Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871

Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

Identification of Dominant Frequencies in Javanese Vocal Phonemes Using Fast Fourier Transform and Random Forest Classification

Muhadi*[1]

[1]Department of Master of Science and Information Technology
President University Bekasi, Indonesia

Email: [1]Muhadi0808@gmail.com

Received : Aug 26, 2024; Revised : Mar 1, 2025; Accepted : Mar 3, 2025; Published : Apr 26, 2025

Abstract

The majority of speech recognition research currently uses English as the research base, but the results can also be used for another language, including Javanese speech recognition. Previous research stated that there were differences in frequency between English and Dutch. This shows that the frequency of Javanese can also be different. The difference in frequency allows for a new way of recognizing Javanese Speech. By using a dataset of Javanese vowel phonemes, this research aims to identify the dominant frequencies in Javanese speech using the Fast Fourier Transform data extraction an2d the Random Forest Classifier. The feature importance level data will be tested with a deep neural network to determine the accuracy and speed of the process. Choosing a dominant frequency is expected to make the process more effective and efficient in using computing resources.

Keywords : *Deep Neural Network, Fast Fourier Transform, Javanese Phoneme, Random Forest Classifier, Speech to Text.*

1. INTRODUCTION

The majority of speech recognition research always uses English basically, this is in accordance with the language mastered by the researchers themselves and also because English is the language most widely used by humans in the world. Even though the research was developed in English, the results of the technology can be used for voice recognition in other languages, including Indonesian and even Javanese.

The existing voice recognition technology can be used for Javanese, but it is possible that Javanese have other methods with better accuracy, lighter resources, and faster processing. Mathilde Theelen stated in 2017 that there are frequency differences between English and Dutch [07]. This shows that different languages can produce voices with different basic frequencies. Based on the results of this research, it is very possible that Javanese also has a different dominant frequency from English. The difference in frequency makes it possible to carry out research which will hopefully result in new ways to Javanese Speech Recognition more quickly and efficiently.

This journal presents an in-depth investigation aimed at enhancing the Javanese speech-to-text technology by applying a novel analysis of the frequency of key vocal phonemes. This method utilizes distinct acoustic characteristics to address specific challenges related to the Javanese language, offering potential benefits for speech recognition technologies in other languages with limited resources. The study focuses on detecting prominent frequencies in Javanese words by employing the Fast Fourier Transform (FFT) to convert speech data into the frequency domain. By applying feature importance techniques from the Random Forest Classifier, we can select the most influential frequencies. This method enables the reduction of training data volume without sacrificing accuracy, making the process

more effective and efficient in terms of computational resource usage. The low computing system requirements resulting from this approach allow speech-to-text processes to run on IoT platforms and embedded systems.

Previous research [06] is trying to overcome the current problem where speech recognition is very difficult to implement widely and efficiently. This is because the current speech recognition system requires quite a large amount of computing power and memory, so it costs quite a lot to implement. The researcher tries to reduce the computational process by changing the deep learning configuration to a minimum, where the architecture utilizes a three-layer model with 130 neurons, 64 inputs, a softmax activation function, Adam optimization, a dropout rate of 0.2, and a batch size of 64. This method can still achieve an accuracy of 74.06 % for training and 71.68 % for validation.

Research [03] aims to explore the application of MFCC-based speech-to-text technology, incorporating FFT features, to develop a speech-to-text application for patient complaints in Bahasa Indonesia during medical histories. TensorFlow is utilized specifically for Bahasa Indonesia within the medical context. The voice dataset used in this study encompasses not only Indonesian words but also various complaints expressed in Javanese. The implementation of speech-to-text aims to accurately transcribe common patient complaints during medical examinations. From a dataset comprising 500 voice recordings from 10 individuals, an accuracy of 64% was achieved. Additionally, similarity calculations were conducted to assess the alignment between spoken words and the generated text. Based on these similarity assessments, the generated text data was categorized into three groups: good, fair, and poor.

Research [10] focuses on Javanese speech recognition using a dataset of male and female voices from the Javanese tribe, processed with noise reduction preprocessing and MFCC feature extraction. Classification employs Random Forest and Neural Network methods. The study involves 150 WAV files recorded for "eating," "drinking," and "sleeping," each spoken 10 times by 5 men and 5 women. Evaluation results show accuracies of 91.3% using Random Forest and 92.2% using Neural Network for Javanese accent speech recognition.

Research [09] presents the development of a hybrid corpus-based machine translation system for the Javanese language, focusing on its complexity in politeness expression and speech levels. The system integrates statistical features and edits shifting distance to enhance alignment efficiency. The study proposes an improved alignment algorithm based on impossible pair restrictions, achieving higher accuracy (93.8%) despite fewer training data compared to the previous model (87.9%).

Research is needed to explore alternative methods for processing Javanese speech into text that reduce the amount of training data required without compromising accuracy. This can be achieved by focusing on training models using datasets that capture the most dominant frequencies in each allophone with a limited number of samples. Utilizing the feature importance technique of the Random Forest Classifier, we can rank and select the frequency data based on their influence on allophone variations. This approach aims to maintain high recognition accuracy while significantly reducing the computational resources and data volume necessary for effective model training. This method was not found in previous research.

Based on research[04], Javanese vocal phonemes consist of 6 pieces: /i, u, e, ð, o/, a/. Of the six vocal phonemes, based on the height of the tongue when pronounced, they can be divided into 3 types, namely high vowels (/i, u), middle vowels (/e, ð, o/), and low vowels (a/). If divided based on the distance between the tongue and the palate or the structure when the phoneme is pronounced, the phoneme is divided into 4 parts, namely: closed (/I, e/), semi-closed (/e, o/), semi-open (ð) and open (a/) as shown in Table 01.

Table 1. Vowel Phoneme

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

		Moving part of the tongue				
		Front	Central	Back		
Tongue hight	Hi	i		u	Close	Between tongue and palate
	Middle	e		o	Close-Mid	
			ð		Open-Mid	
	Low		a		Open	
		UnRound		Round		
		Lip Shape				

From the 6 vocal phonemes, if we develop them into allophone form, they will become 10 allophones : [i], [I], [u], [U], [e], [E], [ð],[o],[ɔ],[a]. Figure 01 and Table 02 show details about the allophones and the example words for each

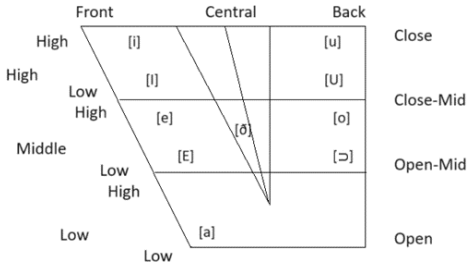

Figure 1. Javanese Vocal Phoneme Allophones

Table 2. Allophones

No	Allophones	Words
1	[i]	iki, dina,timba,amis, pait
2	[I]	alis, balik,sing, ting, kaping, garing
3	[u]	ulah, uwong, pundak, buku, aduh
4	[U]	ayun, getun,sanggul, mung, umuk
5	[e]	enak. edan,bale,gempor
6	[E]	jengkel,waneh,meri,kalen
7	[ð]	pendopo, emas, sego, metu.lemu
8	[o]	loro, kono, obah, olah
9	[>]	pong, obong, gentong, lori,sapa
0	[a]	akal, banter, bakat, bali

The foundational pieces required to develop voice recognition systems are provided by the Python package Librosa, which is designed for music and audio analysis. The Fast Fourier Transform (FFT) method is a crucial tool recognized for its pivotal role in various audio analysis applications, including feature extraction, noise reduction, and audio visualization. Librosa is widely used for audio analysis due to its comprehensive toolkit and user-friendly interface. It supports various tasks including signal processing, feature extraction, and audio data handling, making it a versatile tool for creating and testing audio analysis applications. Its intuitive design caters to both beginners and experienced users alike.

