Real-Time Indonesian Language Speech Recognition with MFCC Algorithms and Python-Based SVM

Wening Mustikarini[1], Risanuri Hidayat[2], Agus Bejo[3]

***Abstract*—Automatic Speech Recognition (ASR) is a technology that uses machines to process and recognize human voice. One way to increase recognition rate is to use a model of language you want to recognize. In this paper, a speech recognition application is introduced to recognize words "*atas*" (up), "*bawah*" (down), "*kanan*" (right), and "*kiri*" (left). This research used 400 samples of speech data, 75 samples from each word for training data and 25 samples for each word for test data. This speech recognition system was designed using Mel Frequency Cepstral Coefficient (MFCC) as many as 13 coefficients as features and Support Vector Machine (SVM) as identifiers. The system was tested with linear kernels and RBF, various cost values, and three sample sizes (*n* = 25, 75, 50). The best average accuracy value was obtained from SVM using linear kernels, a cost value of 100 and a data set consisted of 75 samples from each class. During the training phase, the system showed a f1-score (trade-off value between precision and recall) of 80% for the word "*atas*", 86% for the word "*bawah*", 81% for the word "*kanan*", and 100% for the word "*kiri*". Whereas by using 25 new samples per class for system testing phase, the f1-score was 76% for the "*atas*" class, 54% for the "*bawah*" class, 44% for the "*kanan*" class, and 100% for the "*kiri*" class.**

***Keywords*—Automatic Speech Recognition, Indonesian Language, MFCC, SVM.**

I. Introduction

Research on ASR has been carried out for more than 40 years, but until now there are still studies to find ASR that can recognize speech in any subject by various languages. One way to increase recognition rate is by using a model from a language that needs to be recognized. Indonesian Language is as one of non-mainstream language that does not yet have speech corpus compared to other languages [1], [2], thus it influences speech recognition level in Indonesia.

In addition, devices that use voice input have been limited to languages other than Indonesian, for example, such as the S Voice virtual assistant application from Samsung which only accepts input in English, Mandarin and Korean, or Alexa from the Amazon Echo that receives input in English , France, Japan, Spain, and Italy.

In order for the voice recognition system to work better in Indonesian language, the machine needs to be taught first using the Indonesian speech corpus. However, automatic speech recognition requires a model from a related language, while not all languages have that model. Building a language model is a complicated process because it requires a lot of speech from many speakers. Based on the explanation, it can be concluded that a speech recognition system using the Indonesian language model is needed to improve recognition system performance in recognizing words in Indonesian language.

[1] *Department of Electrical Engineering and Information Technology, Faculty of Engineering, Gadjah Mada University Jl. Grafika No.2, Yogyakarta, 55281 INDONESIA (tlp/fax:0274-552305/547506; e-mail: wening.mustikarini@mail.ugm.ac.id)*

[2, 3] *Lecturer, Department of Electrical Engineering and Information Technology, Faculty of Engineering, Gadjah University Jl. Grafika No.2, Yogyakarta, 55281, INDONESIA (tlp/fax:0274-552305/54750)*

II. Automatic Speech Recognition (ASR)

Automatic Speech Recognition (ASR) is commonly used to convert speech into texts. In addition, ASR is also used for biometric authentication, which authenticates users from their voice. From the identification or recognition process, ASR can be used to perform a task based on recognized instructions.

In order to work properly, ASR requires a configuration or voice that has been saved from the user. Users need to train ASR or machines by storing speech patterns (features) into the system. To obtain these features, data processing (feature extraction) is needed so that a value that represents information contained in the data is produced. In addition, methods are also needed to recognize those features.

A. Feature Extraction with Mel Frequency Cepstral Coefficient (MFCC)

At the training phase, the system goes through a learning process in order to recognize words. To be able to do this, the system requires information in a form of a word pattern obtained from feature extraction. Feature extraction in speech recognition is a computation of the voice signal to produce a features vector that represents the signal. These features are then compared with the test data features. In this paper, Mel Frequency Cepstrum Coefficient (MFCC) was used for feature extraction of 13 coefficients.

At present, MFCC is most commonly used in speech recognition and speaker verification because MFCC can work well on inputs with a high level of correlation, i.e., by removing information that is not needed. In addition, MFCC can represent human voice and music signals well because MFCC uses mel frequency [3].

