ALPHABET SIGN LANGUAGE RECOGNITION USING LEAP MOTION TECHNOLOGY AND RULE BASED BACKPROPAGATION-GENETIC ALGORITHM NEURAL NETWORK (RBBPGANN)

Wijayanti Nurul Khotimah[1)], Risal Andika Saputra[2)], Nanik Suciati[3)], Ridho Rahman Hariadi[4)]
[1, 2,3)] Department of Informatics, Institut Teknologi Sepuluh Nopember (ITS)
Kampus ITS Surabaya, 60111, Indonesia

e-mail: wijayanti@if.its.ac.id[1)], risal.andika@gmail.com[2)], nanik@if.its.ac.id[3)], ridho@if.its.ac.id[4)]

ABSTRAK

Pengenalan bahasa isyarat digunakan untuk membantu manusial normal berkomunikasi dengan manusia yang tuli atau mengalami gangguan dalam pendengaran. Berdasarkan hasil survei yang dilakukan oleh Multi Center Study di Asia Tenggara, Indonesia menempati urutan ke empat dengan penderita gangguan pendengaran yaitu sekitar 4,6% dari total penduduk. Oleh karena itu, keberadaan aplikasi untuk pengenalan bahasa isyarat sangat dibutuhkan. Beberapa penelitian telah dilakukan dalam bidang ini. Beberapa macam tipe neural network telah digunakan untuk mengenali beberapa macam bahasa isyarat. Penelitian ini berfokus pada pengenalan bahasa isyarat alphabet pada kamus SIBI yang menggunakan satu tangan dan 26 gesture. Tiga puluh fitur diekstraksi menggunakan teknologi Leap Motion. Kemudian algoritma Back Propagation Genetic Algorithm Neural Network (BPGANN) digunakan untuk mengenali bahasa isyarat tersebut. Dari hasil uji coba, aplikasi yang diusulkan ini mampu mengenali bahasa isyarat dengan tingkat akurasi mencapai 90%.

***Kata Kunci**: bahasa isyarat, leap motion, backpropagation genetic algorithm neural network.*

ABSTRACT

Sign Language recognition was used to help people with normal hearing communicate effectively with the deaf and hearing-impaired. Based on survey that conducted by Multi-Center Study in Southeast Asia, Indonesia was on the top four position in number of patients with hearing disability (4.6%). Therefore, the existence of Sign Language recognition is important. Some research has been conducted on this field. Many neural network types had been used for recognizing many kinds of sign languages. However, their performance are need to be improved. This work focuses on the ASL (Alphabet Sign Language) in SIBI (Sign System of Indonesian Language) which uses one hand and 26 gestures. Here, thirty four features were extracted by using Leap Motion. Further, a new method, Rule Based-Backpropagation Genetic Algorithm Neural Network (RB-BPGANN), was used to recognize these Sign Languages. This method is combination of Rule and Back Propagation Neural Network (BPGANN). Based on experiment this proposed application can recognize Sign Language up to 93.8% accuracy. It was very good to recognize large multiclass instance and can be solution of overfitting problem in Neural Network algorithm.

***Keywords**: sign language, leap motion, backpropagation genetic algorithm neural network.*

I. INTRODUCTION

SIGN Language recognition was used to help people with normal hearing communicate effectively with the deaf and hearing-impaired. There are two main approaches of Sign Language recognition: image-based approach and sensor-based approach. Each approach has its own adventage and disadventage. In image-based approach, people do not need to wear cumbersome devices. However, mostly this approach requires expensive computation and clear environment. The computation in sensor-based approach is not as expensive as in image-based approach. Nevertheless, in this approach user need to wear cumbersome devices such as gloves [1].

Some research has been conducted by using one of those approaches. In 2005, Khaled and Al-Rouslane did recognition for arabic Sign Language alphabets using polynomial classifiers. In their research, they used image-based approach for the recogniition. Thirty features were extracted in this research. These features are related to the fingertips and their relative positions and orientations with respect to wrist and to each other fingertips. For recognition, they used polynomial classifier, ANFIS systems. From experiment, accuracy of this system reach

93.5% [2]. Another research that used image-based approach is reserach that was done by Feras et. al. [3]. They developed an automatic isolated-word arabic Sign Language recognition system using time delay neural network (TDNN). In their research, two different color gloves were used for performing sign. Then the sign was processed using image processing method. The features, the centroid position for each hand and the change on horizontal and vertical velocity, were extracted after image processing. After being extracted, these features were classified using TDNN. Forty arabic words were tested during the experiment and the recognition rate reach 70%. In 2012, Reyadh et. al. developed a system for recognizing arabic Sign Language using K-Nearest Neighbor algorithm. In their system, Sign Language was represented in to histogram. The distance of histogram in test sign languge was compared to the histogram in training Sign Language. The test Sign Language then was recognized by using K-NN [4].

