Aplikasi Graf Time-Frequency Spektrogram dan Hash dalam Analisis Kemiripan Audio

Indraswara Galih Jayanegara - 13522119[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13522119@std.stei.itb.ac.id

***Abstract*— Audio adalah sesuatu yang sudah biasa kita dengar di zaman ini. Audio juga telah menjadi bagian dari kehidupan kita sehari-hari Baik itu lagu, podcast, telepon, rekaman, dan lain sebagainya. Audio sejatinya adalah sebuah energi mekanik. Namun, dari energi tersebut bisa kita gambarkan menjadi sebuah graf seperti pada fenomena-fenomena alam lainnya. Pada makalah kali ini akan disajikan sebuah audio yang diubah menjadi bentuk graf time-frequency. Pada lagu, kita biasanya kebingungan untuk mencari judul dari lagu tersebut. Aplikasi dari graf time-frequency atau spektrogram yang merepresentasikan graf dari sebuah audio yang nantinya akan dicari kemiripannya. Hal ini memanfaatkan pula hash function sehingga tiap lagu memiliki keunikannya masing-masing dan mudah untuk diidentifikasi.**

***Keywords*— Spektrogram, Hash, Graf, Audio**

I. PENDAHULUAN

Suara adalah fenomena kompleks yang melibatkan berbagai elemen dalam fisika. Suara, adalah gelombang mekanis yang dihasilkan oleh getaran. Ketika sebuah objek bergetar, misalnya, senar gitar atau bibir seseorang saat berbicara, getaran tersebut mengganggu partikel-partikel di sekitarnya. Inilah awal dari getaran suarata yang kemudian ditransfer melalui medium, seperti udara, zat cair, dan zat padat. Ketika partikel-partikel ini bergerak, mereka saling mempengaruhi satu sama lain, menciptakan tekanan yang berbeda yang menyebar ke segala arah. Inilah yang disebut sebagai gelombang suara. Namun, untuk kita dapat mendengar suara, gelombang tersebut harus mencapai telinga kita. Telinga kita memiliki struktur yang memungkinkan kita mendeteksi gelombang suara ini. Ketika gelombang suara mencapai telinga, mereka pertama-tama melewati saluran pendengaran dan akhirnya mencapai gendang telinga. Gendang telinga kemudian bergetar sesuai dengan tekanan gelombang suara yang masuk, mengubahnya menjadi sinyal listrik yang akhirnya diinterpretasikan oleh otak sebagai suara. Hal ini menandakan bahwa suara dapat dideteksi layaknya fenomena-fenomena lain yang ada di alam sehingga dapat dibuat pemodelan yang menggambarkan audio tersebut.

Sebuah spektrogram yang merupakan aplikasi dari graf *time-frequency* adalah cara visual untuk menggambarkan kekuatan sinyal dari suatu sinyal sepanjang waktu pada berbagai frekuensi yang ada dalam gelombang tertentu. Tidak hanya bisa dilihat apakah ada lebih atau kurang energi pada, misalnya, 2 Hz dibandingkan dengan 10 Hz, tetapi juga bisa dilihat bagaimana tingkat energi bervariasi seiring waktu. Selain itu, spektrogram umum digunakan untuk menampilkan frekuensi gelombang suara yang dihasilkan oleh manusia, mesin, hewan, pesawat jet, dan sebagainya. seperti yang direkam dengan mikrofon.

Sebuah data dapat bisa diubah bentuknya menjadi *key* dan value menggunakan *hash function*. Lalu, jika sebuah spektrogram yang merupakan representasi data dari sebuah audio diubah menggunakan hash function lalu disimpan di sebuah penyimpanan sederhana. Hal ini akan menyebabkan pencarian dari key untuk audio yang dicari akan semakin cepat. Namun, jika kita telaah lebih dalam hal ini bisa membuat kita untuk mencari sebuah audio yang audio tersebut akan diubah pula dengan *hash function* dan kita cari kemiripannya berdasarkan *key* yang sudah ada di dalam database.

II. DASAR TEORI

A. Graf

Graf didefinisikan dengan sebuah pasangan himpunan yang terdiri dari himpunan yang tidak kosong dari simpul-simpul yang disebut dengan (*V/Vertice*), dan sisi yang disimbolkan dengan (E */ Edge*). Yang mana kombinasi dari V dan E akan menghasilkan G = (V, E).

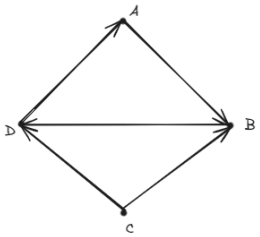
Ada beberapa tipe dari Graf berdasarkan representasi arahnya, Graf berarah dan Tidak berarah.

1. Graf Berarah

   Graf berarah adalah jenis graf di mana setiap sisi memiliki arah yang ditentukan. Artinya, sisi-sisi dalam graf ini memiliki orientasi atau panah yang menunjukkan arah dari satu simpul ke simpul lainnya.