The mel scale itself is an association between frequency (tone) heard or perceived by humans with actual measured frequency [4]. Equation (1) defines a relationship between mel frequency and an actual frequency. Instead, to obtain a frequency value from mel scale, (2) was used.

$$M(f) = 1125\, ln(1 + \frac{f}{700}) \quad (1)$$

$$M^{-1}(m) = 700(e^{\frac{m}{1125}} - 1) \quad (2)$$

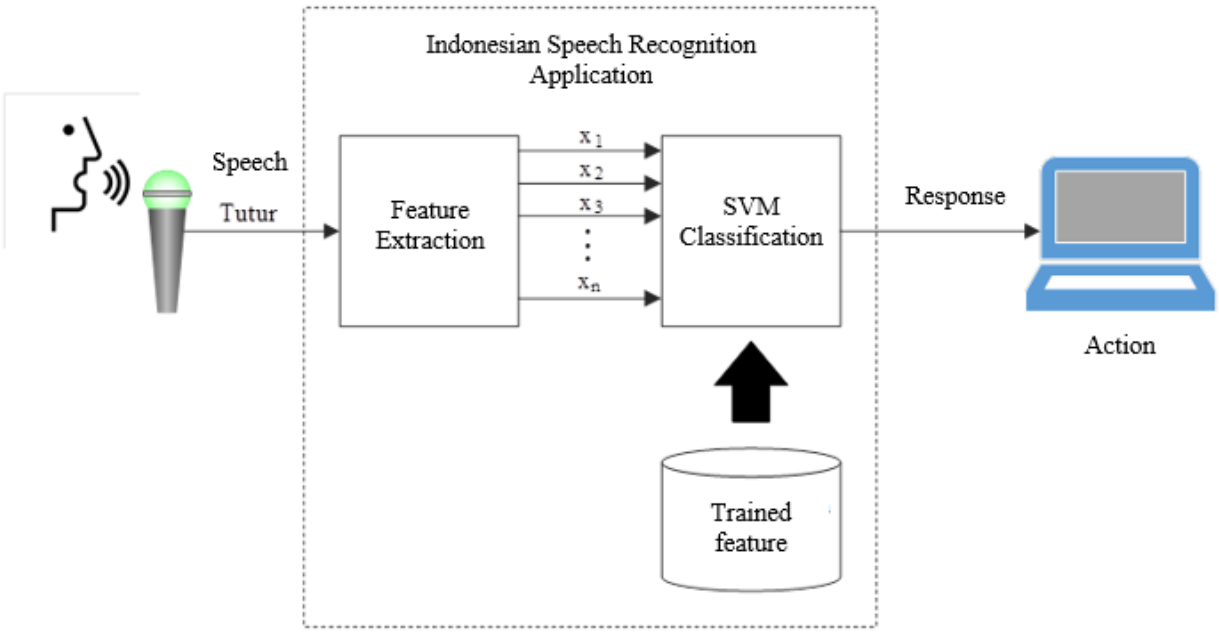

Fig. 1 ASR system design.

B. Support Vector Machine (SVM)

SVM was developed by Bernhard, Vapnik, and Guyon in 1991 and is a binary identifier, namely SVM that divides data into two classes at a time. These classes are negative or positive, so SVM is generally represented in two dimensions or three dimensions. The purpose of SVM algorithm is to find the best hyperplane that can separate classes and maximize margins (distance between support vectors of each class and hyperplane) so that two classes increasingly visible apart.

In two dimensions (two feature vectors), hyperplane is in a form of a line, whereas in third dimension, a hyperplane is a field. If p is the feature dimension, then hyperplane's dimensions are $p-1$ Hyperplane is defined as (3) [5].

$$f(x) = y = x^T \cdot \beta + \beta_0 = 0 \quad (3)$$

For a higher dimension, hyperplane can be described as (4).

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p = 0 \quad (4)$$

with

$f(x)$ = distance of support vector to hyperplane,
β_p = weights of p dimentional hyperplane,
x_p = feature vector,
β_0 = bias.

Hyperplane serves to separate data between classes by quantifying or observing similarities between two data. To observe similarities between two data, i.e., true data and predictable data, inner product between two vector-shaped data can be used.

$$\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j} \quad (5)$$

with

x_i = feature vector of training data,
$x_{i'}$ = feature vector of test data,
i = sample number, therefore i' is test data sample,
j = number of utilized features.