Different to image-based approach system, system that used sensor-based approach requires additional devices such as Kinect or Leap Motion. In 2014, Edon et al. developed Kosova Sign Language recognition system using Kinect. This system was used to recognize nine numbers, fifteen alphabet letters, four words, and one sentence. To detect these signs, this system extracted some features that was obtained from kinect sensors. These features are skeleton positions, the shape of the hand, hand position relative to other body parts, and hand movement direction. From experiments that involved two native signers and one non-native signer, this system was able to recognize those signers with average accuracy 75% [5]. In the same year, Yang et. al. used 3D depth information generated from Microsoft's Kinect sensor and applied a hierarchical CRF (Conditional Random Field) to recognize hand signs. Six features from 3D space and one feature from 2D space were extracted using the detected hand and face region. These features were used to discriminate between sign and non-sign patters by H-CRF. Then BoostMap algorithm recognized the sign patterns. From experiment, this proposed method was able to recognize sign pattern at a rate of 90.4 % [6].

For methods, many neural network types had been used for recognizing many kinds of singn languages. These are: back propagation network that had been used in Japanese language recognition, Elman recurrent network that had been used in Japanese language recognition and Arabic language recognition, fully recurrent network that had been used in Arabic language recognition, and supervised neural network that had been used in Myanmar language recognition. The average accuracy of these neural networks for Sign Language recognition was 85% [7]. Moreover Manar et. al. had compared four types of neural network (feedforward neural network, Elman neural network, Jordan neural network, and recurrent neural network) for recognizing Arabic Sign Language. Accuracy rate of each type of the algorithm was computed. From experiment, recurrent neural netwok gives the highest accuracy and feedforward neural network gives the loweest accuracy [8].

In Indonesia, based on survey that conducted by Multi Center Study in Southeast Asia, Indonesia was on the top four position in number of patients with hearing disability (4.6%). The top three countries with hearing disability patients are Sri Lanka (8.8%), Myanmar (8.8%) and India (6.3%) [9]. Because the number of patients with hearing disability in Indonesia is a lot, in 1994, the ministry of education and culture released SIBI (Sistem Isyarat Bahaasa Indonesia) in the form of dictionary. SIBI is official indonesian Sign Language. This dictionary consists of finger and hand movements that represented indonesian vacabulary. Gestures in the SIBI has been arranged systematically. However, learning tools that refers to SIBI did not interactive. They only consist of Sign Language image and its meaning as shown in Figure 1.

This work focuses on the ASL (Alphabet Sign Language) in SIBI which uses one hand and 26 gestures (see Figure 1). Thirty four features were extracted by using Leap Motion. Further, Rule Based-Backpropagation Genetic Algorithm Neural Network (RB-BPGANN) was used to recognize these Sign Languages. The remainder of this paper is organized as follows; section two describes the method of this research, in section III and section IV, result and analysis are presented respectively. Section V presents conclusion.

II. RESEARCH METHOD

This section describes the methods component used in this system. Those components are system architecture, feature extraction, normalization, calibration, BPGANN, and RB-BPGANN

A. System Architecture

The architecture of this proposed system was divided in to two: training system architecture and testing system architecture. The training system architecture was shown in Figure 2 while the testing system architecture was shown in Figure 3. Both in training process and testing process, the Leap Motion will receive hand gesture as input. The information from hand gesture will be extracted in to 34 features. In training process, 260 hand gestures will be extracted and the result will be saved in DataSet files. The DataSet will be grouped in to three groups based on

