LAPORAN AKHIR

PENELITIAN INTERNAL DOSEN

PROGRAM STUDI TEKNIK INFORMATIKA

Improved Feature Extraction for Sound Recognition using Combined Constant-Q Transform (CQT) and Mel Spectrogram for CNN Input

TIM PENGUSUL

Silvester Dian Handy Permana

T.K. Abdul Rahman

UNIVERSITAS TRILOGI

November 2023

HALAMAN PENGESAHAN

Judul Penelitian **:** Improved Feature Extraction for Sound Recognition using Combined Constant-Q Transform (CQT) and Mel Spectrogram for CNN Input

Ketua Peneliti :

a. Nama : Silvester Dian Handy Permana, S.T., M.T.I.
b. NIDN : 0326119001
c. Jabatan Fungsional : Lektor Kepala
d. Program Studi : Teknik Informatika

Lama Penelitian : 2 Bulan (1 September - 25 Oktober 2023)

Hasil Luaran : 2023 International Conference on Modeling & E-Information Research, Artificial Learning and Digital Applications (ICMERALDA)

Jakarta, 1 November 2023

Menyetujui,

Ir. Yaddarabullah, M.Kom., ASEAN Eng.

(Kaprodi Teknik Informatika)

DAFTAR ISI

RINGKASAN

This paper delves into an innovative paradigm for enhancing feature extraction in the realm of speech recognition. It integrates the Constant-Q Transform (CQT) and Mel Spectrogram techniques to elevate the performance of Convolutional Neural Networks (CNN) in the domain of speech recognition. The study introduces a novel augmentation to the traditional Mel-frequency cepstral coefficients (MFCC) extraction methodology. This enhancement involves utilizing the frequency representations obtained from CQT and Mel Spectrogram as inputs for the CNN model. Extensive experimentation and rigorous evaluation validate the efficacy of the proposed approach. It is evident from the results that this approach yields more intricate and meaningful spectral features when compared to the conventional MFCC technique. These findings underscore the significant potential inherent in amalgamating CQT and Mel Spectrogram techniques, showcasing their ability to significantly augment the accuracy and overall performance of CNN models in the context of speech recognition tasks. This breakthrough underscores the continuous evolution of speech recognition methodologies and presents a promising avenue for further research and development in the field.

Kata Kunci : Sound recognition, Improved Feature Extraction, Constant-Q Transform (CQT), Mel Spectrogram, Convolutional Neural Network (CNN)

BAB I. PENDAHULUAN

1.1 **Latar Belakang**

Sound recognition, a fundamental task in audio processing and machine learning, plays a crucial role in various real-world applications, including speech recognition, music classification, environmental sound monitoring, and human-machine interaction. An essential component of successful sound recognition systems is the extraction of discriminative features that capture the relevant characteristics of audio signals. The choice of feature representation significantly influences the performance of sound recognition models, affecting their ability to handle different sound classes and exhibit robustness to noise and variations.

Mel-frequency cepstral coefficients (MFCC) have been widely used as a popular feature extraction method in sound recognition tasks. MFCC leverages the logarithmically spaced Mel filterbank to approximate the nonlinear frequency response of the human auditory system, offering perceptually relevant features that emphasize relevant spectral information. However, MFCC may not fully exploit fine-grained spectral details and may not capture the frequency resolution optimally [1].

To address these limitations, this paper presents an improved feature extraction approach by combining the Constant-Q Transform (CQT) and Mel Spectrogram for Convolutional Neural Network (CNN) input in sound recognition tasks. CQT is an alternative to the traditional Short-Time Fourier Transform (STFT) that maintains a constant Q factor (quality factor) in each frequency band, providing better frequency resolution at low frequencies and higher resolution at high frequencies. On the other hand, Mel Spectrogram applies Mel filter banks directly to the magnitude spectrum, enabling a more perceptually meaningful representation of the audio signal.

The main objective of this research is to investigate the effectiveness of combining CQT and Mel Spectrogram with MFCC in enhancing the representation of audio signals for CNN-based sound recognition. The proposed approach aims to capture both the fine-grained spectral information and perceptually relevant features, ultimately improving the performance and robustness of sound recognition models.