Therefore, SVM linear can be defined as (6).

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle \quad (6)$$

with

x = new feature vector, that is, data that has not been classified,
x_i = feature vector of test data,
β_0 = bias.

To find a solution from $f(x)$, only inner product generalization is needed in (5) in a form of a kernel function, $K(x)$.

$$K(x_i, x_{i'}) \quad (7)$$

Basically, kernel serves as hyperplane, therefore (6) becomes (8).

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i K(x_i, x_{i'}) \quad (8)$$

III. METHODS

In this paper an Indonesian language recognition system was developed using machine learning utilizing an MFCC method to extract features and SVM for classification. This system's working mechanism is shown in Fig. 1. System received real-time input in a form of speech in Indonesian language and processed it into an order in a computer with a Windows operating system.

This speech recognition system design was carried out in several phases, namely data acquisition, feature extraction, feature classification (identifiers training), identifier testing, and recognition system implementation as shown in Fig. 2.

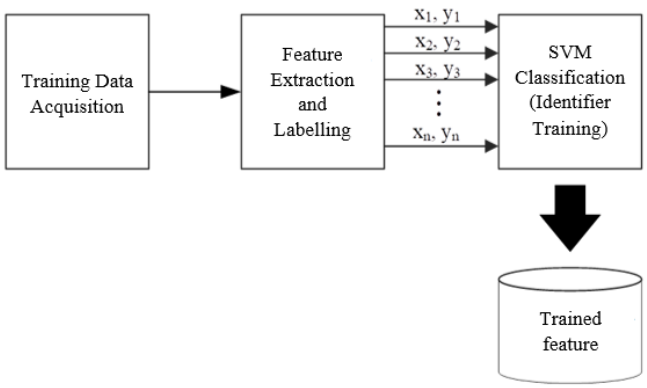

Fig. 2 Flowchart for speech recognition.

A. Data Acquisition

The first process of the system was speech signals acquisition as input. System had to be able to distinguish signals containing information with noise signals so that the processed signals were in a form of relevant information.

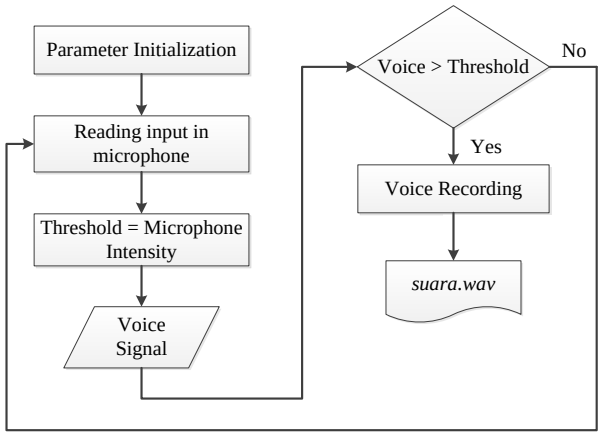

Fig. 3 Flowchart for speech acquisition.

As shown in Fig. 3, this system used a threshold value that distinguished speech signals with noise signals. That threshold value was obtained from calculating average signal frequency intensity read by microphone.

Acquisition of testing data and train data (offline) underwent identical processes as in Fig. 3. Training data and test data were stored in folders according to their class respectively. It aimed to label each sample. While real-time test data was only given one file name so that in the next test data acquisition, the new data would overwrite or replace the old data. Table I shows number of acquired samples. There were 75 samples from each class for training data and 25 samples from each class for recognition test data.

B. Feature Extraction

In this system design, MFCC was utilized extract features from each sample that would be entered into the identifier. In MFCC, a high-pass filter was used with a filter coefficient of 0.97, signals frame length of 25 ms, Han window for scheduling, and 512 Fourier transformation points.

TABLE I
CLASS IN BINARY

No.	Class/Label	Number of samples	
		Training Data	Testing Data
1.	*Atas*	75	25
2.	*Bawah*	75	25
3.	*Kanan*	75	25
4.	*Kiri*	75	25
Total		300	100

In addition, labelling was also carried out for each sample. This process produced a $(n \times 13)$ size feature matrix with n was number of used samples.

As for labelling, a label vector was generated containing classes of each sample namely "*atas*", "*bawah*", "*kanan*", and "*kiri*". That label vector could also be converted into a label matrix containing classes in binary, as shown in Table II.