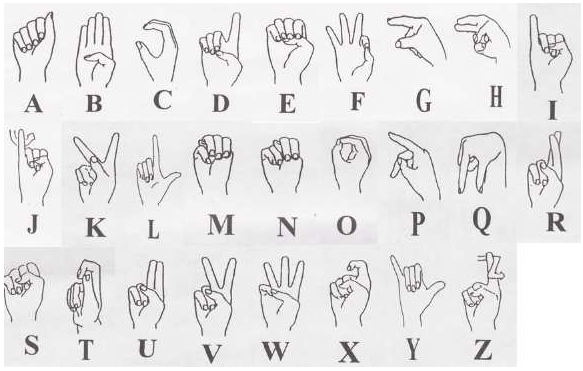

Figure 1. Alphabet Sign Language in SIBI

Table 1. Each group data will be processed using BPGANN method. The output of BPGANN is classifier that will be saved in xml files. Meanwhile, in testing one hand gesture will be extracted and used as input for the classifier based on rule in Table 1. The output of this process is the alphabet that will be written in text.

B. Feature Extraction

Leap Motion is very sensitive tool. In this research, one hand gesture was obtained from 10 consecutive frames. Thirty four features of each frame were extracted in this research. The features from those frames were averaged and were used as features of one sample data. These features are related to the fingertips and their relative positions and orientations with respect to palm and to each other fingertips.

In this research, thumb finger is referred to as the first finger, index finger is called as the second finger, the middle finger is referred to as the third finger, the ring finger is called as the fourth finger, and the little finger is referred to as the fifth finger. The first feature until the fourth feature were taken from research that was conducted by Aliyu [10]. The rest of the features were created from observation. These features are: fist radius; three features are from rotation around the x-axis, y-axis, and z-axis; five features are from the distance between each finger and the palm of the hand in x-axis; five features are from the distance between each finger and the palm of the hand in y-axis, five features are from the distance between each finger and the palm of the hand in z-axis, four features are from distance between the first finger and the others finger in x-axis; four features are from distance between the first finger and the others finger in y-axis; four features are from distance between the first finger and the others finger in z-axis; and three features are from distance between the second finger and the third finger to the x-axis, y-axis, and z-axis.

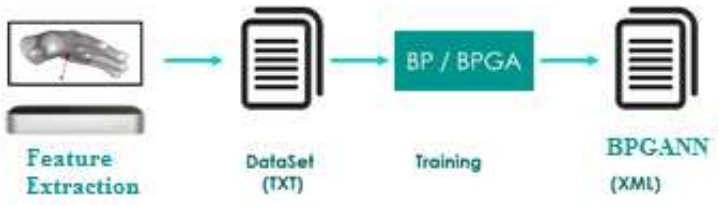

Figure 2. Training Architecture

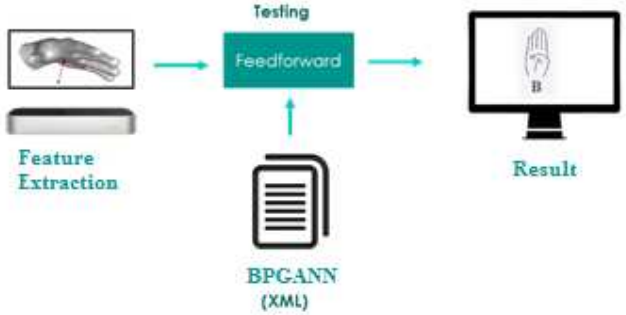

Figure 3. Testing Architecture

Input	Features
Output	Reducted Features

```
for i=0:populasi.Length
    for j=0:kromosom.Length
     kromosom[i] ← RandomBinary()
for i=0:populasi.Length
    for j=0:kromosom.Length
     if kromosom[i][j] == 0
        Fitur.RemoveBit(i)
      end
```

Figure 4 Chromosome Initialization Pseudocode

C. Normalization

Feature normalization was computed by using Equation (1). The purpose is to guarantee all features have proportional range. X is the value of feature before normalization, X_b is feature's value after normalization, X_{min} and X_{max} are the minimum and the maximum value of feature respectively. The maximum and minimum value of those features were obtained from observation. In this research the maximum and minimum value of those features are the minimum and the maximum value of data training in each feature.