In the subsequent sections, we provide a detailed description of the proposed methodology, including the formulation of CQT and Mel Spectrogram, the integration of these features with MFCC, and the design of the CNN architecture for sound recognition. Extensive experiments and evaluations are conducted on benchmark datasets to demonstrate the advantages and effectiveness of the proposed approach compared to the traditional MFCC-based methods.

In a previous study with the title ”Implementation of Constant-Q Transform (CQT) and Mel Spectrogram to converting Bird's Sound” [2]. Two techniques are used to convert bird sounds into digital images for classification: Constant-Q Transform (CQT) and Mel Spectrogram. The CQT maps the audio signal from the time domain to the frequency domain, converting the frequency to a log scale and the amplitude to decibels to form a spectrogram. The spectrogram is then mapped to the mel scale to form the mel spectrogram, which optimizes the frequency and ensures a consistent frequency scale between images. This digital image of bird sounds can be classified using CNN for classification of bird sounds.

The contributions of this paper lie in presenting a novel approach that leverages the complementary strengths of CQT and Mel Spectrogram to enrich the feature extraction process for sound recognition. The results showcase significant improvements in accuracy and generalization performance, indicating the potential of the proposed approach in advancing sound recognition technologies across various domains.

1.2 Rumusan Masalah

How to Improved Feature Extraction for Sound Recognition using Combined Constant-Q Transform (CQT) and Mel Spectrogram for CNN Input?

1.3 Tujuan Penelitian

Improved Feature Extraction for Sound Recognition using Combined Constant-Q Transform (CQT) and Mel Spectrogram for CNN Input

1.4 Target Penelitian

Penelitian ini diharapkan dapat bermanfaat sebagai berikut:

a. Sebagai bagian dari inovasi di bidang teknologi

b. Sebagai media informasi kepada masyarakat umum

1.5 Luaran Hasil Penelitian

Luaran hasil dari penelitian ini adalah sebagai berikut :

a. Dalam bentuk sebuah produk aplikasi.

b. Dalam bentuk jurnal nasional ber-ISSN

BAB II
METODE PENELITIAN

Data Collection Process

This study used recordings of Javanese Hawk Eagle (Nisaetus Bartelsi) vocalizations in three different contexts: normal, foraging, and mating. The sound data was collected using a sound recording device placed one meter from the birds.

Pre-emphasis

Pre-emphasis is a technique used in speech processing. It involves applying a high-pass filter to the audio signal before further processing. The purpose of pre-emphasis is to emphasize the higher frequencies in the signal and compensate for the natural attenuation of high-frequency components during speech production and transmission [3]. The pre-emphasis filter can be applied to a signal x using the first order filter in the following equation:

$$y(t) = x(t) - \alpha x(t-1) \quad \square\square\square$$

where typical values for the filter coefficient (α) are 0.95 or 0.97.

Frame the Signal

In the process of calculating MFCC, the speech signal is divided into frames. Each frame typically has a duration of 25 milliseconds with a shift of 10 milliseconds [4]. This frame-based approach allows for the analysis of the speech signal at a more granular level, capturing temporal variations and preserving the dynamic characteristics of the signal [5].

Windowing

The Hamming window is a commonly used window function in the calculation of MFCC. It is applied to each frame of the speech signal during the framing process in order to reduce spectral leakage and improve the accuracy of subsequent analysis [6]. The Hamming window is a type of tapering function that smoothly reduces the amplitude of the signal towards the edges of the frame. It is defined by the following equation:

$$W(n) = 0.54 - 0.46 \cos\left(\frac{2\pi n}{N-1}\right) \quad \square\square\square$$

where w(n) is the value of the window at sample index n, and N is the total number of samples in the frame. By applying the Hamming window to each frame, the discontinuities at the edges of the frame are minimized, reducing the spectral leakage that can occur during the Fourier transform. This helps to preserve the spectral characteristics of the signal and improve the accuracy of subsequent analysis, such as the calculation of MFCC coefficients