TABLE II
CLASS IN BINARY

Class	Class Vector
Atas	[1 0 0 0]
Bawah	[0 1 0 0]
Kanan	[0 0 1 0]
Kiri	[0 0 0 1]

C. Feature Classification (Identifier Training)

SVM model training used training data of 300 samples or 75 samples from each class. From feature extraction process a (300×13) feature matrix was produced.

Grid search with 10-folds cross validation was employed to find hyper parameter of the best SVM model as well as to estimate model's actual capabilities. The utilization of 10-folds cross validation aimed to produce a model that had a low variance and bias so that the model did not experience overfitting.

In grid search, hyper parameters of several kernel types were tested, various cost values (hyperplane location errors on training data, always positive and greater than the number of tolerated errors), and gamma values. After that, the hyper parameter was observed which produced the best precision value or f1-score (trade-off value of precision and recall).

In addition, identifiers were also tested with three sample sizes, namely $n = 25$, $n = 50$, and $n = 75$. Of those three sample sizes, a sample size producing the best f1-score was observed.

IV. RESULTS AND DISCUSSION

A. SVM Identifier Optimization

In training phase, there were several parameters that can be changed to obtain the best classifier. Those parameters are as follows.

- Linear kernels and RBF.
- Cost values or *C*, with the variation was 0.1; 1; 10; 100 and 1,000 for linear kernels, and 1 1; 10; 100 and 1,000 for RBF kernels.
- Gamma values varied in values from 0.001 and 0.0001.

After grid search was performed, average accuracy value of each parameter was obtained with each considering precision and recall. Precision values were obtained with 10-folds cross validation, as presented in Table III and Table IV.

TABLE III
ACCURACY AVERAGE FROM PRECISION OPTIMIZATION PRODUCED BY RBF KERNEL

γ \ C	1	10	100	1,000
0.001	0.519	0.519	0.894	0.896
0.0001	0.519	0.519	0.519	0.894

TABLE IV
ACCURACY AVERAGE FROM PRECISION OPTIMIZATION PRODUCED BY LINEAR KERNEL

C = 0.1	C = 1	C = 10	C = 100	C = 1,000
0.894	0.895	0.909	0.931	0.946

Table III and Table IV each show the effect of kernel types on average value of accuracy when precision was optimized. With RBF kernel, the obtained highest accuracy average was of 0.896 with a value of C = 1,000 and gamma = 0.001. Whereas with the linear kernel, the highest average accuracy was 0.946 with a value of C = 1,000.

Apart from being observed in a state of optimal precision, accuracy average was also observed when recall value was optimal, as shown in Table V and Table VI.

TABLE V
ACCURACY AVERAGE FROM RECALL OPTIMIZATION PRODUCED BY RBF KERNEL

γ \ C	1	10	100	1,000
0.001	0.602	0.602	0.843	0.856
0.0001	0.602	0.602	0.602	0.843

TABLE VI
ACCURACY AVERAGE FROM RECALL OPTIMIZATION PRODUCED BY LINEAR KERNEL

C = 0.1	C = 1	C = 10	C = 100	C = 1,000
0.852	0.890	0.921	0.921	0.934

Based on Table V and Table VI, with RBF kernel, the obtained highest accuracy average value was of 0.856 with a value of C = 1,000 and gamma = 0.001. Whereas with linear kernel, the highest average accuracy was 0.934 with a value of C = 1,000.

Similar to precision optimization, from Table V and Table VI it can be concluded that linear kernel with a C value of = 1,000 produces the best identifier. Therefore, linear kernel with a value of C = 1,000 is used.

In addition to finding the best hyperparameter, SVM optimization could also be done by repairing and converting datasets. Seeing that the system's purpose was to be able to read as many as possible true positive or had the highest precision value possible, then a dataset combination producing precision trade-off and recall was sought or searching for the number of dataset producing the highest f1-score. Table VII shows effects of sample sizes on f1-score.

TABLE VII
EFFECTS SAMPLE SIZE ON F1-SCORE

Class	Number of samples per class		
	n = 25	n = 50	n = 75
Atas	75%	86%	80%
Bawah	85%	79%	81%
Kanan	88%	83%	86%
Kiri	100%	100%	100%
Average	87%	87%	87%

From Table VII it can be concluded that dataset containing 75 samples per class produces the best identifier performance, so that it is used for training.