$$\boldsymbol{X_b = \frac{X - X_{min}}{X_{max} - X_{min}}} \quad (1)$$

D. Calibration

Calibration was done by comparing the user's hand to the trainer's hand using Equation (2) and (3). This process was implemented because the size of user's hand and trainer's hand was different. M_w is width multiplier, M_l is length multiplier, N_{wuser} is width size of user's hand, $N_{wtrainer}$ is width size of trainer's hand, N_{luser} is length size of user's palm, $N_{ltrainer}$ is length size of trainer's palm hand. Hand's width is a distance between thumb finger and little finger on x-axis. Palm's length is distance between palm position and middle finger on y-axis.

$$\boldsymbol{M_w = \frac{N_{wuser}}{N_{wtrainer}}} \quad (2)$$

$$\boldsymbol{M_l = \frac{N_{luser}}{N_{ltrainer}}} \quad (3)$$

E. BPGANN

Neural network that based on back-propagation algorithm has been widely used. However, some insufficient exist in BP-algorithm such as the solution is plunging into local minimum, the goal convergence is low, etc. A hybrid neural network based on combination of genetic algorithm (GA) and BP-algorithm (BPGANN) was proposed to minimize these insufficiencies.

In this algorithm, GA was used to search the initial weight and biases of the network. And to accelerate the convergence of neural network when BP algorithm makes convergence become slow around the training goal [11]. Different to previous research that used GA for weight initialization, in this research the used of GA in BPNN was for feature extraction. The process of BPGANN are as follows:

1. Using GA to search the best chromosome. This chromosome then was decoded as the features that will be used in neural network. In this research, one chromosome has 34 gens. Each gen represented one feature and had value either 0 or 1. For example the first gen represented the first feature. If the first gen has value 1 means that the first feature will be used in the neural network training.
2. Using BP algorithm to train the network. In this network, the features are result of GA.
3. If the performance of neural network is very different then go to step 5, otherwise go to step 4.
4. Update the weight as the initial population of the GA and use it to search the most superior chromosome, then go to step 2.
5. Determine whether the result satisfy or not. If the result satisfies, stop the iteration else go to step 2. The diagram of BPGANN algorithm was shown in Figure 5.

Input	Chromosome
Output	Chromosome with fitness value

```
for i=0:populasi.Length
    totalCorrect ← 0
  for j=0:totalTesting
     FF.Run(j)
     correct ← true
     targetClass ← csfonClass.GetTarget(listDataSet[j].ClassName)
  for k=0:TotalClass
     if target [k] != FF.Actual()[k]
       correct ← false
     end
     if correct
       totalCorrect++
     end
   chromosome[i].fv ← totalCorrect/totalTest
```

Figure 6. Fitness Value Pseudocode

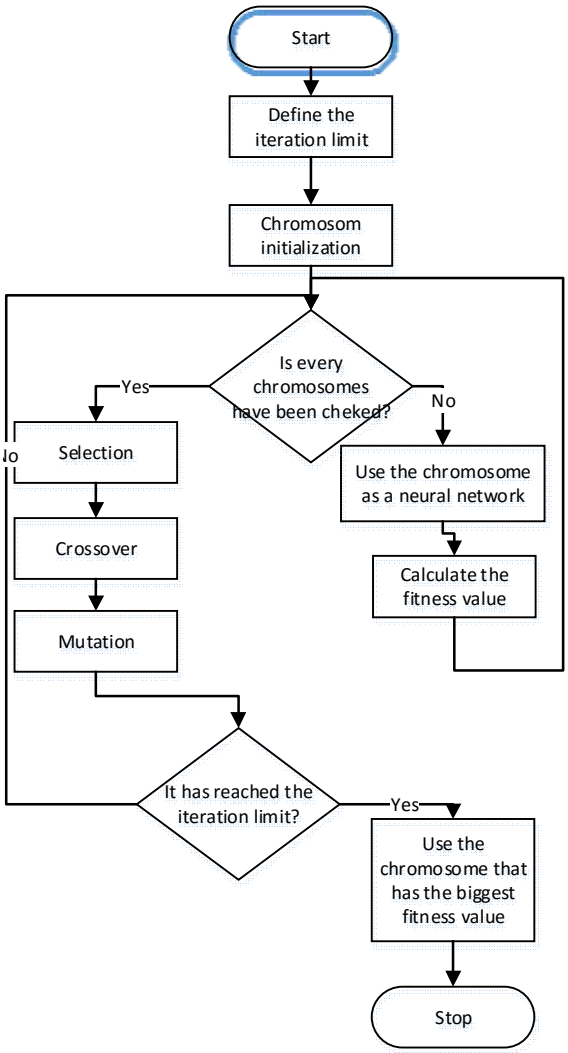

Figure 5. BPGANN Flow Diagram

Chromosome Initialization

In the initialization, each gen in a chromosome was filled with random binary. The length of the chromosome is