Implementation of Constant-Q Transform (CQT)

Constant Q Transform is a modification of STFT (Short-Time Fourier Transform). STFT is one way to convert a time domain signal into a frequency domain so that we can analyze how the frequency components change over time. CQT was designed to overcome some of the limitations of STFT [7]. The CQT transformation can be seen in equation 1 below:

$$X[\mathrm{k,n}] = \sum_{q=n-[\frac{Nk}{2}]}^{n+[Nk/2]} x(q) a_k^*\left(q - N + \frac{Nk}{2}\right) \quad \square\square\square$$

Where k is assigned values from 1 to 2 and so on, it corresponds to the indexing of the catch-frequency coefficient of CQT. $[\frac{Nk}{2}]$ Shows rounding to an infinite negative value, and a_k^*(n) show the conjugation complex of a_k (n). The basic function of a_k (n) is a waveform with complex values. Then in the atomic equation 2 the time frequency is defined as follows:

$$a_k(n) = \frac{1}{Nk}\ \omega\ \left(\frac{1}{Nk}\right) \exp\left(-j2\pi\ \frac{fk}{fs}\right) \qquad (4)$$

The value fk is the storage center frequency k, and fs denotes the sampling rate, and w (t) is a continuous window function sampled at the point determined by t. Nk is the length of the window which is inversely proportional to fk in order to achieve the same Q-factor for all k containers. The center frequency fk is defined as equation 3 follows:

$$f_k\ = f_1\ 2\ \frac{k-1}{B} \qquad (5)$$

Where f_1 is the middle frequency of the lowest frequency container, and B is the coefficient to determine the number of containers per octave. In the process, B is an important parameter to make choices when using CQT, because it determines the time-frequency resolution considerations of CQT [2].

Filter Banks

The process of constructing filter banks involves designing a set of triangular filters that are evenly spaced on the mel scale. The mel scale is a perceptual scale that approximates the human ear's frequency perception. The mel scale is nonlinear and emphasizes lower frequencies more than higher frequencies, which aligns with human auditory perception [8].

The triangular filters in the filter bank are designed to have a center frequency and bandwidth that corresponds to specific mel scale values. The Mel-scale aims to mimic the non-linear human ear perception of sound, by being more discriminative at lower frequencies and less discriminative at higher frequencies. We can convert between Hertz (f) and Mel (m) using the following equations:

$$m = 2595log_{10}\left(1 + \frac{f}{700}\right) \qquad (6)$$

$$f = 700\left(10^{m/2595} - 1\right) \qquad (7)$$

Implementation of Filterbanks to CQT output

Filterbanks are designed to mimic the way the human auditory system responds to different frequency components in sound [9]. When applied to CQT output, they help extract relevant spectral features that can be further analyzed and interpreted.These feature representations are often used in advanced audio analysis and machine learning applications, which can enhance our ability to understand and manipulate audio signals in a variety of domains [10]. Here is a more detailed explanation of each step:

- Magnitude Spectrogram of CQT value. This is done by taking the magnitude (absolute value) of the complex values in the CQT representation. The magnitude spectrogram represents the amplitude of different frequency components over time [11].

- Apply Mel Filterbanks for each Magnitude Spectrogram filterbank by taking the dot product of the filterbank with the magnitude spectrum. This will give a set of values that represent the energy in different Mel frequency bins [12].

- Logarithmic Compression to compress the dynamic range of a Mel spectrogram. The Mel spectrogram is a representation of the power spectral density of a signal in a way that mimics the human auditory system's frequency perception. Logarithmic compression is often applied to the Mel spectrogram to make it more suitable for tasks machine learning on audio data [13].

- Normalizing a Mel spectrogram is a common preprocessing step in audio signal processing and machine learning tasks. Normalization helps ensure that the data is on a consistent scale, which can be important for training machine learning models and making the data more interpretable [14].

