



Design and Implementation of REST API for Predicting the Recitation of the Qur'an using Machine Learning

Krisna Santosa^{1*}, Rifdah Hansya Rofifah², Alfia April Riani³, Indira Syawanodya⁴, Iqbal Tawaka⁵
Software Engineering, Universitas Pendidikan Indonesia, Indonesia

Correspondence: E-mail: krisnasantosa@upi.edu

ABSTRACT

The Qur'an is the Muslim holy book, consisting of 30 juz and 114 surahs of varying length and number of verses. Reading the Qur'an involves special techniques to understand and read with similarity or consistency to the verse being read. In the digital era, technology enables the development of applications that support the learning and analysis of Qur'anic recitation. This research aims to design and implement a REST API to predict Qur'an recitation using a machine learning (ML) model. This API accepts voice recordings from users and provides output in the form of an assessment of the similarity of their recitation to the desired verse. Using FastAPI and pre-trained models such as Wav2Vec2, the system can translate audio into text with fairly good accuracy. Experimental results show a word error rate (WER) of 30%, which indicates the need for further improvement but is sufficient in the experimental context. The technology is useful as a self-learning tool for the Qur'an, but it does not replace the role of the teacher. Future research should focus on improving model accuracy and integrating more user-friendly features.

ARTICLE INFO

Article History:

Submitted/Received 02 Feb 2024

First Revised 29 Feb 2024

Accepted 18 Apr 2024

First Available online 24 Apr 2024

Publication Date 01 June 2024

Keyword:

Qur'an,

REST API,

Machine Learning,

Speech Recognition.

1. INTRODUCTION

The Qur'an is the Muslim holy book, consisting of 30 juz and 114 surahs. Each surah has various lengths and different numbers of verses, starting with surah Al-Baqarah, which has as many as 286 verses, and An-Nas, which only has 5 verses. Reading the Qur'an is not just reading; it involves special rules that need to be considered so that Muslims can read the Qur'an according to the reading taught by the Prophet Muhammad and his companions as the Qur'an was revealed (Syaifullah et al., 2021).

In the current digital era, the use of technology has enabled the development of various applications that support the learning and use of the Qur'an. One innovative approach is the application of machine learning to predict and analyze the recitation of the Qur'an. In this context, the use of the REST API (Representational State Transfer Application Programming Interface) becomes very relevant because it allows system integration with a wide range of applications and platforms. In addition, the implementation of REST facilitates the development of software applications outside the system because it provides an interface similar to REST and API documentation, which allows the use of different programming languages or platforms (Ahmad et al., 2021).

This research aims to design and implement a REST API to predict the recitation of the Qur'an with high accuracy, which matches the recitation performed by the user. Furthermore, the REST API will be implemented as an interface to integrate the Machine Learning model with other applications or platforms. This API will receive input in the form of voice recordings from users and provide output in the form of scores or points that reflect how well the user is able to recite or read the Qur'an.

2. METHODS

This research uses the Waterfall method to develop and design a Qur'an recitation prediction system based on an REST API and machine learning models. This method is sequential and step-by-step, where each step must be completed thoroughly before moving on to the next step. Figure 1 illustrates the Waterfall stages applied in this research.