The Fast Fourier Transform (FFT) is a computational algorithm utilized to compute the discrete Fourier transform (DFT) or its inverse (IDFT). The DFT is a mathematical technique used to convert a

Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

function's representation from the time or space domain into its corresponding frequency domain., and vice versa. This method finds extensive applications across diverse fields including signal processing, image processing, audio analysis, and various other disciplines. The FFT is particularly important because it drastically reduces the number of computations needed to perform the DFT, especially for large input sizes.

Fast Fourier Transform (FFT), frequently abbreviated as FFT, is a fundamental algorithm utilized in digital signal processing to compute discrete Fourier transforms quickly and efficiently. Fast Fourier Transform currently shares fields that apply this technology, such as the field of telecommunications, image processing, and audio signal analysis. The Fast Fourier Transform (FFT) is a highly efficient algorithm used to compute the Discrete Fourier Transform (DFT) and its inverse, significantly reducing computational complexity from O(N^2) to O(N log N).

Feature importance techniques evaluate the significance of input features in predicting the target variable. These methods assess the contribution of each feature to the decision-making process within the Random Forest (RF) classifier. This review explores the concept of feature importance in RF, the methods for determining it, and its various applications and limitations. Feature importance can be employed to reduce dimensionality and potentially enhance model performance by identifying and selecting the most relevant features in high-dimensional datasets. This is especially advantageous in disciplines such as genomics, where the quantity of features (genes) can be substantial.

2. METHOD

This research proposes to find the dominant frequency using the Fast Fourier Transform and Random Forest Classifier. An overview/block diagram of the outcome of this research is illustrated in Figure 02. In Figure 1, we can see the audio features that will be extracted from the dataset using Librosa and Fast Fourier Transform (FFT). Then, the Random Forest Classifier is applied for extraction to determine the dominant frequency from the dataset that influences allophone changes.

The first 20 feature importance results from the random forest classifier will be compared with all data from the original dataset using deep learning. This is to show that using 20 dominant frequency data will have a faster processing time. All processes will be done in the google colab.

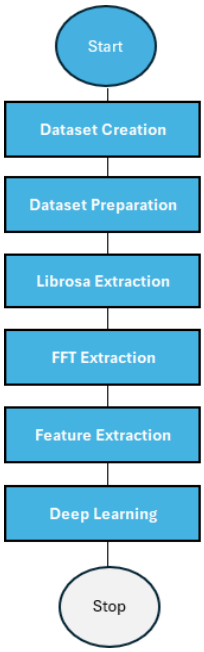

Figure 2. Diagram Block

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

2.1. Voice Dataset

The dataset consists of 800 vocal phonemes recorded using smartphones from 30 male and female individuals. There are two types of sentences recorded, the first is specific sentences, and the second is words referring to the 10 allophones in Javanese. The two types of voice data are then cut into pieces manually and then labeled.

The initial sentence before the cutting process was: ‘ jengkel banget dino iki kudu ngolah sego sing enak tapi mung kaya ngene’. After cutting process this sentence into syllables, it can represent the 10 allophones in Javanese. Figure 03 shows the first type of audio signal before the cutting process, and Table 03 shows a list of truncated syllables and their allophones. Figure 04 shows the second type of audio signal before the cutting process. All cutting processes are performed using Audacity software.

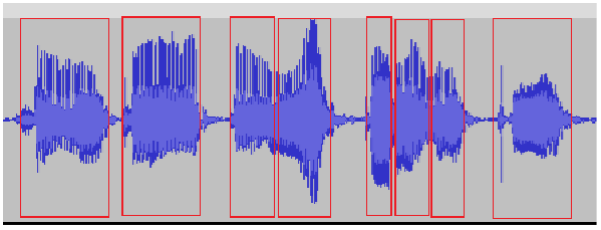

Figure 3. Sounds Files Before Cutting

Table 3. Syllables and Allophones

Syllables	Allophones
jeng, kel	[E]
ba, lah, nak, ta	[a]
nget, se	[ð]
di, i, ki, pi	[i]
no, go, ka, ya	[>]
ku, du,	[u]
ngo	[o]
sing	[I]
e, nge, ne	[e]
mung	[U]

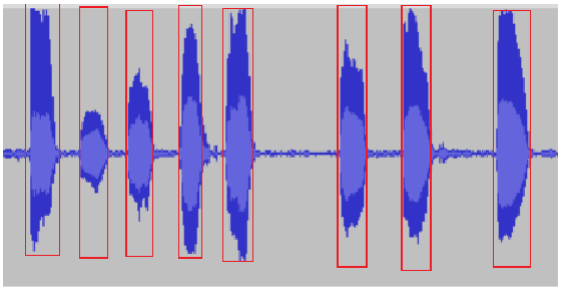

Figure 4. Allophone Sound File Before Cutting

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

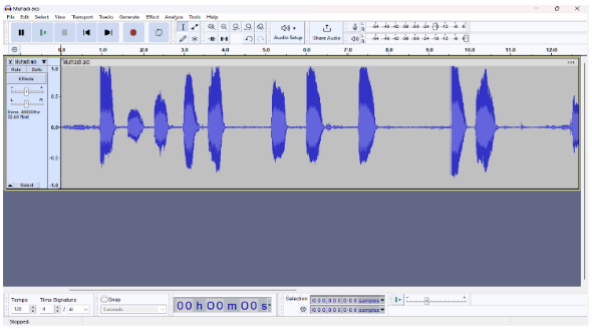

Figure 5. Audacity Software for Audio Cutting

Figure 05 shows that the cutting process is carried out using Audacity software. The results are saved in a certain file format, which can show the allophone ID and also the sound source ID

2.2. Librosa and Fast Fourier Transform (FFT)

The first step taken in the FFT process is to convert the dataset voice file into array data using the Librosa Module. One of Librosa's functions is to load audio files and convert them into data arrays. This array represents the audio signal as a time series, where each element of the array corresponds to an audio sample. In Python, the task to do this work can be done using the function: librosa.load(). Figure 06 shows the Python code for librosa extraction:

```
import librosa

nmfile = '/content/drive/MyDrive/dataset/fonem/se_3_3_2_1_data.mp3'
y, sr = librosa.load(nmfile)
```

Figure 6. Librosa Code

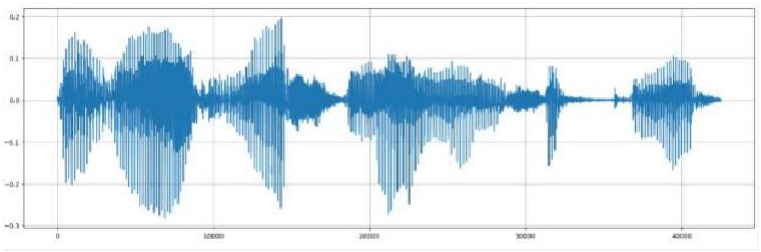

Figure 7. Librosa Extraction

Next, the process involves performing a Fast Fourier Transform (FFT) action to process of converting the signal from its time domain representation to the frequency domain. Figure 06 displays the voice extraction data in the time domain, while Figure 08 illustrates the voice data in the frequency domain after applying the FFT process using Python coding as depicted in Figure 07. After the FFT process, all data will be collected in data frame format as shown in Figure 09. This research uses FFT with several parameters as follows:

Tn: Sample rate = 22050
T: Sampling Internal = Tn/Sr = 2048 / 22050
L: Total Number Sample = 2048
t: Time Vector = 1/Sr = 1/22050