Convert sound into audio images

Transforming audio into visual representations, the process involves transforming a normalized Mel Spectrogram into an audio image using the Matplotlib library in Python. After obtaining the Mel Spectrogram, which is a vital representation of audio data, it is first normalized to ensure uniformity in scale. Then, Matplotlib, a versatile Python library for data visualization, is employed to create an audio image. This image provides an

insightful view of the audio's spectral content, making it easier to analyze and understand. The Matplotlib library's capabilities enable users to customize and display the audio image for various applications, from audio processing to machine learning tasks, enhancing our ability to work with sound data effectively.

BAB III
PEMBAHASAN

In this study, the vocalizations of the Javanese Hawk Eagle (Nisaetus Bartelsi) were recorded in three different contexts: normal, looking for food, and looking for partner. The data was collected using a sound recording device placed under the tree where the bird was perched. A total of 12 recordings were made, each containing 4 vocalizations. The recordings were then named according to the following format: (sequence number)(bird sound condition)(.wav), for example "1Normal.wav". All recordings were saved in the same folder, which was named "clean".

Sampleset of birds sound recordings

Figure 1 illustrates an example of the sound data collected. There are 12 audio recordings for each of the four audios with different bird sound conditions. This audio file is in WAV format, which is the abbreviation of Waveform audio format.

In this study, we used librosa, a free and open-source Python library for music and audio analysis. Each sound that has been collected will be pre-emphasized. To do that we can use librosa's preemphasis function which can be seen as follows:

```
emphasized_signal = librosa.effects.preemphasis(

                        y=audio_signal,
```

In the code-box above, librosa.effects.preemphasis() takes two parameters, the first is y which contains the audio signal, then the second parameter is coef which contains the coefficient value of the pre-emphasis which is 0.95 or 0.97.

After performing pre-emphasis, we will compute the Constant-Q Transform (CQT) of the audio output signal. To do this, we can use the librosa's cqt function, which is shown below:

```
# Compute Constant Q Transform of emphasize signal
with default # window is hamming

CQT = librosa.cqt( y=emphasized_signal,

                sr=sample_rate,
                fmin=librosa.note_to_hz('C4'),
```

In the code box above, we use 7 parameters to calculate CQT using the librosa function librosa.cqt(), each parameter will be explained as follows:

- y=emphasized_signal: The y parameter specifies the input audio signal, in this case, emphasized_signal value.
- sr=sample_rate: The sr parameter sets the sample rate of the input signal, which is essential for accurate frequency analysis.
- fmin=librosa.note_to_hz('C4'): The fmin parameter sets the minimum frequency (in Hertz) for the CQT. In this code, it's set to the frequency of the note 'C4' (approximately 261.63 Hz).
- n_bins=128: The n_bins parameter determines the number of frequency bins to be used in the CQT. In this case, it's set to 128.
- bins_per_octave=24: This parameter specifies the number of bins per octave in the CQT, affecting the pitch resolution. It's set to 24, which is a common choice for audio analysis.
- hop_length=256: The hop_length parameter determines the hop size (frame shift) in samples between consecutive CQT frames. A larger value reduces time resolution but speeds up computation. In this code, it's set to 256.
- window='hamming': The window parameter defines the windowing function applied to each frame of the input signal before computing the CQT. In this case, it's set to 'hamming,' which is a commonly used window function for spectral analysis tasks.

By default, librosa provides the convenience of not implementing the frame and windowing processes, because these processes are already performed automatically by librosa from the parameters we input. The output of the librosa.cqt() function is a CQT representation of the signal. This representation can be used to analyze the frequency content of the signal and to identify audio features.

Next, we will calculate the magnitude spectrogram of the CQT signal. Magnitude spectrograms are a common analysis technique in audio signal processing that provide a visual representation of the signal's spectral content over time. They are calculated by taking the magnitude (absolute value) of the complex values in the CQT representation. This can be done as follows:

```
# Magnitude Spectrogram
```

.

In the code box above, magnitude_spectrogram is where the resulting magnitude spectrogram will be stored. The np.absolute(cqt_spectrogram) function calculates the absolute magnitude of the CQT representation using NumPy's np.absolute function. The CQT representation is the spectral information of the audio signal after applying the Constant Q Transform. Taking the absolute value ensures that we only consider the magnitude or amplitude of the complex numbers obtained from the transformation.

