesture recognition can be done by calculation of bone section coordinate transformation, in addition to image recognition. Because of the tight time frame in this study, the amount of data collected is limited and is transformed from a single gesture to a specific letter gesture. In the future, if a large amount of gesture data for various transformations are accumulated, and because coordinates are the advantage of quantitative calculation, it is possible to quickly identify various continuous gestures.

In the future, rather than increasing the amount of data, coordinate vector calculation and image recognition can complement each other so that the accuracy could be further increased, which is really a field worth exploring.

Bibliography

Afonja, T. (2018, July 13). Kernel Functions. Retrieved July 05, 2020, from https://towardsdatascience.com/kernel-function-6f1d2be6091

Coordinate System. (n.d.). Retrieved July 22, 2020, from https://developer.leapmotion.com/documentation/v4/concepts.html

Dataflair Team. (2018, November 16). What is XGBoost Algorithm - Applied Machine Learning. Retrieved July 04, 2020, from https://data-flair.training/blogs/xgboost-algorithm/

Deshpande, M. (2019, September 25). Classification with Support Vector Machines. Retrieved July 05, 2020, from https://pythonmachinelearning.pro/classification-with-support-vector-machines/

Fels, S. (1970, January 01). Using Normalized RBF Networks to Map Hand Gestures to Speech. Retrieved July 04, 2020, from https://link.springer.com/chapter/10.1007/978-3-7908-1826-0_3

Gad, A. (2018, June 27). Beginners Ask "How Many Hidden Layers/Neurons to Use in Artificial Neural Networks?" Retrieved July 05, 2020, from https://towardsdatascience.com/beginners-ask-how-many-hidden-layers-neurons-to-use-in-artificial-neural-networks-51466afa0d3e

Gandhi, R. (2018, July 05). Support Vector Machine - Introduction to Machine Learning Algorithms. Retrieved July 05, 2020, from https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

Guest Blog. (2020, May 24). XGBoost Algorithm: XGBoost In Machine Learning. Retrieved July 04, 2020, from https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/

Hidden Layers in Neural Networks. (2019, October 19). Retrieved July 05, 2020, from https://www.i2tutorials.com/hidden-layers-in-neural-networks/

The Editors of Encyclopaedia Britannica. (2020, March 31). William C. Stokoe, Jr. Retrieved July 04, 2020, from https://www.britannica.com/biography/William-C-Stokoe-Jr

Ma, J., & Gao, W. (2000, October 16). A PARALLEL MULTI-STREAM MODEL FOR SIGN LANGUAGE RECOGNITION. Retrieved July 04, 2020, from https://www.isca-speech.org/archive/archive_papers/icslp_2000/i00_2751.pdf

Mandot, P. (2019, February 10). How exactly XGBoost Works? Retrieved July 04, 2020, from https://medium.com/@pushkarmandot/how-exactly-xgboost-works-a320d9b8aeef

McGregor, M. (2020, July 02). SVM Machine Learning Tutorial – What is the Support Vector Machine Algorithm, Explained with Code Examples. Retrieved July 25, 2020, from https://www.freecodecamp.org/news/svm-machine-learning-tutorial-what-is-the-support-vector-machine-algorithm-explained-with-code-examples/

Murakami, K. Taguchi, H. (1991, March 01). Gesture recognition using recurrent neural networks. Retrieved from https://dl.acm.org/doi/10.1145/108844.108900

Nguyen, X. (2020, June 29). Minimizing the cost function: Gradient descent. Retrieved July 04, 2020, from https://towardsdatascience.com/minimizing-the-cost-function-gradient-descent-a5dd6b5350e1

Pattern Recognition. (n.d.). Retrieved July 04, 2020, from https://www.mathworks.com/discovery/pattern-recognition.html

Sharma, S. (2019, February 14). Activation Functions in Neural Networks. Retrieved July 23, 2020, from https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6

Support Vector Machine (SVM) Algorithm - Javatpoint. (n.d.). Retrieved July 05, 2020, from https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm

Ultraleap. Tracking Model. (n.d.). Retrieved July 04, 2020, from https://developer-archive.leapmotion.com/documentation/python/devguide/Leap_Tracking.html

Ultraleap. (2019, December 15). Leap Motion Controller Data Sheet. Retrieved July 04, 2020, from https://www.ultraleap.com/datasheets/Leap_Motion_Controller_Datasheet.pdf

Ultraleap. Bone. (n.d.). Retrieved July 04, 2020, from https://developer-archive.leapmotion.com/documentation/python/api/Leap.Bone.html

Ultraleap. (n.d.). About. Retrieved July 04, 2020, from https://www.ultraleap.com/company/about/

Ultraleap. (n.d.). World-leading Hand Tracking: Small. Fast. Accurate. Retrieved July 04, 2020, from https://www.ultraleap.com/tracking/#how-it-works

Weichert, F., Bachmann, D., Rudak, B., & Fisseler, D. (2013, May 14). Analysis of the accuracy and robustness of the leap motion controller. Retrieved July 04, 2020, from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3690061/