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

The amount of data is limited to 200 data from a frequency of 10 Hz to a frequency of 2153 Hz. Frequencies below 2 kHz cover a significant portion of the essential range of human speech, which includes the most important elements for vowel sounds and certain consonants. This can be beneficial for applications such as voice recognition, linguistic research, or vocal analysis, where higher frequencies may have less significance.

```
from numpy.fft import fft

tX = fft(y)
y =  np.abs(tX)

tN = len(y)
tn = np.arange(tN)
T = tN/sr
f = tn/T
```

Figure 8. FFt Process in Python

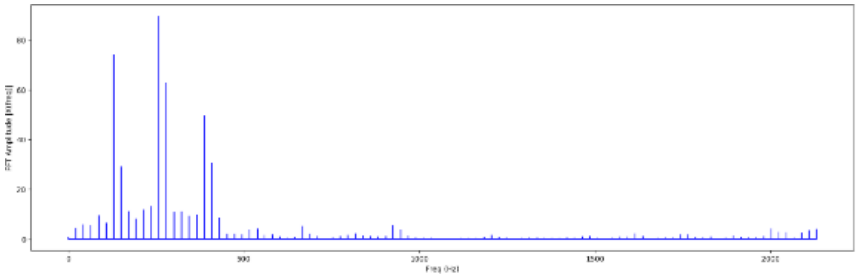

Figure 9. Data Extraction After The FFT Process

Figure 10 shows the entire dataset after the FFT process, which consists of 804 rows of data and 201 columns of data, including the label column.

	10	21	32	43	53	64	75	86	96	107	...	2077	2088	2099	2110	2121	2131	2142	2153	allophones
0	2.2	3.3	3.9	10.8	13.7	12.9	14.6	16.8	6.0	9.1		4.4	14.9	10.1	9.1	5.6	3.9	5.8	4.1	6
1	1.9	2.9	3.1	5.2	6.5	9.9	11.1	7.1	5.0	14.1		0.2	0.3	2.3	1.6	0.8	1.6	2.0	1.4	6
2	6.8	6.8	6.7	5.0	4.1	4.5	13.3	45.7	49.8	38.4		2.3	0.8	1.4	0.9	6.2	8.7	4.2	2.5	0
3	2.7	2.4	2.1	7.8	5.1	12.5	26.8	2.8	9.4	10.2		1.4	1.5	1.6	1.1	0.9	1.9	1.2	1.2	7
4	3.9	2.5	7.4	8.2	10.8	9.9	7.9	9.2	4.9	3.0		2.5	1.0	2.9	4.0	2.8	0.6	0.5	1.9	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
799	9.9	9.6	9.3	10.4	10.2	10.1	10.6	11.3	12.3	11.8		0.2	0.3	0.3	0.3	0.3	0.2	0.2	0.2	3
800	0.7	0.7	0.5	0.9	0.7	1.9	1.0	1.7	2.4	3.9		4.3	7.5	10.9	24.5	33.0	19.7	37.0	15.6	5
801	6.4	6.5	6.4	6.1	7.6	6.3	7.0	6.1	8.2	7.9		0.4	0.3	0.5	0.4	0.4	0.3	0.6	0.4	7
802	2.7	3.0	3.2	3.1	2.5	5.0	4.3	4.7	6.4	14.6		0.0	0.0	0.1	0.0	0.0	0.0	0.1	0.0	8
803	1.7	0.5	1.6	0.6	0.8	0.8	1.8	1.4	0.7	2.6		1.0	1.4	1.0	1.7	0.3	1.6	0.1	0.8	0

804 rows × 201 columns

Figure 10. Dataset After The FFT Process

2.3. Random Forest Classifier

The Random Forests method utilizes an ensemble of decision tree classifiers, with each tree being trained on a bootstrap sample of data points. At each node, the tree splits based on a randomly dominant subset of attributes. Classification is determined by aggregating the predictions from all trees within the

Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

forest. To construct each tree in the forest using a dataset with N data points and M attributes, the process is as follows:

1. Randomly pick N data points from the entire dataset with replacements to create a training sample.
2. At each node in the tree, choose a random subset of m attributes from the full set of M attributes in the dataset. The specific number of m is set in relation to the total number of attributes and stays constant during the forest-building process.
3. Identify the optimal attribute split for the existing node using the dominant subset of m attributes chosen in step 2. Repeat steps 2 and 3 until the tree reaches its full depth (no pruning is applied).

This process is repeated independently for each tree in the forest, resulting in an ensemble model where each tree contributes to the final classification decision through a voting mechanism.

```
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(n_estimators=1000, random_state=42);
rfc.fit(dfx, dfy);

score = np.round(rfc.feature_importances_,6)
importances = pd.DataFrame({'feature':dfx.columns,'importance':score})
importances = importances.sort_values('importance',ascending=False).set_index('feature')
indices = np.argsort(score)[::-1]
plt.rcParams['figure.figsize'] = (20, 4)
importances.plot.bar();
```

Figure 11. Feature Importance by Random Forest Classifier

In the Feature Impotance by Random Forest Classifier, there are 2 parameters that we can change, the first is n_estimators and the next is random_state. The value of n_estimators has no definite limit, generally between 100 to 1000. The number of trees in the forest is determined by the n_estimators parameter. A random portion of the data is used to train each tree, and the combined forecasts of all the trees—typically by majority vote in classification tasks—make the final prediction. The Random_State parameter in a Random Forest Classifier is used to control the randomness of the bootstrapping of the samples used when building the trees and the randomness involved in selecting features for each split in a tree. In this research for Random-State in all experiments will use a single number 42, and for n_estimators, we will find out the best number by 2 methods. Figure 11 shows the process we do in Google Colab for the feature importance process of random forest classification.

2.3.1. GridSearchCV

This approach systematically explores all possible combinations of hyperparameter values in a given grid. The algorithm thoroughly evaluates all possible combinations of hyperparameters and selects the combination that produces the highest performance based on a given metric. In this research, we only change the n_estimators parameter with 5 nested choices from 50 to 1000. Then evaluate each value using cross-validation. Figure 12 shows the python used for this process.

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import accuracy_score

X = dframe.drop(['allophones'], axis=1)
y = dframe['allophones']
param_grid = {
    'n_estimators': [50, 100, 200, 500, 1000]
}
rf = RandomForestClassifier(random_state=42)

grid_search = GridSearchCV(estimator=rf, param_grid=param_grid, cv=5, scoring='accuracy')
grid_search.fit(X, y)

print(f"Best n_estimators: {grid_search.best_params_['n_estimators']}")
print(f"Best cross-validation accuracy: {grid_search.best_score_}")

best_rf = grid_search.best_estimator_
best_rf.fit(X, y)

predictions = best_rf.predict(X)
accuracy = accuracy_score(y, predictions)
print(f"Accuracy on training data: {accuracy}")
results = grid_search.cv_results_
df_results = pd.DataFrame({
    'params': results['params'],
    'mean_test_score': results['mean_test_score'],
    'std_test_score': results['std_test_score']
})
print()
print(df_results)
```

Figure 12. GridSearchCV Code

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

2.3.2. Manual Check

In this method, several choices of existing n_estimators values, one by one will be checked to evaluate the results of its accuracy testing with a random forest classifier and then ranked to find the best n_estimator value. The choices of n_estimators given are still the same as those of the previous method, namely 50,100,200,500,600 and 1000. Figure 13 shows the coding code in Google Colab for the manual check method.

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

# Load dataset
X = dframe.drop(['allophones'], axis=1)
y = dframe['allophones']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

n_estimators_values = [50, 100, 200, 500, 1000]

best_n_estimators = None
best_accuracy = 0

for n in n_estimators_values:
    rf = RandomForestClassifier(n_estimators=n, random_state=42)
    rf.fit(X_train, y_train)
    y_pred = rf.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)

    print(f"n_estimators = {n}: Accuracy = {accuracy:.4f}")

    # Check if this is the best accuracy we've seen so far
    if accuracy > best_accuracy:
      best_accuracy = accuracy
      best_n_estimators = n

print(f"\nBest n_estimators value: {best_n_estimators}")
print(f"Best accuracy: {best_accuracy:.4f}")
```

Figure 13. Manual Check Python Code

2.4. Deep Neural Network

Deep Neural Networks (DNN), employed in this study using frameworks like TensorFlow and Keras, are frequently utilized to analyze sequences of data in programming languages. These networks are widely known for their effectiveness in tasks such as image recognition and classification but can also be applied to other tasks such as natural language processing and signal processing.