Once we have the magnitude spectrogram, we can apply filterbanks to it. This can be done using the melspectrogram function from the Librosa library. The melspectrogram function creates a mel spectrogram, which is a type of spectrogram that uses special filterbanks to represent audio frequencies in the mel scale. The mel scale is a frequency transformation that is designed to mimic the way the human ear hears. Mel

spectrograms are often used in audio signal processing tasks such as speech recognition and music analysis. This can be done as follows:

```
#Compute Mel Spectrogram

mel_spectrogram = librosa.feature.melspectrogram(

                          S=magnitude_spectrogram**2,
```

In the code box above, we use 4 parameters to calculate Mel Spectrogram using the librosa function librosa.feature.melspectrogram(), each parameter will be explained as follows:

- S: This is the input spectrogram. In this code, it's computed from the magnitude spectrogram (magnitude_spectrogram) squared. The magnitude spectrogram represents the magnitude (amplitude) of different frequency components of the audio signal at different points in time. Squaring it helps emphasize the energy of the different frequencies.

- sr: This is the sample rate of the audio signal, which indicates how many samples are taken per second. It's used to scale the frequency axis correctly.

- n_mels is parameter controls the resolution of the mel scale, we chose the value 128. The number 128 is a common choice for the n_mels parameter because it strikes a good balance between resolution and computational efficiency

- The fmax parameter sets the maximum frequency (in Hz) to be used when computing the Mel spectrogram. It's set to half of the sample rate (sample_rate/2), which is often done to include all audible frequencies.

Next, we apply logarithmic compression to mel spectrogram output. Applying logarithmic compression (log-scale) to a Mel spectrogram output is a common technique used in audio signal processing and analysis. It is done to enhance the representation of audio data, especially when dealing with human auditory perception, where we perceive loudness and pitch on a logarithmic scale. This can be done as follows:

```
# Apply logarithmic compression (log-scale)

mel_spectrogram_dB  = librosa.power_to_db(
```

In the code box above, we use 2 parameters to calculate logarithmic compression using the librosa function librosa.power_to_db() each parameter will be explained as follows:

- mel_spectrogram: The input Mel spectrogram you want to convert.

ref: This argument specifies the reference power level to use when computing dB. In this case, ref=np.max means that the maximum value in the input mel_spectrogram will be used as the reference power level. This is a common practice and ensures that the loudest part of the audio will be set to 0 dB,

- and all other values will be scaled relative to this loudest part.

Once we have logarithmically compressed the Mel spectrogram, we can normalize it. Normalization ensures that the data is on a consistent scale, which is important for

training machine learning models and making the data more interpretable. Normalization can be done as follows:

```
# Normalize the Mel spectrogram

mean = np.mean(mel_spectrogram_dB)
```

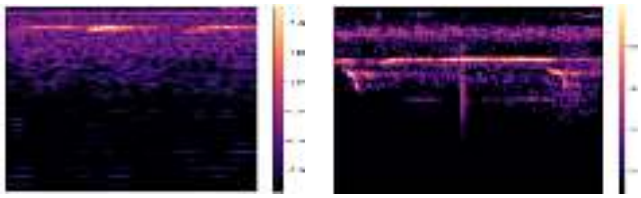
In the code box above, the given code snippet is responsible for normalizing the Mel spectrogram which has previously been converted to a logarithmic scale in decibels (dB). Here's a breakdown of the function of each piece of code:

- mean: This line calculates the mean (average) value of all the elements in the mel_spectrogram_dB. This provides a measure of the central tendency of the data.
- std: This line calculates the standard deviation of the elements in the mel_spectrogram_dB. The standard deviation measures the amount of variation or dispersion in the data. It indicates how spread out the values are from the mean.
- normalized_mel_spectrogram: This variable is created to store the result of the normalization process.