World Health Organization. Estimates. *Prevention of Blindness and Deafness.* (2018, July 26). Retrieved July 04, 2020, from https://www.who.int/pbd/deafness/estimates/en/

[1] World Health Organization. Estimates. *Prevention of Blindness and Deafness.* (2018, July 26). Retrieved July 04, 2020, from https://www.who.int/pbd/deafness/estimates/en/

[2] Ultraleap. About. (n.d.). Retrieved July 04, 2020, from https://www.ultraleap.com/company/about/

[3] Fels, S. (1970, January 01). Using Normalized RBF Networks to Map Hand Gestures to Speech. Retrieved July 04, 2020, from https://link.springer.com/chapter/10.1007/978-3-7908-1826-0_3

[4] The Editors of Encyclopaedia Britannica. (2020, March 31). William C. Stokoe, Jr. Retrieved July 04, 2020, from https://www.britannica.com/biography/William-C-Stokoe-Jr

[5] Murakami, K. Taguchi, H. (1991, March 01). Gesture recognition using recurrent neural networks. Retrieved from https://dl.acm.org/doi/10.1145/108844.108900

[6] Ma, J., & Gao, W. (2000, October 16). A PARALLEL MULTI-STREAM MODEL FORSIGN LANGUAGE RECOGNITION. Retrieved July 04, 2020, from https://www.isca-speech.org/archive/archive_papers/icslp_2000/i00_2751.pdf

[7] Pattern Recognition. (n.d.). Retrieved July 04, 2020, from https://www.mathworks.com/discovery/pattern-recognition.html

[8] Coordinate System. (n.d.). Retrieved July 22, 2020, from https://developer.leapmotion.com/documentation/v4/concepts.html

[9] Weichert, F., Bachmann, D., Rudak, B., & Fisseler, D. (2013, May 14). Analysis of the accuracy and robustness of the leap motion controller. Retrieved July 04, 2020, from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3690061/

[10] Ultraleap. (2019, December 15). Leap Motion Controller Data Sheet. Retrieved July 04, 2020, from https://www.ultraleap.com/datasheets/Leap_Motion_Controller_Datasheet.pdf

[11] Ultraleap. (n.d.). World-leading Hand Tracking: Small. Fast. Accurate. Retrieved July 04, 2020, from https://www.ultraleap.com/tracking/#how-it-works

[12] Ultraleap. Tracking Model. (n.d.). Retrieved July 04, 2020, from https://developer-archive.leapmotion.com/documentation/python/devguide/Leap_Tracking.html

[13] Ultraleap. Bone. (n.d.). Retrieved July 04, 2020, from https://developer-archive.leapmotion.com/documentation/python/api/Leap.Bone.html

[14] Gandhi, R. (2018, July 05). Support Vector Machine - Introduction to Machine Learning Algorithms. Retrieved July 05, 2020, from https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

[15] Support Vector Machine (SVM) Algorithm - Javatpoint. (n.d.). Retrieved July 05, 2020, from https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm

[16] Afonja, T. (2018, July 13). Kernel Functions. Retrieved July 05, 2020, from https://towardsdatascience.com/kernel-function-6f1d2be6091

[17] Deshpande, M. (2019, September 25). Classification with Support Vector Machines. Retrieved July 05, 2020, from https://pythonmachinelearning.pro/classification-with-support-vector-machines/

[18] McGregor, M. (2020, July 02). SVM Machine Learning Tutorial – What is the Support Vector Machine Algorithm, Explained with Code Examples. Retrieved July 25, 2020, from https://www.freecodecamp.org/news/svm-machine-learning-tutorial-what-is-the-support-vector-machine-algorithm-explained-with-code-examples/

[19] Dataflair Team. (2018, November 16). What is XGBoost Algorithm - Applied Machine Learning. Retrieved July 04, 2020, from https://data-flair.training/blogs/xgboost-algorithm/

[20] Guest Blog. (2020, May 24). XGBoost Algorithm: XGBoost In Machine Learning. Retrieved July 04, 2020, from https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/

[21] Mandot, P. (2019, February 10). How exactly XGBoost Works? Retrieved July 04, 2020, from https://medium.com/@pushkarmandot/how-exactly-xgboost-works-a320d9b8aeef

[22] Nguyen, X. (2020, June 29). Minimizing the cost function: Gradient descent. Retrieved July 04, 2020, from

https://towardsdatascience.com/minimizing-the-cost-function-gradient-descent-a5dd6b5350e1

[23] Gad, A. (2018, June 27). Beginners Ask "How Many Hidden Layers/Neurons to Use in Artificial Neural Networks?" Retrieved July 05, 2020, from https://towardsdatascience.com/beginners-ask-how-many-hidden-layers-neurons-to-use-in-artificial-neural-networks-51466afa0d3e

[24] Hidden Layers in Neural Networks. (2019, October 19). Retrieved July 05, 2020, from https://www.i2tutorials.com/hidden-layers-in-neural-networks/

[25] Sharma, S. (2019, February 14). Activation Functions in Neural Networks. Retrieved July 23, 2020, from https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6