Activation functions introduce non-linear properties into Deep Neural Networks (DNNs), enabling them to capture and learn complex relationships between input and output data. This research uses activation functions ReLU (Rectified Linear Unit) and sigmoid.

A DNN will train two datasets: normal data, and 'feature-important' data, and then we will analyze both results to find differences between both data regarding the accuracy and size of the resulting model file. Figure 14 shows the Python code for the DNN Model.

```
import tensorflow as tf
from tensorflow import keras
from keras.models import Sequential
from keras.layers import Dense, Activation, Flatten

model = Sequential()
model.add(Dense(len(x[0]), activation='relu', input_shape=x[0].shape))
model.add(Dense(len(x[0]), activation='relu'))
model.add(Dense(len(y[0,:]), activation='sigmoid',))
```

Figure 14. DNN Model

Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

3. RESULT

3.1. Finding RFC Parameter

The most important feature of a Random Forest classifier is its ability to assess the importance of each feature in making predictions. This is typically done through the calculation of feature importance scores, which indicate how valuable each feature is in predicting the target variable. For the Random Forest Classifier model, this research first finds the appropriate n_estimators parameter value that will produce results with a fairly high accuracy value. The following are the results of searching for n_estimates values in 2 methods.

The first method is the GirdSearchCV method, with this method from the experiment with 6 choices of n_estimators values (50, 100, 200, 500, 600, 1000) the highest accuracy results were obtained when using the n_estimators value of 600. Where with the n_estimators value, the Cross Validation Accuracy was obtained at 45.01% and the Training Accuracy value was 99.87%. Table 5 shows the detailed results of GridRecearchCV, where it can be seen that the mean_train_score value of all data has the same output of 99.87%, but for the mean_test_score there is a variation where n_estimators 600 has the highest accuracy, followed by n_estimators 500 and 1000 with the same value of 45.7%. The next sequence is n_estimators 200 then 50, and the lowest accuracy value is n_estimatord 100 with an accuracy value of 43%. mean_test_score is the average test score across all cross-validation folds for each hyperparameter combination. This is the primary metric used to determine the best hyperparameters, and mean_train_score is the average training score across all cross-validation folds for each hyperparameter combination.

Table 5. GridSearchCV Result

n_estimators	mean_test_score	mean_train_score	rank_test_score
50	0,432	0,998	5
100	0,43	0,998	6
200	0,442	0,998	4
500	0,457	0,998	2
600	0,46	0,998	1
1000	0,457	0,998	3

The second method is by manual check, with the same input parameter options, one by one the parameters are tested with the random forest classifier model and the accuracy results are evaluated. The results of this method obtained the best accuracy results on n_estimators 500 with an accuracy value reaching 0.4969. Table 6 shows the detailed results of this test where it can be seen that n_estimators 500 gets ranked 1, followed by n_estimators 600 with an accuracy of 0.472. The worst value is obtained by n_estimaros 50 with an accuracy of 0.4534.

Table 6. Manual Check Result

n_estimators	Accuracy	rank_test_score
50	0,4534	6
100	0,4596	5
200	0,4658	4
500	0,4969	1
600	0,472	2
1000	0,4658	4

Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

Based on the 2 methods above, n_estimators 500 and 600 are the most appropriate choice for the existing dataset and can produce maximum accuracy. This research will use n_estimators 500 to find the most important frequencies.

3.2. Importance Frequency

With the existing dataset, where the data consists of 200 data and frequencies from 10Hz to 2153Hz and uses the feature importance of the random forest classifier with a n_estimators value of 500, the most important frequency sequence is obtained as shown at table 7. Figure 15 shows a Bar Chart for the Feature Importance plot.

From the table above, the order of the most important frequencies is seen from the highest percentage score of influence of 0.92% with a frequency of 764 Hz. This means that the data at that frequency has the most influence on vocal phoneme changes. The frequency with the least influence is the frequency of 1830hz with a percentage of influence of 0.28%. From the table above, it can be seen that the 10 most influential frequencies are: 764 Hz, 1388 Hz, 785 Hz, 1227 Hz, 1345 Hz, 419 Hz, 645 Hz, 279 Hz, 775 Hz, and 441 Hz

Table 7. List of Importance Frequency

Number	Subject	1	2	3	4	5	6	7	8	9	10
1 ~ 10	Freq (Hz)	764	1388	785	1227	1345	419	645	279	775	441
	Score (%)	0,92	0,91	0,86	0,86	0,82	0,76	0,76	0,76	0,75	0,74
11 ~ 20	Freq (Hz)	1291	1410	753	678	1281	710	689	2153	656	818
	Score (%)	0,74	0,74	0,73	0,70	0,69	0,68	0,68	0,67	0,67	0,67
21 ~ 30	Freq (Hz)	430	635	2002	301	581	1313	2024	613	850	1335
	Score (%)	0,67	0,66	0,66	0,66	0,65	0,64	0,63	0,63	0,63	0,63
31 ~ 40	Freq (Hz)	452	409	570	269	882	1367	796	872	1453	258
	Score (%)	0,62	0,62	0,62	0,62	0,62	0,61	0,61	0,61	0,61	0,61
41 ~ 50	Freq (Hz)	2131	732	1356	1378	1248	1324	236	2034	742	861
	Score (%)	0,61	0,60	0,60	0,60	0,60	0,60	0,60	0,60	0,59	0,59
51 ~ 60	Freq (Hz)	2045	1302	2142	322	624	559	1238	506	2099	947
	Score (%)	0,58	0,58	0,58	0,58	0,57	0,57	0,57	0,56	0,56	0,56
61 ~ 70	Freq (Hz)	1431	1195	721	936	226	462	247	807	344	290
	Score (%)	0,56	0,56	0,56	0,56	0,56	0,56	0,56	0,56	0,55	0,55
71 ~ 80	Freq (Hz)	602	549	1399	312	366	1012	1259	1141	1894	376
	Score (%)	0,55	0,55	0,55	0,55	0,55	0,54	0,54	0,54	0,54	0,53
81 ~ 90	Freq (Hz)	1981	839	1022	2056	667	893	516	355	2088	829
	Score (%)	0,53	0,53	0,52	0,52	0,52	0,52	0,52	0,51	0,51	0,51
91 ~	Freq (Hz)	1087	1970	398	1959	1464	1270	904	1905	1518	592
100	Score (%)	0,51	0,51	0,51	0,51	0,50	0,50	0,50	0,50	0,50	0,49
101 ~	Freq (Hz)	527	1442	538	990	1216	915	2077	925	387	495
110	Score (%)	0,49	0,49	0,49	0,49	0,48	0,48	0,47	0,47	0,47	0,46
111 ~	Freq (Hz)	204	43	2121	96	215	118	1507	1001	333	1173
120	Score (%)	0,46	0,46	0,46	0,46	0,46	0,46	0,45	0,45	0,45	0,45
121 ~	Freq (Hz)	1421	1205	473	2067	968	107	2110	1991	2013	699
130	Score (%)	0,44	0,44	0,44	0,44	0,44	0,43	0,43	0,43	0,43	0,42
131 ~	Freq (Hz)	484	1884	1119	1948	1044	1873	1528	161	1108	75
140	Score (%)	0,42	0,42	0,42	0,42	0,42	0,41	0,41	0,41	0,41	0,41

Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

141 ~	Freq (Hz)	1841	32	193	1184	1152	1916	150	1937	1550	183
150	Score (%)	0,40	0,40	0,40	0,40	0,40	0,40	0,40	0,40	0,40	0,39
151 ~	Freq (Hz)	1851	1055	1076	1033	1098	129	172	1130	979	1787
160	Score (%)	0,39	0,39	0,39	0,39	0,39	0,39	0,39	0,39	0,39	0,38
161 ~	Freq (Hz)	86	139	1862	1593	1927	1485	1475	1701	958	1065
170	Score (%)	0,38	0,38	0,38	0,38	0,38	0,37	0,37	0,37	0,37	0,37
171 ~	Freq (Hz)	1496	10	1722	21	64	53	1808	1162	1819	1539
180	Score (%)	0,37	0,37	0,37	0,36	0,36	0,36	0,36	0,36	0,36	0,35
181 ~	Freq (Hz)	1571	1582	1636	1711	1604	1679	1776	1765	1614	1625
190	Score (%)	0,35	0,34	0,33	0,33	0,33	0,33	0,33	0,32	0,32	0,32
191 ~	Freq (Hz)	1733	1561	1647	1744	1658	1754	1668	1798	1690	1830
200	Score (%)	0,31	0,30	0,30	0,29	0,29	0,29	0,28	0,28	0,28	0,28

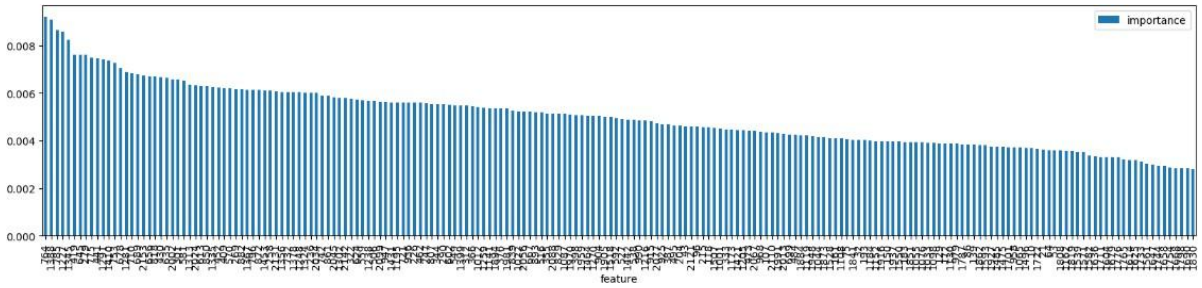

Figure 15. Feature Importance Frequency Chart

3.3. Deep Neural Network (DNN)

To confirm the correctness of the selection of the frequency sequence, a test was carried out with a Deep Neural Network to compare the accuracy testing of the resulting model.

3.3.1. Training Accuracy

In the first results of the first experiment, namely 20 sequence data, and 20 dominant frequency data, it can be seen that by using 20 feature importance data, the accuracy is much higher, almost 2 times. Likewise with the number of data 30, between the seq data and the dominant data, the difference is large and this difference is consistent until epoch 300 and there are no signs of a decrease in the difference. However, at the number of data 40, there is a phenomenon where at the beginning of training there is a difference but this difference is getting smaller, and finally, at epoch 270 both data already have the same accuracy. Furthermore, with the increasing number of 50 and 100 data, the training accuracy of both types of data has a relatively the same or close training accuracy value.

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

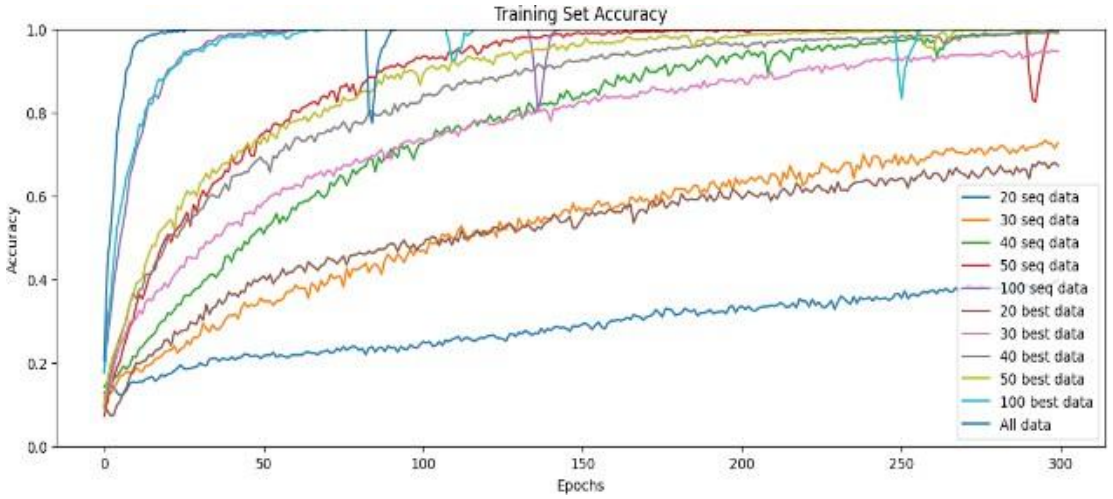

Figure 16. Training Accuracy Chart

3.3.2. Testing Accuracy

For accuracy testing, we conducted 8 trials to obtain more accurate data. From the results in Table 8, it can be seen that the results of each trial are not always the same. The input data for testing can be different because the data is randomly selected from the existing dataset.

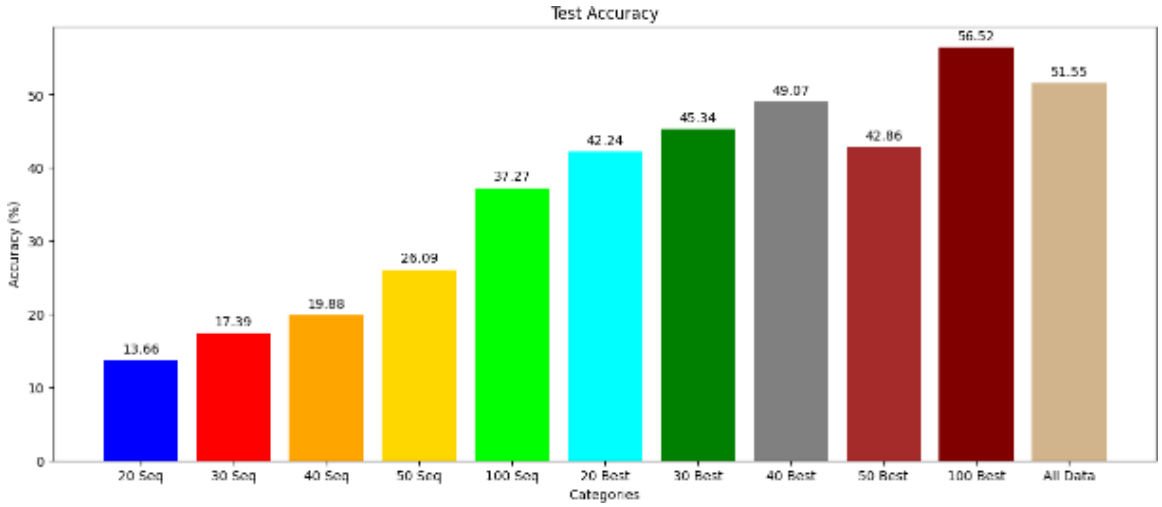

Figure 17. Testing Accuracy Chart

As seen in Figure 17, where there is a graph of accuracy test results, The smallest accuracy was obtained when using 20 sequence frequency datasets of 13.66%. The next results on the 30,40,50, and 100-frequency datasets are getting bigger, and it can be seen that the more datasets there are, the greater the accuracy obtained. For the accuracy results of 20 dominant frequency datasets, the results were 42.24%, this result is much greater than the same amount of data for 20 sequential frequency datasets, even though this result is greater than 100 sequential datasets which only have an accuracy of 37.27%. Different results occurred in the accuracy of 50 dominant frequency datasets which only showed results of 42.86% smaller than the results of 40 datasets which had an accuracy of 49.07%. The accuracy of 100 datasets reached 56.52%, the highest accuracy obtained, even greater than the accuracy of 200 datasets which was 51.55%.