B. Identifier Testing

After training the identifier with a dataset containing 75 samples per class with the best hyper parameter, i.e., with linear kernel with a value of C = 1,000, identifier was tested with unknown data with 25 data samples for each class or 100 new data samples. Table VIII shows values of precision, recall, and f1-score produced by that identifier to recognize 100 new data samples.

TABLE VIII
IDENTIFIER PERFORMANCE IN NEW DATA

Class	Classification Result		
	Precision	Recall	F1-score
Atas	61%	100%	76%
Bawah	100%	28%	44%
Kanan	52%	56%	54%
Kiri	100%	100%	100%
Average	78%	71%	68%

From Table VIII, it can be seen that there is a decrease in the values of precision, recall, and f1-score compared to identifier optimization phase. During identifier optimization phase, the utilized testing data were from same dataset as training data, therefore identifier testing result showed a high value. However, in testing with unknown data, the identifier did not have information about the data so that it produced a lower value.

Table VIII shows that "*bawah*" and "*kanan*" classes have the lowest f1-score. A low f1-score indicates that only a few true positive are recognized by a system. It was caused by the "b-" and "-h" sounds in word "*bawah*" and "k-" and "-n" sounds in word "*kanan*" sounds were often unreadable by the system. It can be seen from MFCC comparison of "*bawah*" and "*kanan*" words in training phase and testing phase with new data in Fig. 4. The low F1 results caused misclassification because the data was recognized as another class.

C. Speech Recognition System Implementation

GUI that has been designed and connected with processes occuring at the back-end was executed. Users must press

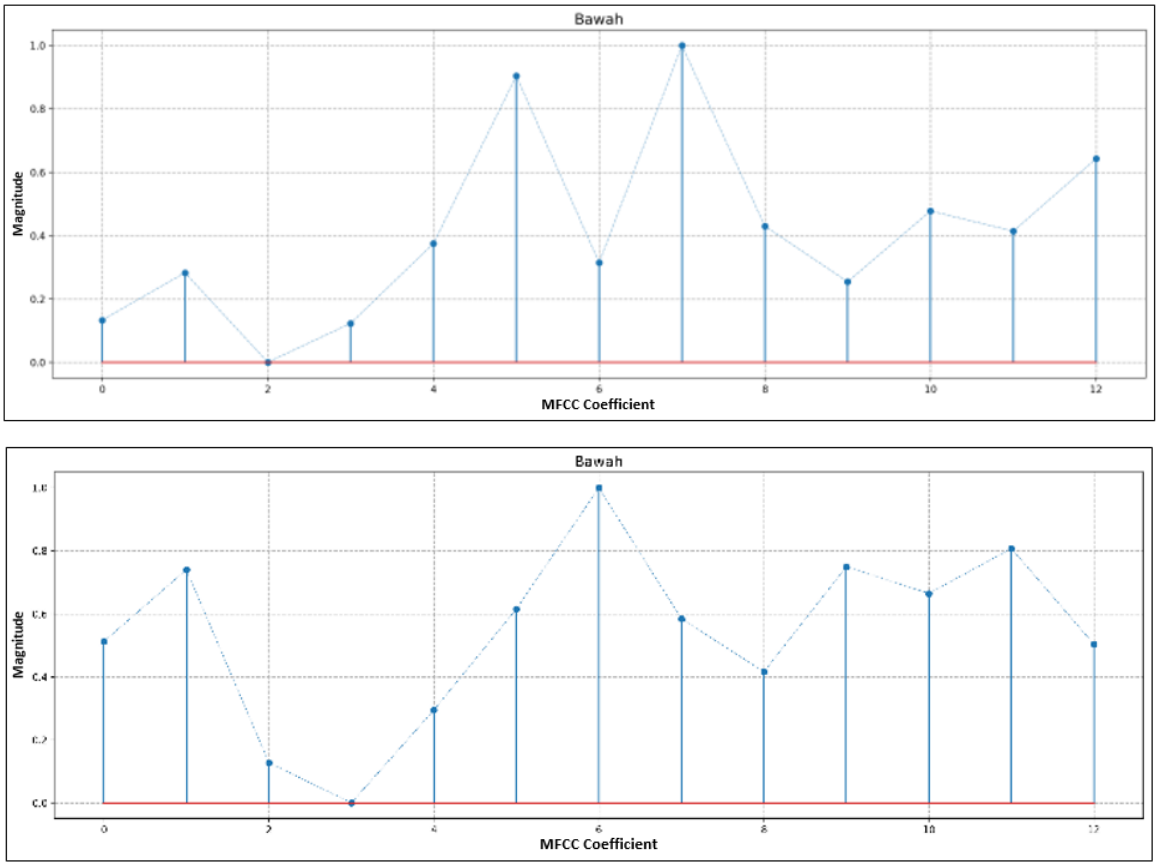

Fig. 4 MFCC Comparison of "*bawah*" word in training data and new data.