Input	A set of chromosome
Output	A set of chromosome after crossover

```
for i=0:populasi.Length:i+=2
    nextIndex ← i+1
    if nextIndex >= populasi.Length
     nextIndex ← i-1
    end
    bit1 ← chromosom[i].Bit
    bit2 ← chromosome[nextIndex].Bit
    for j=bit1.Length/2:bit1.Length:j++
      temp ← bit1[j]
      bit1[j] ← bit2[j]
      bit2[j] ← temp
      chromosom[i].Bit ← bit1
      chromosom[nextIndex].Bit ← bit2
```

Figure 8. Chromosome Crossover

Input	A set of chromosome
Output	A set of chromosome after mutation

```
for i=0:populasi.Length
    chanceChromoMut ← GetRandom()
    if chanceChromoMut <= GetRandom()
      bitIndex ← Random(chromo[i].Length)
      if chromo[i][bitIndex] == 0
        chromo[i][bitIndex] ← 1
      else chromo[i][bitIndex] ← 0
      end
    end
```

Figure 7. Chromosome Mutation

34 (the number of feature in this research). The pseudocode of chromosome initialization was shown in Figure 4.

Calculate the Fitness Value

Fitness value of each chromosome was computed using Equation (4). Fitness value was computed after the chromosome used in neural network (NN). The error rate of the NN was used as fitness value with TC is total correct of NN and TT is total testing data. The pseudocode of fitness value calculation was shown in Figure 6.

$$\boldsymbol{FV} = \frac{\boldsymbol{TC}}{\boldsymbol{TT}} \quad (4)$$

Crossover

If crossover occurs in two chromosomes A and B, each chromosome was divided in to two pieces, front side and back side. The front side of chromosome A was combined with the back side of chromosome B and the front side of chromosome B was combined with the backside of chromosome A. The pseudocode of crossover was shown in Figure 7.

Mutation

Chromosome mutation occurs randomly in random gen. Pseudocode of chromosome mutation was shown in Figure 8.

F. Rule Based BPGANN (RB-BPGANN)

In this research, the number of alphabets that were recognized are 26. These alphabets are grouped into three

Table 1 The rule for alphabet classification

Rule	If	Then	Member
1	Feature 5 **more or equal to** feature 6	`Classifier1.xml`	E, I, J, M, N, S, and T
2	Feature 5 **less than** feature 5 during calibration	`Classifier2.xml`	A, C, D, G, H, O, P, Q, X, and Y
3	Else	`Classifier3.xml`	B, F, K, L, R, U, V, W, and Z

groups. The first group is alphabets which the position of thumb finger on x-axis more or equal to the position of index finger on x-axis. These alphabets are E, I, J, M, N, S, and T. The second group is alphabets which the position of thumb finger on x-axis is less than the position of thumb finger on x-axis during calibration. These alphabets are A, C, D, G, H, O, P, Q, X, and Y. And the last group is alphabets which the index finger and the middle finger are vertical. These alphabets are B, F, K, L, R, U, V, W, and Z. The detail rule of this group was shown in Table 1.

Those all data training were grouped into three groups based on rule in Table 1. In the training process, BPGANN was applied for each group in order to generate classifier. The result of BPGANN in the first group was named `Classifier1.xml`, the result of BPGANN in the second and third group was named as `Classifier2.xml` and `Classifier3.xml` respectively. In testing process, when an instance of data come, the data was extracted and processed using Table1. If the data was suitable with Rule 1 then the data was classified using Classifier1.xml.