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

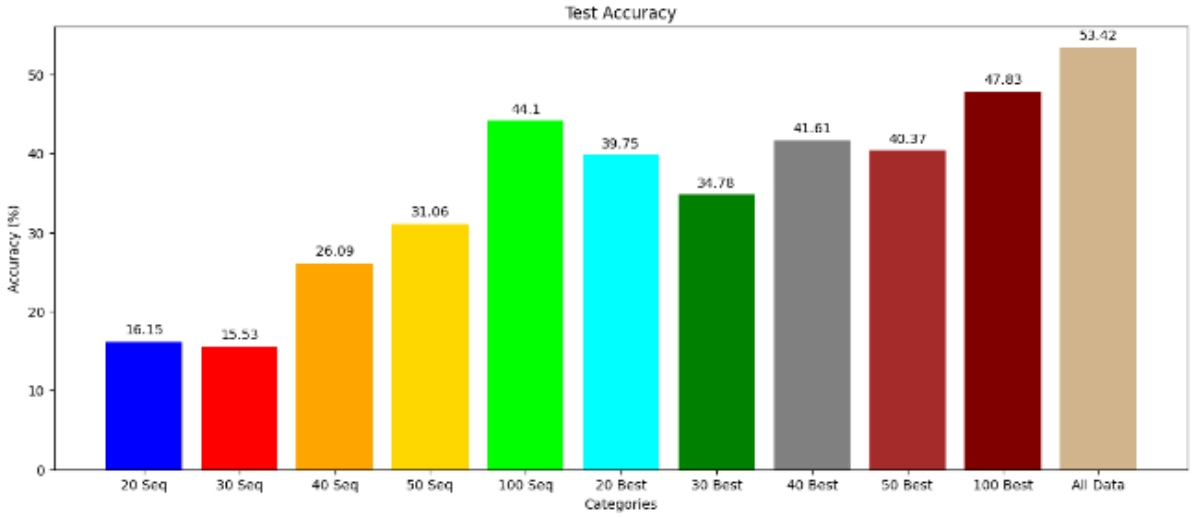

Figure 18. Testing Accuracy Chart 2

Figure 18 shows the results of the second test, it can be seen that the accuracy testing results of the 20 sequential frequency dataset increased to 16.15% but this result is still lower than the accuracy of the 20 dominant frequency dataset which reached 39.75%. The highest increase was in the accuracy of the 100 sequential frequency dataset reaching 44.1% but still below the results of the accuracy of the 100 dominant frequency dataset of 47.83%. Overall, the second test still proves that dominant frequencies produce higher accuracy testing.

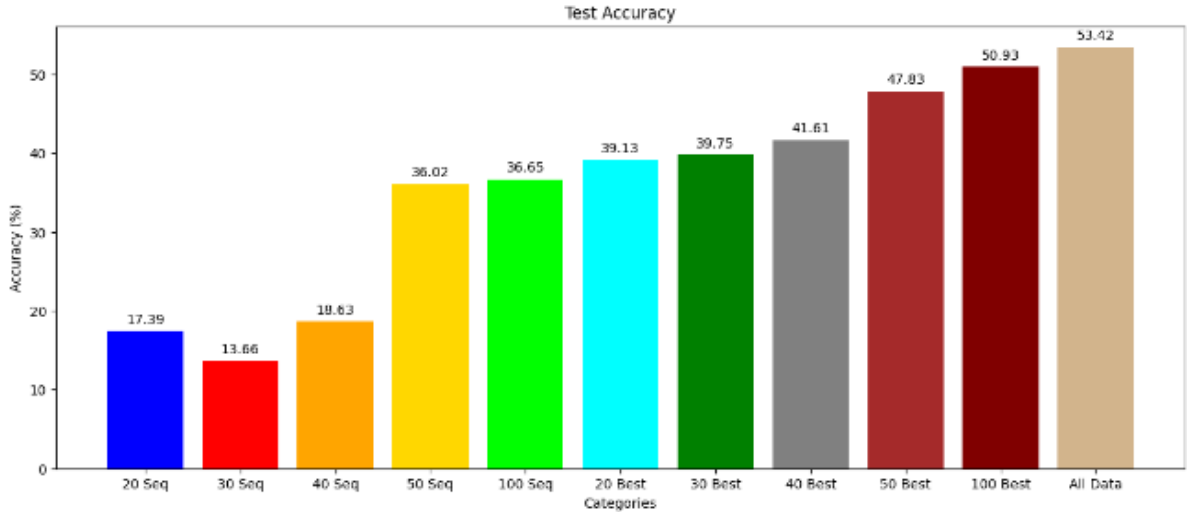

Figure 19. Testing Accuracy Chart 3

Figure 19 shows the third test; the chart showed no significant changes in the accuracy results of the 20 and 30-frequency sequence datasets compared to the previous test; a decrease in value occurred in the accuracy of the 40-frequency sequence dataset to 18.63% from the previous result which reached 16.09%. If, in the last test, the accuracy of the 100 frequency sequence dataset was higher than the accuracy of the 20 dominant frequency dataset, this test showed the opposite where the accuracy of the 20 dominant frequency dataset had a higher value. The test results on other datasets showed nominal or relatively the same values as the previous test.

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

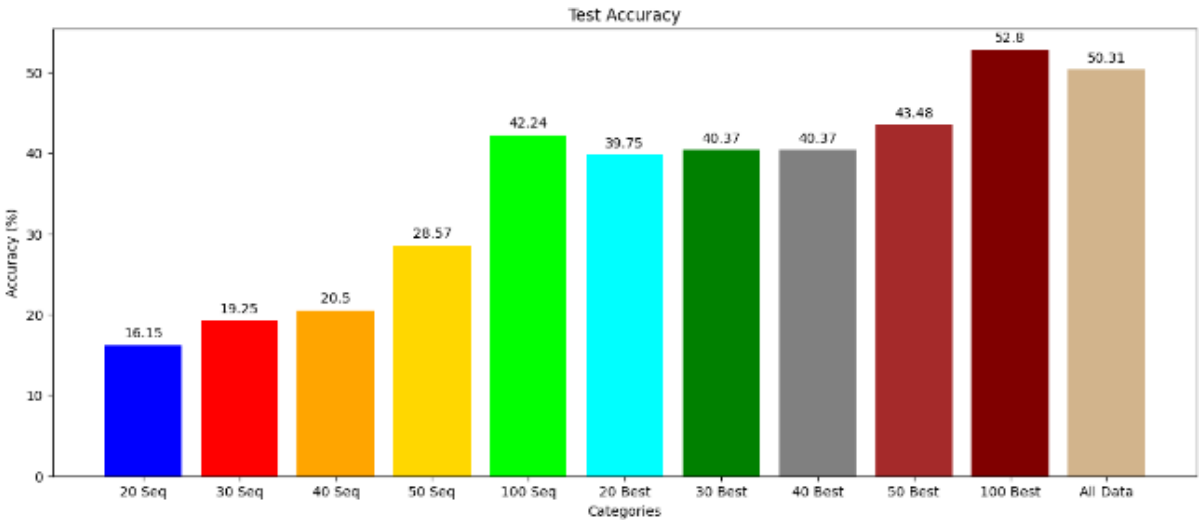

Figure 20. Testing Accuracy Chart

There are not many significant changes in this fourth test, as shown in Figure 20; there is only an increase in 2 datasets, namely the accuracy of the 100 frequency sequence dataset and the accuracy of the 100 dominant frequency dataset. The accuracy value of the 100-frequency sequence dataset reaches 42.24%, and the accuracy of the 100 dominant frequency dataset reaches 52.8%; this value is the highest accuracy value, exceeding the accuracy value of the dataset of all frequencies (200 frequencies).