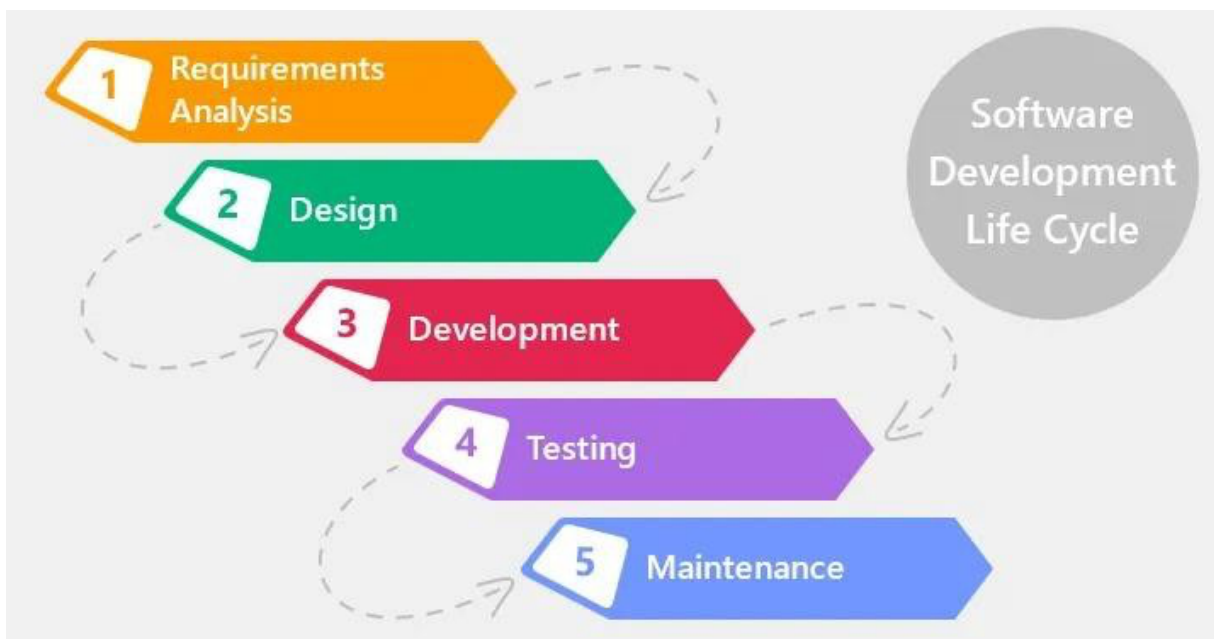


Figure 1. Waterfall Method

1. Requirements Analysis

The initial step in this process is to collect all relevant information related to the system requirements, including technical and functional specifications. This information is then analyzed to determine the software solution that fits those needs.

2. Design

Based on the requirements analysis in the initial stage, the design stage focuses on designing a structured system architecture. This design includes how the API will receive input, process data, and provide output. The system design also includes the selection of technologies such as the FastAPI framework for REST API creation and the Wav2Vec2 machine learning model for speech recognition.

3. Development

Based on the system architecture design that has been compiled, the next step is to translate it into concrete programming code. This implementation is focused on developing the backend of the application using the FastAPI framework. The Wav2Vec2 machine learning model, adapted from Nuwaisir's Kaggle repository/Quran_speech_recognizer is used in this process.

4. Testing

After the code implementation is complete, the next step is system testing. At this stage, the author uses the Word Error Rate (WER) testing method to measure the accuracy of the model in recognizing and translating voice recordings into text. WER is calculated as the percentage of words that are wrong out of all spoken words.

5. Maintenance

The last stage is the maintenance of the system that has been developed to find errors or bugs in the application and ensure that the entire system runs optimally. Maintenance is carried out regularly to ensure that the system continues to function properly and is responsive to user needs.

2.1. Method of Collecting Data

Literature Review

A literature study is a data collection method that seeks data and information through research journals, e-books, the internet, and books that support the thesis writing process. The author uses the references above to produce scientific work that has credibility.

1. Speech Recognition

Speech recognition is a process of voice identification based on words spoken and captured by an audio device (voice input device). Speech recognition is a form of biometric recognition where a computer system can identify what is spoken by someone through a microphone by analyzing the intonation of the voice, which is then converted into a digital print (Rumopa, 2015).

Speech recognition plays an important role as an early stage in the data collection process. When a user reads a verse of the Qur'an and his voice is recorded, speech recognition converts the voice recording into text that can be further processed by the system. After the speech recognition text is obtained, the REST API then sends the text using machine learning. The machine learning model will analyze the text and look for patterns in the pronunciation of the letters in the Qur'an. The results of this analysis are then returned through the REST API as output to the user.