After getting normalization from the mel spectrogram, then we can visualize using the Matplotlib library. The results of the visualization using the matplotlib library can be seen in Figure 2, Figure 3, and Figure 4. The results of this visualization show the differences in the characteristics of the sounds produced by the normal, looking for food, and looking for partner.

This paper discusses the steps involved in computing the Constant-Q Transform (CQT) and filter bank, as well as the motivation behind each step. It is interesting to note that all of the steps required to compute a filter bank are motivated by the nature of the speech signal and how humans perceive it. However, there is an additional step required to compute Mel-frequency cepstral coefficients (MFCCs), which is to compute the

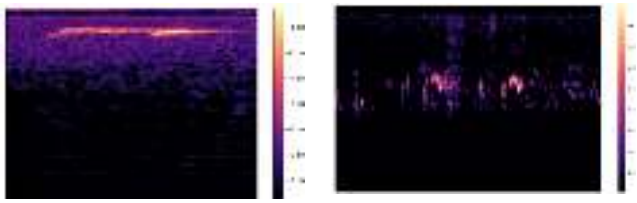
Discrete Cosine Transform (DCT). This extra step is motivated by the limitations of some machine learning algorithms [15].



(a) (b)

Spectogram of Normal Tweets (a) Basic MFCC (b) Improved MFCC

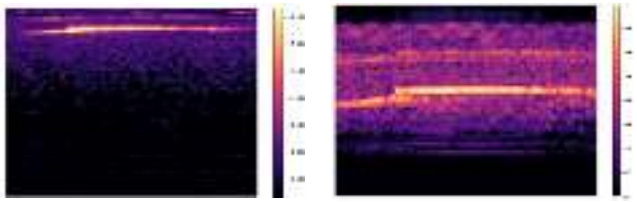
Figure 2 depicts the normal sound spectrogram of a Javanese Eagle. The sound produced has flat and lengthy characteristics with a vivid color.



(a) (b)

Spectrogram of the Looking for Partner condition (a) Basic MFCC (b) Improved MFCC

Figure 3 illustrates a Javanese Eagle's usual sound spectrogram. As seen by the red ellipse on the spectrogram, the sound created has a flat, lengthy characteristic and a very vivid color at the beginning of the sound, indicating a high frequency emitted by the Javanese Eagle.



(a) (b)

Spectrogram of the Looking for Food condition(a) Basic MFCC (b) Improved MFCC

Figure 4 illustrates a Javanese Eagle's usual sound spectrogram. The produced sound has a flat characteristic and is split into two halves, indicating that two sound tweets are occurring at the same time.

The DCT is required to decorrelate the filter bank coefficients, which is a process also known as whitening. If the steps to compute the DCT are required, then we must go back one step before performing the normalization computations. This can be seen as follows:

```
# Compute Mel-frequency cepstral coefficients (MFCCs)

n_mfcc = 13

mfcc = librosa.feature.mfcc(
```

In the code box above, the provided code is an example of how to compute Mel-frequency cepstral coefficients (MFCCs) using the Librosa library in Python. Here's a breakdown of the code:

- S=mel_spectrogram_dB: This parameter specifies the input Mel Spectrogram. The variable mel_spectrogram_dB is assumed to contain the Mel Spectrogram in decibel (dB) scale. MFCCs are typically computed from the Mel Spectrogram.

- n_mfcc=n_mfcc: This parameter specifies the number of MFCC coefficients to compute, and it uses the value stored in the n_mfcc variable.

After this process, if necessary, the MFCC output can be normalized as in the steps discussed above. Normalization of the Mel spectrogram functions to balance the spectrum and increase the Signal-to-Noise Ratio (SNR). The outcomes of the spectrogram that resulted from normalization of the Mel spectrogram.

BAB IV
PENUTUP

The improved Mel-frequency cepstral coefficients (MFCC) made the different images from the same sounds, indicating the technique works for high intonation. Combining the Constant-Q Transform (CQT) and Mel Spectrogram methods for Convolutional Neural Network (CNN) input makes this study's method for creating features for speech recognition work better. The new method suggests using the CNN model with CQT frequency representation and Mel Spectrogram as input. This gives the CNN model more useful and rich speech recognition features than the usual MFCC method. The main purpose of this study is to find out how much combining CQT, Mel Spectrogram, and MFCC can make the way audio data are represented better for CNN-based speech recognition. The suggested method aims to gather more detailed spectral data and perceptually relevant features. This should make speech recognition models work better and be more reliable.