From all the charts, the figure shows that the results of each trial are slightly different but not significant. Overall, it can still be seen that using the dominant frequency will get a higher accuracy test. Table 8 displays the details of the data obtained from the results of the accuracy test of 8 trials.

Table 8. Testing Accuracy Result

Dataset	Acc 1	Acc 2	Acc 3	Acc 4	Acc 5	Acc 6	Acc 7	Acc 8	Average
20 Seq	13,66	16,15	17,39	16,15	18,63	19,25	13,66	16,77	16,46
30 Seq	17,39	15,53	13,66	19,29	19,88	15,53	26,09	19,25	18,33
40 Seq	19,88	26,09	18,63	20,5	25,47	21,12	25,47	22,98	22,52
50 Seq	26,09	31,06	36,02	28,57	31,68	29,81	27,33	24,84	29,43
100 Seq	37,27	44,1	36,65	42,24	40,99	41,61	39,75	40,37	40,37
20 Best	42,24	39,75	39,13	39,75	39,13	39,13	37,27	48,45	40,61
30 Best	45,34	34,78	39,75	40,37	45,34	39,13	37,27	44,1	40,76
40 Best	49,07	41,61	41,61	40,37	40,99	37,27	45,96	43,48	42,55
50 Best	42,86	40,37	47,83	43,48	44,72	45,34	43,48	42,24	43,79
100 Best	56,52	47,83	50,93	52,8	50,31	55,28	49,07	54,66	52,18
All Data	51,55	53,42	53,42	50,31	50,93	49,07	54,66	52,8	52,02

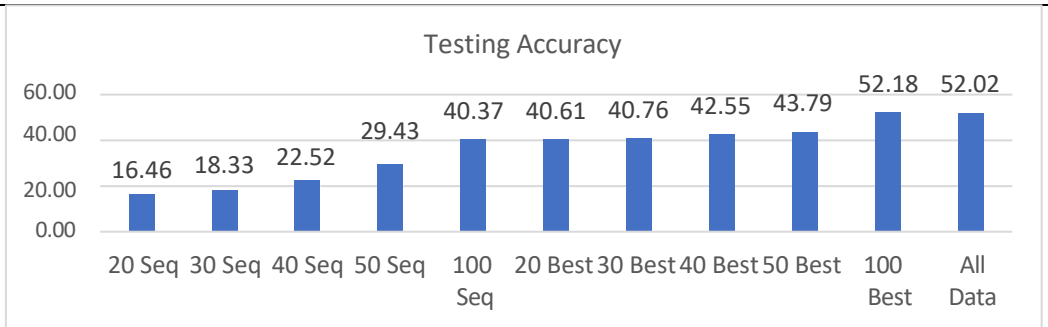

Figure 21. Summary Testing Accuracy Chart

Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

Figure 21 shows a bar graph of the average accuracy testing value of various types and numbers of datasets. The lowest accuracy testing is 16.46%, obtained from a dataset of 20 frequencies in order from the smallest 10Hz to 215Hz. Meanwhile, the accuracy testing value of the 20-frequency dataset selected based on the Feature Importance of the Random Forest Classifier can produce an accuracy of 40.61%. For the 100-frequency dataset, the accuracy testing is 40.37%, while the 100-dominant frequency dataset can produce an accuracy testing of 52.18%. The figure is slightly larger than the accuracy testing of the entire dataset (200 Frequencies), which is 52.02%.

3.3.3. Model File Size

The second objective of this study is to reduce the use of computer resources. The desired result is how much the file size decreases in relation to the reduction in the number of datasets when we only take certain frequencies. Table 9 shows the data on the size of the file model generated in the Deep Neural Network for each type of dataset. And also compared to the number of datasets themselves. The reference value of 100% is for the number of data 200 frequencies from 10Hz to 2153Hz and the file size for the model with that dataset. From the table, the size of the file model for 200 data is 997Kb. Then, if we look at the 100 Frequency dataset (50%), it will produce a file of 290Kb (28%), which means that the decrease in file size is greater than the decrease in the number of datasets. The decrease is clearly visible in Figure 19, which displays the decrease in the form of a bar graph.

Table 9. Model File Size

No	Model File Name	Qty Data	Percent Qty Data	Type Data	File Size (Kb)	Percent File Size
1	model_all_200_2048_500.h5	200	100,0%	Sequence	997	100,0%
2	model_f100a_200_2048_500.h5	100	50,0%	Sequence	280	28,1%
3	model_f50a_200_2048_500.h5	50	25,0%	Sequence	97	9,7%
4	model_f40a_200_2048_500.h5	40	20,0%	Sequence	75	7,5%
5	model_f30a_200_2048_500.h5	30	15,0%	Sequence	58	5,8%
6	model_f20a_200_2048_500.h5	20	10,0%	Sequence	45	4,5%
7	model_f100x_200_2048_500.h5	100	50,0%	Best Frequency	280	28,1%
8	model_f50x_200_2048_500.h5	50	25,0%	Best Frequency	97	9,7%
9	model_f40x_200_2048_500.h5	40	20,0%	Best Frequency	75	7,5%
10	model_f30x_200_2048_500.h5	30	15,0%	Best Frequency	58	5,8%
11	model_f20x_200_2048_500.h5	20	10,0%	Best Frequency	45	4,5%

Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

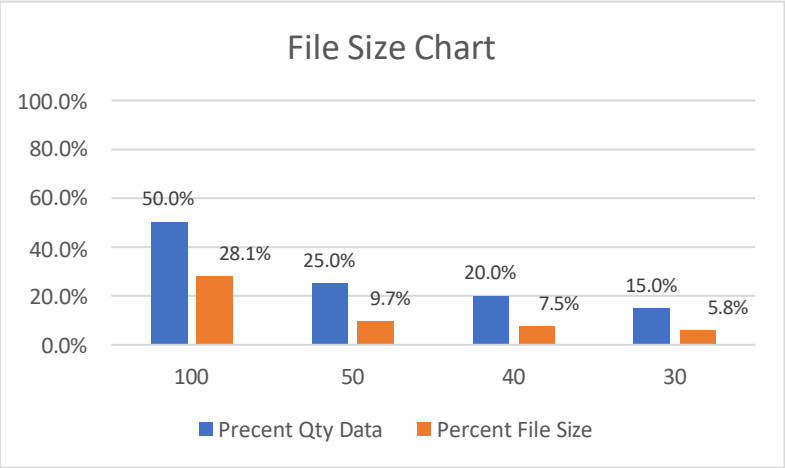

Figure 19. File Size Chart

Percent Accuracy Degradation between all data(200) compare to 100 Data:
(52.18 – 52.02) / 52.18 = 0.16 / 52.18 = 0.3%
Percent File Size Reduction between all data(200) compare to 100 Data :
(997 – 280) / 997 = 717 / 997 = 71,9 %

4. DISCUSSIONS

This research has highlighted the significant advantages of using dominant frequencies in developing and testing machine learning models. For initial training accuracy, there appears to be no significant difference for larger data sets. Although there is a similarity in training accuracy values for large datasets, the results for test accuracy show a significant difference, where it is seen that the use of dominant frequencies can get higher results. One of the main findings is that for the same number of datasets, the use of dominant frequencies by the feature importance method of the random forest classifier can produce a substantial increase in testing accuracy compared to using the full frequency spectrum. This suggests that not all frequencies contribute equally to the predictive power of the model and that removing less relevant frequencies can improve the model's ability to produce better testing accuracy.