2. Machine Learning

Machine learning is the study of algorithms that enable systems to learn to perform certain tasks automatically, similar to the way they are performed by humans. In this context, “learning” refers to the ability of a system to perform certain previously learned activities or to discover significant new patterns from observed data. This subfield of artificial intelligence has far-reaching impacts in various fields, including statistics, mathematics, and various other theoretical computer science disciplines (Fathurohman, 2021).

Machine learning can be used to develop algorithms that are able to recognize voices and analyze the recitation of the Qur'an. By integrating machine-learning technology, the system can become more interactive and responsive to user needs.

3. API

API, or Application Programming Interface, is an interface used to access applications or services from a program. In other words, the role of the API is to act as an intermediary between different applications, either on the same platform or across platforms. APIs facilitate developers to use existing functions from other applications, so there is no need to create them from scratch. In the context of the web, APIs are used to access functions via Hyper Text Transfer Protocol (HTTP) and get responses in the form of Extensible Markup Language (XML) or JavaScript Object Notation (JSON). The use of APIs aims to exchange data between different applications. In addition, APIs are also used to speed up application development by providing separate functions so that developers do not need to develop similar features independently (Kurniawan & Rozi, 2020).

4. REST

REST, or Representational State Transfer, is an API architecture that transmits data through a standardized interface such as HTTP. The REST API allows clients to request data or perform actions on the server through HTTP requests that have methods such as GET, POST, PUT, and DELETE. The client can send a request to the server through the HTTP protocol, and then the server provides a favorable response to the client. In the REST architecture, the REST server provides resources, and the REST client accesses and displays the resources for further use. Each resource is identified by a URI (Universal Resource Identifier) or global ID. The resource is represented in text format, either JSON or XML. Generally, the formats used are JSON or XML.

The REST API facilitates seamless integration between the customized machine learning model and the Qur'an recitation prediction application. This model is implemented using the Python programming language and the FastAPI API framework.

3. RESULT AND DISCUSSION

3.1. Implementation of the Qur'an Recitation Prediction API

The implementation of the API to predict the recitation of the Quran uses a Machine Learning (ML) model that has been adapted from previous models. This model is taken from the Kaggle website with the Nuwaisir/Quran_speech_recognizer repository. This API was built using the Python programming language and the FastAPI library as a support for making REST- API. FastAPI is an open-source library developed to be a solution for making APIs quickly and efficiently because the documentation will be automatically created by the FastAPI library, which is ready for deployment (Hasan Abdulqader et al., 2022). Machine Learning technology

is very suitable to be combined with FastAPI because both use the same programming language, making the integration process easier. In addition, the performance generated by FastAPI is very good for handling many requests at once. In its technical implementation, this API will receive input in the form of a voice file in WAV format, then the model processes the voice file, then predicts and translates it into text.

3.2. Machine Learning Model Architecture

In developing a Machine Learning model for the prediction of al-Qur'an recitation, an architecture that combines Convolutional Neural Networks (CNN) and Transformers is used. This model also uses a Self-Supervised Pretraining approach to improve its ability to recognize sound patterns. The following is a more detailed explanation of the architecture components used:

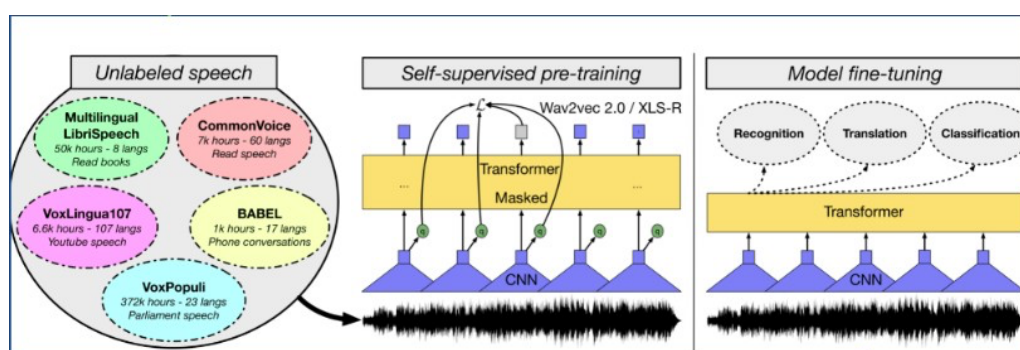


Figure 2. Machine Learning Architecture

Unlabeled Speech

In the first stage, the model was trained using unlabeled speech data from various sources:

- Multilingual LibriSpeech: A dataset containing 50,000 hours of recordings in 8 languages derived from books read.
- CommonVoice: This dataset contains 7,000 hours of recordings in 60 languages generated from recited speeches.
- VoxLingua107: This dataset contains 6.6 thousand hours of recordings in 107 languages taken from speeches from the YouTube platform.
- BABEL: This dataset contains 1,000 hours of recordings in 17 languages consisting of phone conversations.
- VoxPopuli: This dataset contains 372,000 hours of recordings in 23 languages from speeches in parliament.
- Self-Supervised Pre-Training

In the self-supervised pretraining stage, the model uses self-supervised learning techniques to learn from unlabeled data:

- CNN (Convolutional Neural Networks): Used to perform basic feature extraction from audio data. CNN processes the raw audio signal to produce a higher feature representation (Hasan Abdulqader et al., 2022).
- Transformer: Used to capture context and long-term relationships in audio data. In this architecture, transformers are trained by predicting missing or masked parts of the audio data, which allows the model to understand the structure and patterns in the data (Zayat et al., 2023).

- Masked Modeling: The model is trained to predict the masked portion of the audio, which forces the model to understand the global context of the input data (Xin et al., 2023).
- The resulting model from this stage is Wav2Vec 2.0, or XLS-R, which has been trained on various unlabeled data sources to understand the basic representation of audiodata.
- Fine-Tuning Model

At this stage, the pretrained model is fine-tuned for specific tasks such as speech recognition, translation, and classification. Fine-tuning is done with labeled data to optimize the model's performance on specific tasks.

 1. *Recognition*: The model is fine-tuned for speech recognition tasks, such as recognizing Qur'an recitations from audio input.
 2. *Translation*: The model is fine-tuned for audio-to-text translation tasks in other languages.
 3. *Classification*: The model is fine-tuned for audio classification tasks, such as identifying sound types or sound sources.

During fine-tuning, the model uses a combination of CNNs and transformers that have been trained in the pretraining stage. This fine-tuning is done with labeled datasets to improve the accuracy and performance of the model on specific tasks of interest (Meghanani & Hain, 2024).

The resulting model can be seen in the following image:

| Model Scores | Training Accuracy | | Validation Accuracy | | Validation Loss | Test Accuracy | |
|---------------------|-------------------|---------------|---------------------|---------------|------------------------|------------------|---------------|
| | WER | Edit Distance | WER | Edit Distance | | WER | Edit Distance |
| Our Model | 0.3258 | 46 | 0.30638 | 35 | 0.4511878 192424774 | 0.379194 6309 | 55 |
| Common Voice Arabic | 0.9744 4 | 91 | 0.99574 | 70 | 4.1038255 69 | 0.966442 953 | 87 |

Figure 3. Model Results

From the image, the results of the model compared with other models show that the model developed is better in various aspects. However, in the context of using AI-Quran reading predictions, with a Word Error Rate (WER) of 30% it cannot be said to be good in use, because usually AI-based translation and navigation applications require a WER below 10% (Helmke et al., 2021). This does not mean that the model cannot be used, but its use must be adapted to the application context.

3.3. Architecture API

The API was developed using the FastAPI framework and consists of functions to build the main components of the API to process files and return them as text.

API Key Components:

Endpoint for Audio File Upload:

Use UploadFile and File from FastAPI to receive audio files in WAV format. The received files are stored in a predefined directory.