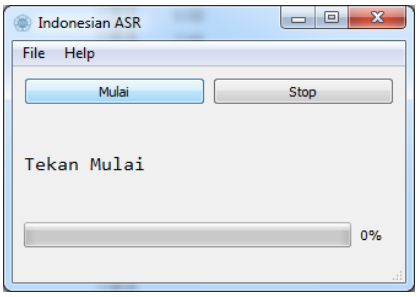

Fig. 5 GUI interface when it just started.

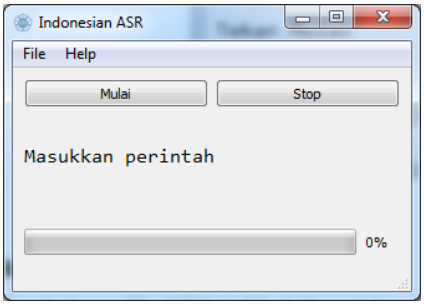

Fig. 6 GUI interface after "*Mulai*" button is clicked.

"*Mulai*" button as shown in Fig. 5. After that, the GUI displays the text so that the user says a word as shown in Fig. 6.

After inputting speech signals, GUI calls the stored identifiers. Fig. 7 shows output i GUI after system recognizes

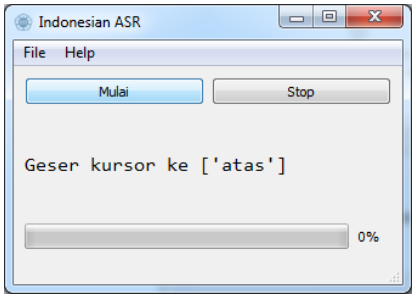

Fig. 7 GUI interface after recognizing word "*atas*".

word "*atas*". GUI will display "*Geser krusor ke [kata yang dikenali]*" (slide crusor to [recognized word]) text. GUI will continue to run in the background until the "Stop" button is pressed.

V. CONCLUSION

Generally, this paper described a course of making real-time speech recognition system in Indonesian language. Words recognized by the system are "*atas*", "*bawah*", "*kanan*", and "*kiri*". Speech recognition used MFCC as a feature extraction method and SVM as an identifier.

The SVM model's training utilized a voice dataset containing speech signals of 300 data samples or 75 samples for each class. From identifier optimization process, the best hyper parameter was obtained in a form of linear kernel with a cost value of

1,000. At the training phase, the obtained f1-score was of 80% for "*atas*" word, 81% for "*bawah*" word, 86% for "*kanan*" word and 100% for "*kiri*" word.

To test the SVM model, a dataset of 25 new data was utilized for each class. From the testing, the obtained f1-score was of 76% for "*atas*" class, 44% for "*bawah*" class, 54% for "*kanan*" class and 100% for "*kiri*" class.

REFERENCES

[1] K. Precoda, "Non-mainstream Languages and Speech Recognition: Some Challenges," *CALICO Journal*, Vol. 21, No. 2, pp. 229-243, 2004.

[2] E. Cahyaningtyas and D. Arifianto, "Development of Under-resourced Bahasa Indonesia Speech Corpus," *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, 2017, pp. 1097-1101.

[3] A. Winursito, "Peningkatan Akurasi Pengenalan Tutur Vokal Bahasa Indonesia Menggunakan Algoritma MFCC PCA/SVD," Magister thesis, Universitas Gadjah Mada, Yogyakarta, Indonesia, 2018.

[4] S.S. Stevens, J. Volkmann, and E.B. Newman, "A Scale for the Measurement of the Psychological Magnitude Pitch," *Journal of the Acoustical Society of America*, Vol. 8, No. 3, pp. 185-190, 1937.

[5] T. Hastie, R. Tibshirani, and J.H. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, New York, USA: Springer, 2001.