Furthermore, this approach also results in the creation of more efficient model files in terms of size. The ability to produce smaller model files without sacrificing testing accuracy, and in some cases even increasing it, is an important advantage, especially in environments where computing resources or storage capacity are limited. By reducing model complexity while maintaining its effectiveness, this method achieves a balance between performance and resource utilization. This is evident in this research when dominant frequencies are used, even with only half of the total data available, the test accuracy can slightly exceed that obtained when using all frequencies ranging from 10 to 2153 Hz. This finding underscores the power of frequency selection, as it allows the model to focus on the most informative aspects of the data, leading to better performance even with reduced input data. Overall, these findings support the idea that frequency selection in Javanese language identification can not only improve the efficiency and accuracy of machine learning models but also offer practical benefits in terms of model size and resource management. This approach can be particularly useful in scenarios where computational overhead needs to be minimized, providing a path to more efficient and effective model development.

Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

5. CONCLUSION

This research has highlighted the significant advantages of using selected frequencies in developing and testing machine learning models. For initial training accuracy, there appears to be no significant difference for larger data sets. Although there is a similarity in training accuracy values for large datasets, the results for test accuracy show a significant difference, where it is seen that the use of selected frequencies can get higher results. One of the main findings is that for the same number of datasets, the use of frequencies selected by the feature importance method of the random forest classifier can produce a substantial increase in testing accuracy compared to using the full frequency spectrum. This suggests that not all frequencies contribute equally to the predictive power of the model and that removing less relevant frequencies can improve the model's ability to produce better testing accuracy.

Furthermore, this approach also results in the creation of more efficient model files in terms of size. The ability to produce smaller model files without sacrificing testing accuracy, and in some cases even increasing it, is an important advantage, especially in environments where computing resources or storage capacity are limited. By reducing model complexity while maintaining its effectiveness, this method achieves a balance between performance and resource utilization. This is evident in this research when selected frequencies are used, even with only half of the total data available, the test accuracy can slightly exceed that obtained when using all frequencies ranging from 10 to 2153 Hz. This finding underscores the power of frequency selection, as it allows the model to focus on the most informative aspects of the data, leading to better performance even with reduced input data.

In this research, we searched for the most dominant frequencies in several Javanese speech samples, with a focus on finding the most effective way to recognize Javanese and reduce the magnitude of the Javanese language recognition process. This study successfully identified dominant frequencies in Javanese vocal phonemes using a combination of Fast Fourier Transform (FFT) and Random Forest classification. The effectiveness of using dominant frequencies is seen when measuring Accuracy Testing using Deep Neural Network. when we use 100 (50%) of selected frequency data, we can get the same accuracy or even more than using 200 frequency data. Effectiveness is also seen in the size of the model file, the file size can be reduced by 71.9%. This approach can be particularly useful in scenarios where computational overhead needs to be minimized, providing a path to more efficient and effective model development.

REFERENCES

[1] Elavarasi, S., and G. Suseendran. 2019. "MFCC Using Speech Recognition in Computer Applications for Deaf." International Journal of Recent Technology and Engineering 8(2 Special Issue 11): 217–24. doi:10.35940/ijrte.B1036.0982S1119.

[2] Hannun, Awni. 2021. "The History of Speech Recognition to the Year 2030." http://arxiv.org/abs/2108.00084.

[3] Laksono, Teguh Puji, Ahmad Fathan Hidayatullah, and Chanifah Indah Ratnasari. 2018. "Speech to Text of Patient Complaints for Bahasa Indonesia." Proceedings of the 2018 International Conference on Asian Language Processing, IALP 2018: 79–84. doi:10.1109/IALP.2018.8629161.

[4] Marsono, Marsono. 2012. "Fonem Vokal Bahasa Jawa Kuna Dan Alofon-Alofonnya." Humaniora 11(1): 56–62. https://journal.ugm.ac.id/jurnal-humaniora/article/view/625.

[5] Reddy, V. Madhusudhana, T. Vaishnavi, and K. Pavan Kumar. 2023. "Speech-to-Text and Text-to-Speech Recognition Using Deep Learning." Proceedings of the 2nd International Conference on Edge Computing and Applications, ICECAA 2023 (Icecaa): 657–66. doi:10.1109/ICECAA58104.2023.10212222.

[6] Sen, Tjong Wan. 2019. "Voice Activity Detector for Device with Small Processor and Memory." ICSECC 2019 - International Conference on Sustainable Engineering and Creative Computing: New Idea, New Innovation, Proceedings: 212–17. doi:10.1109/ICSECC.2019.8907081.

Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708

[7] Theelen, Mathilde. 2017. "Fundamental Frequency Differences Including Language Effects." Junctions: Graduate Journal of the Humanities 2(1): 9. doi:10.33391/jgjh.25.

[8] Winursito, Anggun, Risanuri Hidayat, Agus Bejo, and Muhammad Nur Yasir Utomo. 2018. "Feature Data Reduction of MFCC Using PCA and SVD in Speech Recognition System." 2018 International Conference on Smart Computing and Electronic Enterprise, ICSCEE 2018: 1–6. doi:10.1109/ICSCEE.2018.8538414.

[9] Balabanova, Ivelina, Georgi Georgiev, Boyan Karapenev, and Valentina Rankovska. 2022. "Voice Analysis for Personal Identification Using FFT, Machine Learning and AI Techniques." In AIP Conference Proceedings, American Institute of Physics Inc. doi:10.1063/5.0099672.

[10] Tobing, Fery. "PENERAPAN ALGORITMA FAST FOURIER TRANSFORM (FFT) PADA PROTOTYPE DOOR LOCK SYSTEM BERBASIS VOICE RECOGNITION The Application of the Fast Fourier Transform (FFT) Algorithm to the Voice Recognition Based Door Lock Prototype System." doi:10.30813/j-alu.v2i2.2090.

[11] Nassif, Ali Bou, Ismail Shahin, Imtinan Attili, Mohammad Azzeh, and Khaled Shaalan. 2019. "Speech Recognition Using Deep Neural Networks: A Systematic Review." IEEE Access 7(February): 19143–65. doi:10.1109/ACCESS.2019.2896880.

[12] Agustina, Triya, Masrizal Masrizal, and Irmayanti Irmayanti. 2024. "Performance Analysis of Random Forest Algorithm for Network Anomaly Detection Using Feature Selection." sinkron 8(2). doi:10.33395/sinkron.v8i2.13625.

[13] Wibawa, Aji P., Andrew Nafalski, and Wayan F. Mahmudy. 2014. "Augmented Javanese Speech Levels Machine Translation." ECTI Transactions on Computer and Information Technology (ECTI-CIT) 8(1): 45–55. doi:10.37936/ecti-cit.201481.54387

[14] Nugroho, Kristiawan. 2020. "Javanese Gender Speech Recognition Based on Machine Learning Using Random Forest and Neural Network." Sisforma 6(2): 50–54. doi:10.24167/sisforma.v6i2.2402.

[15] Kim, Callie Y, Diana Chou, and Geng Liu. "Voice Recognition on Simple Microcontrollers." : 1–6. https://cmusphinx.github.io.

Jurnal Teknik Informatika (JUTIF)
P-ISSN: 2723-3863
E-ISSN: 2723-3871
Vol. 6, No.2, April 2025, Page. 871-890
https://jutif.if.unsoed.ac.id
DOI: https://doi.org/10.52436/1.jutif.2025.6.2.2708