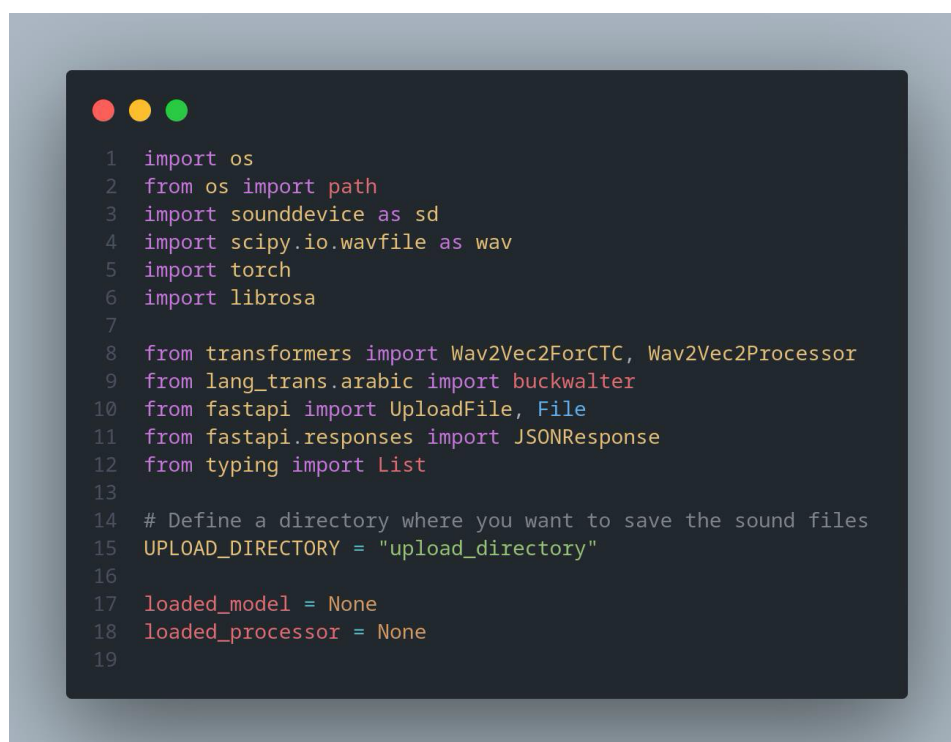
Model Invocation:

The Wav2Vec2 model is loaded when the API is first run and used for prediction. The load_model function is responsible for loading the model and processor.

Prediction and Returns:

The predict_sound_file function performs pre-processing on the audio file, runs the model, and returns the prediction result.

The prediction result is returned in JSON format via JSONResponse. Code Implementation:



```
1 import os
2 from os import path
3 import sounddevice as sd
4 import scipy.io.wavfile as wav
5 import torch
6 import librosa
7
8 from transformers import Wav2Vec2ForCTC, Wav2Vec2Processor
9 from lang_trans.arabic import buckwalter
10 from fastapi import UploadFile, File
11 from fastapi.responses import JSONResponse
12 from typing import List
13
14 # Define a directory where you want to save the sound files
15 UPLOAD_DIRECTORY = "upload_directory"
16
17 loaded_model = None
18 loaded_processor = None
19
```

Figure 4. Import the required libraries

```

1  async def quran_recognizer(sounds: List[UploadFile] = File(...)):
2      """Quran Recognizer
3
4          This API is used to recognize quran sound.
5
6          the sound file must be in .wav format.
7      """
8      if not os.path.exists(UPLOAD_DIRECTORY):
9          os.makedirs(UPLOAD_DIRECTORY)
10
11     if loaded_model is None:
12         load_model() # Load the model if it's not already loaded
13
14     for sound in sounds:
15         # Save the uploaded sound file to a local directory
16         sound_path = os.path.join(UPLOAD_DIRECTORY, sound.filename)
17         with open(sound_path, "wb") as file:
18             file.write(sound.file.read())
19
20     # Predict the transcribed text
21     transcribed_text = predict_sound_file(sound_path, loaded_model, loaded_processor)
22     response_data = {"message": transcribed_text, "success": True, "code": 200, "meta": {}, "data": ""}
23
24
25     return JsonResponse(content=response_data)

```

Figure 6. Voice recognition and text sending functions

3.4. Experiment Results

The experimental results show that the fine-tuned Wav2Vec2 model for reciting al-Qur'an successfully translates sound into text quite well. Below is the documentation of the experiment.