Using a Convolutional Neural Network (CNN) is one way to sort pictures into groups. First, each sound wave from the sound data that will be labeled with CNN must be turned into a picture. Once these changes are made, a spectrogram will be made, which can show colors that match sound waves that are shorter or longer. This spectrogram was then changed into a Mel spectrogram, which tries to place time and frequency scales so that the picture is clearer and has better color. The results of this study can be used to test and train the CNN method to make it better at identifying sounds.

DAFTAR PUSTAKA

[1] Fahad, M.S., Deepak, A., Pradhan, G. and Yadav, J., 2021. DNN-HMM-based speaker-adaptive emotion recognition using MFCC and epoch-based features. Circuits, Systems, and Signal Processing, 40, pp.466-489.

[2] Permana, S.D.H. and Bintoro, K.B.Y., 2021, July. Implementation of Constant-Q Transform (CQT) and Mel Spectrogram to converting Bird's Sound. In 2021 IEEE International Conference on Communication, Networks and Satellite (COMNETSAT) (pp. 52-56). IEEE.

[3] J. Hendry, B. Sumanto, Y. Mileniandi, & P. Wening, "Implementation of pre-emphasis analog filter on raspberry pi using zero-order hold discretization as a pre-processing to humanoids' commands recognition", Jurnal Listrik, Instrumentasi, Dan Elektronika Terapan, vol. 3, no. 2, 2022.

[4] D. Ortega and N. Vu, "Lexico-acoustic neural-based models for dialog act classification", 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2018.

[5] T. Lin and Y. Zhang, "Speaker recognition based on long-term acoustic features with analysis sparse representation", IEEE Access, vol. 7, p. 87439-87447, 2019.

[6] N. Benayad and Z. Soumaya, "Features selection by genetic algorithm optimization with k-nearest neighbour and learning ensemble to predict parkinson disease", International Journal of Electrical and Computer Engineering (IJECE), vol. 12, no. 2, p. 1982, 2022.

[7] A. Sehgal and N. Kehtarnavaz, "A convolutional neural network smartphone app for real-time voice activity detection", IEEE Access, vol. 6, p. 9017-9026, 2018.

[8] S. Bhosale, I. Sheikh, S. Dumpala, & S. Kopparapu, "End-to-end spoken language understanding: bootstrapping in low resource scenarios", Interspeech 2019, 2019.

[9] C. Kopp-Scheinpflug, J. Sinclair, & J. Linden, "When sound stops: offset responses in the auditory system", Trends in Neurosciences, vol. 41, no. 10, p. 712-728, 2018.

[10] I. López-Espejo and J. Jensen, "Improved external speaker-robust keyword spotting for hearing assistive devices", IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 28, p. 1233-1247, 2020.

[11] K. Khoria, A. Patil, & H. Patil, "On significance of constant-q transform for pop noise detection", Computer Speech &Amp; Language, vol. 77, p. 101421, 2023.

[12] N. Zeghidour, N. Usunier, I. Kokkinos, T. Schaiz, G. Synnaeve, & E. Dupoux, "Learning filterbanks from raw speech for phone recognition",, 2018.

[13] M. Angrick, C. Herff, E. Mugler, M. Tate, M. Slutzky, D. Krusienskiet al., "Speech synthesis from ecog using densely connected 3d convolutional neural networks", Journal of Neural Engineering, vol. 16, no. 3, p. 036019, 2019.

[14] N. Mehlman, A. Sreeram, R. Peri, & S. Narayanan, "Mel frequency spectral domain defenses against adversarial attacks on speech recognition systems", 2022.

[15] Z. Abdul and A. Al-Talabani, "Mel frequency cepstral coefficient and its applications: a review", IEEE Access, vol. 10, p. 122136-122158, 2022.