III. EXPERIMENT AND RESULTS

A. Experiment

In this experiment, 260 data set from 26 alphabets were used for training. Each alphabet consists of 10 data. These data set was collected from one user. In training, the interface for feature extraction was shown in Figure 9. While testing was done online. Each alphabet was tested five times. The example interface for testing was shown in Figure 10. In the learning process, the number of layer in NN is 3. The number of node in the input layer is dynamic depended on the result of GA. The number of hidden layer is a half of the number of node in the input layer. While the number of node in the output layer is depended on the number of class in a group, there are 3 nodes, 4 node, and 4 node for group one, group two, and group three respectively. In GA process, the number of chromosome in a population is 5. Crossover was done for each chromosome, for example: chromosome 1 was crossed over with chromosome 2, etc. For mutation, each gen in each chromosome has mutation probability that was generated randomly and the probability of mutation gen also generated randomly by system.

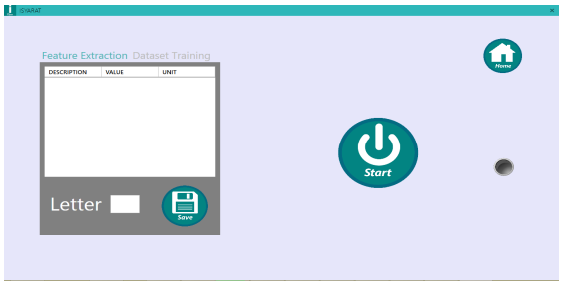

Figure 9. Interface for feature extraction

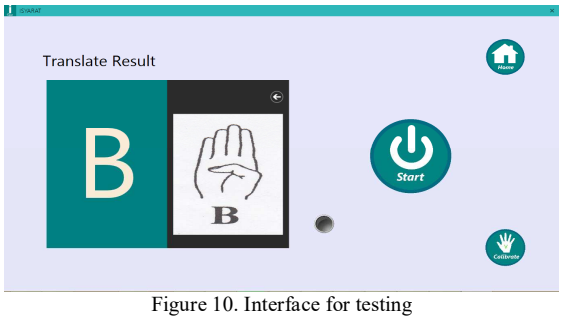

Figure 10. Interface for testing

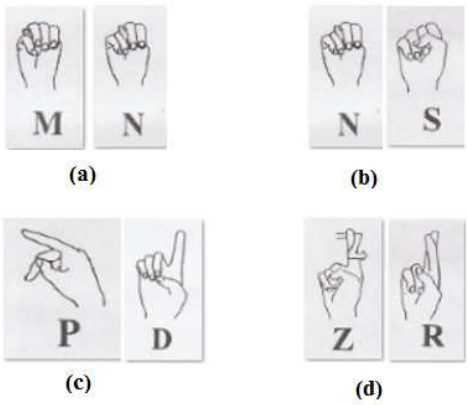

Figure 12. The Misclassified Alphabet

Table 2. System Accuracy

Iteration	Acuration
10000	83,84 %
15000	93,80 %
20000	87,60 %

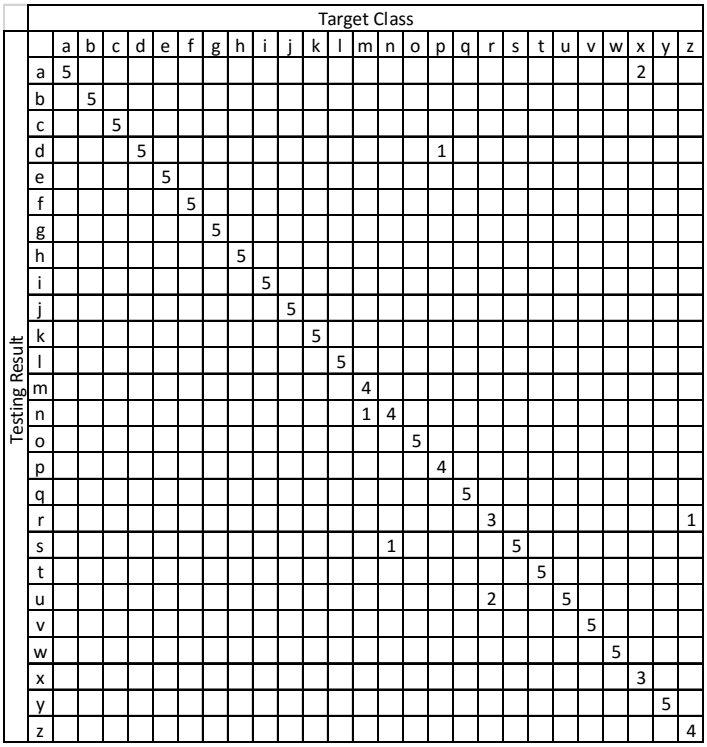

Figure 11. Confusion matrix of object recognition

B. Result

In this experiment, the classifier was created in different iteration. The result of testing with classifier in different iteration was shown in Table 2. From the table, it is shown that the best iteration of the classifier is 1500. The detail of the recognition result was shown in confusion matrix Figure 11.