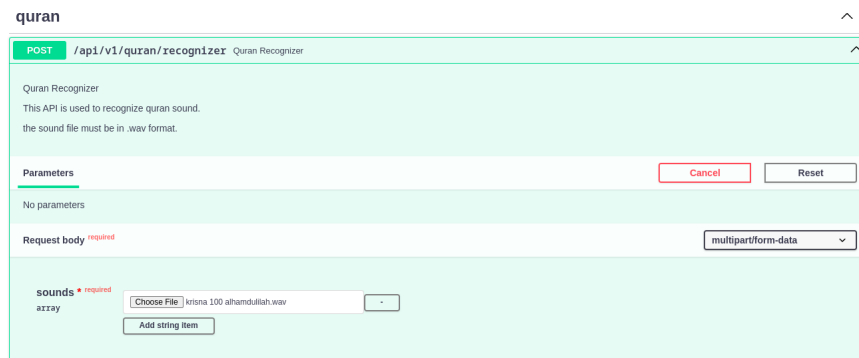


Figure 7. Voice input

Figure 6 presents a visualization of how this API works to accept input in the form of files with WAV format. The API view shown in the figure is the Swagger UI view commonly used in API development standards.

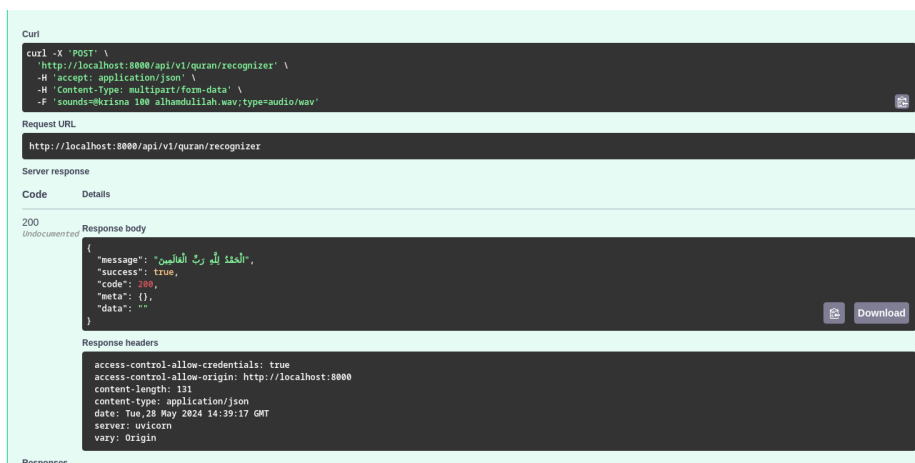


Figure 8. Execution result

In Figure 7, there is a response result from the server that returns text in the form of Qur'anic recitations that have been predicted using the Machine Learning model. This text can later be processed into various other implementations, such as finding the similarity of the Qur'an text with the text translated by the Machine Learning model, the more similar the text, the higher the level of correctness in reading the Koran.

Error Analysis:

It should be underlined that learning the Koran requires assistance from experts such as ustadz, this is because the Koran is very sensitive to pronunciation. The slightest difference in pronunciation can change the original meaning. Therefore, there are limits that should be obeyed that this technology is present not to replace a teacher in learning the Koran, but as a tool in the learning process, so that people can learn the Koran anywhere with the help of technology. Not only that, the Machine Learning model currently implemented is not an up-to-date model, requiring in-depth customization to obtain a high level of model accuracy.

4. CONCLUSION

The conclusion of this research is that the Qur'an, as the holy book of Muslims consisting of 30 juz and 114 surahs, has special recitation rules that aim to make the reading match the way taught by the Prophet Muhammad and his companions. In the digital era, the use of Machine Learning technology and REST APIs opens up new opportunities for learning and analyzing the recitation of the Qur'an. This research successfully designed and implemented a REST API to predict the recitation of the Qur'an using a Machine Learning model. This REST API is able to receive input in the form of voice recordings and provide output in the form of an assessment of the suitability of the user's reading according to the applicable rules.

Experimental results show that the fine-tuned Wav2Vec2 model can translate voice to text quite well, although the resulting Word Error Rate (WER) is still around 30%, which suggests that there is room for further improvement. This technology is useful as a learning aid that allows users to learn Qur'anic recitation independently. However, this research also recognizes the limitations, especially in terms of model accuracy, which still needs to be improved in order to be used more effectively in real-world applications.

In addition, this study emphasizes that this technology is not intended to replace the role of teachers or experts in learning the Qur'an but as a supporting tool that facilitates learning anywhere. Future research could focus on improving the accuracy of the model through further fine-tuning and more extensive data collection. In addition, integration with more user-friendly applications and the development of additional features for recitation analysis and training can be on the next research agenda.

5. REFERENCES

- Ahmad, I., Suwarni, E., Borman, R. I., Asmawati, Rossi, F., & Jusman, Y. (2021). Implementation of RESTful API Web Services Architecture in Takeaway Application Development. *2021 1st International Conference on Electronic and Electrical Engineering and Intelligent System, ICE3IS2021*, 132–137.
- Fathurohman, A. (2021). MACHINE LEARNING UNTUK PENDIDIKAN: MENGAPA DAN BAGAIMANA. *Jurnal Informatika Dan Teknologi Komputer (JITEK)*, 1(3), 57–62.

- Hasan Abdulqader, A., AbdulRahman Al-Haddad, S., Abdo, S., Abdulghani, A., & Natarajan, S. (2022). Hybrid Feature Extraction MFCC and Feature Selection CNN for Speaker Identification Using CNN: A Comparative Study. *2022 2nd International Conference on Emerging Smart Technologies and Applications (ESmarTA)*, 1–6.
- Helmke, H., Shetty, S., Kleinert, M., Ohneiser, O., Prasad, A., Motlice, P., Cerna, A., & Windisch, C. (2021). *How to Measure Speech Recognition Performance in the Air Traffic Control Domain? The Word Error Rate is only half of the truth.*
- Kurniawan, I., & Rozi, F. (2020). *REST API Menggunakan NodeJS pada Aplikasi Transaksi Jasa Elektronik Berbasis Android* (Vol. 1, Issue 4).
- Meghanani, A., & Hain, T. (2024). *SCORE: Self-supervised Correspondence Fine-tuning for Improved Content Representations.*
- Rumopa, V. W. (2015). *KONTROL PENERANGAN RUANGAN MENGGUNAKAN SENSOR SUARA (SPEECH RECOGNITION) BERBASIS ANDROID.*
- Syaifullah, A., Rahmah, F. M., Salamah, F., & Srisantyorini, T. (2021). PENERAPAN ILMU TAJWID DALAM PEMBELAJARAN AL-QURAN UNTUK MENGEMBANGKAN BACAAN AL-QURAN. *Prosiding Seminar Nasional Pengabdian Masyarakat LPPM UMJ*, 1(1).
- Xin, Y., Peng, X., & Lu, Y. (2023). *Masked Audio Modeling with CLAP and Multi-Objective Learning.* 2763–2767.
- Zayat, A., A. Hasabelnaby, M., Obeed, M., & Chaaban, A. (2023). Transformer Masked Autoencoders for Next-Generation Wireless Communications: Architecture and Opportunities. *IEEE Communications Magazine*, PP, 1–8.