IV. DISCUSSION AND CONCLUSION

From confusion matrix in Figure 11, it showed that the missed recognition was occurred for alphabet M, N, P, R, X, Z. From 5 times testing, alphabet M was misclassified as N, N misclassified as S, P misclassified as D, Z misclassified as R, one time. Two times misclassification out of five times testing occurs on alphabet R and X.

Some problems that could be become reason of these misclassifications are:

a) M misclassified as N because the sign of M and N are similar. It can be seen in Figure 12 (a). From that figure it can be shown that in M sign the position of thumb nail is between ring finger and little finger while in N sign, the position of thumb nail is between middle finger and ring finger.
b) The sign of N and S can be seen in Figure 12 (b). These signs are similar. The deferent is in N, the thumbnail is inside between the ring finger and middle finger, in S the thumbnail also between the ring finger and middle finger but from outside. Sometimes, Leap Motion cannot capture these difference.
c) Sometimes, P was misclassified as D because the distance among the fingers in those signs are similar. It can be seen in Figure 12 (c).
d) Generally, Z and R also have similar signs. The different only in Z the index finger is standing while in R the standing finger are not only index finger but also middle finger. The movement of index finger in 3D in Z signs cannot be captured perfectly by Leap Motion. This condition also become the reason of the misclassification.

REFERENCES

[1] M. Mohandes, M. Deriche e J. and Liu, "Image-Based and Sensor-Based Approaches to Arabic Sign Language Recognition," IEEE Transactions on Human-Machine Systems, 2014.
[2] K. Assaleh and M. Al-Rousan, "Recognition of Arabic Sign Language Alphabet Using Polynomial Classifiers," EURASIP Journal on Advances in Signal Processing, vol. 2005, no. 13, p. 507614, Aug. 2005.
[3] F. Fares, E. Fares e a. M. Othman, "System Based on Time Delay Neural Networks," Research Journal of Applied Sciences, Engineering and Technology, vol. 7, no. 11, pp. 2261-2265, 2014.
[4] R. Naoum, H. Owaied and S. Joudeh, "Arabic Sign Language Recognition Using K-Nearest Neighbor Algorithm," Journal of Emerging Trends in Computing and Information Sciences, vol. 3, no. 8, pp. 1173-1178, 2012.
[5] E. Mustafa and K. Dimopoulos, "Sign Language Recognition using Kinect," in Sixth International Conference on Contemporary Computing, Noida, 2013.
[6] H.-D. Yang, "Sign Language Recognition with the Kinect Sensor Based on Conditional Random Fields," Sensors, vol. 15, pp. 135-147, 2014.
[7] N. A. Ibraheem e R. Z. Khan, "Vision Based Gesture Recognition Using Neural Networks Approaches: A Review," International Journal of Human Computer Interaction (IJHCI), vol. 3, no. 1, 2012.
[8] M. Maraqa, F. Al-Zboun, M. Dhyabat and R. A. and Zitar, "Recognition of Arabic Sign Language (ArSL) Using Recurrent Neural Networks," Journal of Intelligent Learning Systems and Applications, vol. 4, pp. 41-52, 2012.
[9] N. Ulfah, "Detik Health," Detik, 8 January 2010. [Online]. Available: http://health.detik.com/read/2010/01/09/155558/1274969/. [Acesso em 26 December 2014].
[10] M. Mohandes, S. Aliyu, and M. Deriche, "Arabic sign language recognition using the leap motion controller," in 2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE), 2014, pp. 960–965.
[11] S. Wang, S. Yin e M. and Jiang, "Hybrid Neural Network Based On GA-BP for Personal Credit Scoring," em Fourth International Conference on Natural Computation, 2008.
[12] A. W. Yanuardi, S. Prasetio, and P. P. J. Adi, "Indonesian Sign Language Computer Application for the Deaf," in 2010 2nd International Conference on Education Technology and Computer, 2010, vol. 2, pp. V2–89–V2–92.