   Dalam graf berarah, jika terdapat sisi yang menghubungkan simpul A ke simpul B, arahnya akan jelas menunjukkan bahwa perjalanan atau hubungan dari simpul A menuju simpul B, tetapi tidak sebaliknya, kecuali ada sisi lain yang menghubungkan kembali dari B ke A.

   Misalnya, dalam representasi graf berarah untuk jaringan transportasi antar-kota, jika ada jalan satu arah dari kota A ke kota B, itu akan direpresentasikan oleh sisi dengan panah yang menunjukkan arah dari A ke B. Namun, tidak ada jalur langsung yang mengizinkan perjalanan balik dari B ke A kecuali ada sisi lain yang menghubungkan secara terpisah.



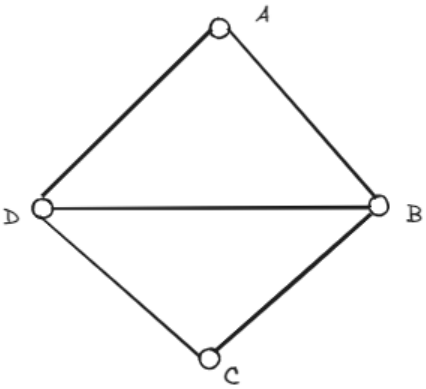
Gambar 1 graf berarah

Sumber: dokumen penulis

Graf berarah memberikan informasi tentang arah dari interaksi, aliran, atau hubungan antara entitas yang direpresentasikan oleh simpul-simpulnya. Hal ini sangat penting dalam banyak aplikasi seperti jaringan komputer, diagram aliran, perencanaan rute, dan sistem terdistribusi di mana arah perjalanan atau hubungan memiliki signifikansi dan pengaruh pada analisis yang dilakukan terhadap graf tersebut.

2. Graf Tidak Berarah

Graf berarah adalah jenis graf yang tidak memiliki arah dari *edge*-nya. Pada gambarnya sendiri graf beraraf berarah tidak memiliki arah pula. Contoh dari penggunaan graf tak berarah adalah pada struktur molekul, Sistem Transportasi, dan lain-lain.

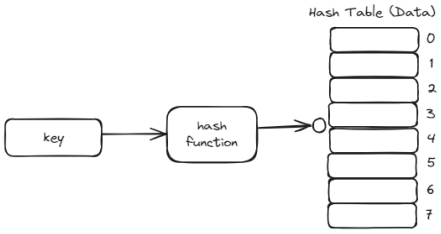


Gambar 2 Graf tak-berarah

Sumber: dokumen penulis

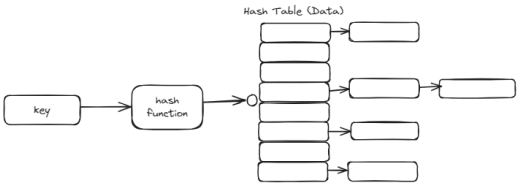
B. Hash

Hash merupakan fungsi yang menerima sebuah input dari penerima yang nantinya akan diubah menjadi key value dan mencari key value yang mirip untuk mencari value yang dicari. Lebih spesifik lagi hash function adalah sebuah metode komputasi yang memetakan sekumpulan data menjadi data yang jumlahnya lebih pasti. Hash sendiri biasanya digunakan di dalam kriptografi. Beberapa jenis yang umum dari hash adalah MD5, SHA-1, SHA-2, dan SHA-3. Keluaran dari fungsi hash disebut dengan nilai hash. Fungsi hash dirancang untuk menhasilkan nilai hash yang unik. Namun, dalam beberapa kasus sebuah input akan memiliki hasil atau nilai hash yang sama, ini yang dinamakan dengan collision. Ada beberapa Teknik untuk mengatasi hal ini, diantaranya adalah Teknik *chaining* atau menggunakan list dengan struktur berkait untuk menampung data dengan nilai hash yang sama.



Gambar 3 Hash Function

Sumber: dokumen penulis



Gambar 4 hash dengan chain

Sumber: dokumen penulis

Gambar di atas merepresentasikan fungsi hash dengan chain untuk menghindari *collision* atau memiliki data yang sama dari *key* yang berbeda.

C. Graf Time-Frequency

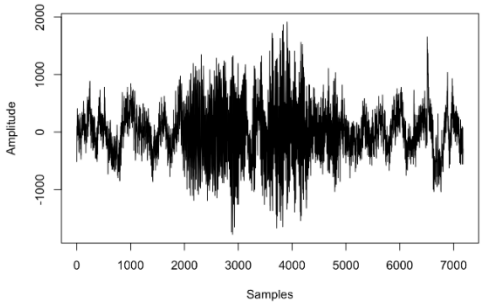
Graf *time-frequency* adalah teknik analisis sinyal yang mempelajari sinyal dalam domain waktu dan frekuensi secara simultan. Teknik ini memungkinkan kita untuk memvisualisasikan bagaimana karakteristik frekuensi sinyal berubah seiring waktu.

Graf time-frequency sangat berguna dalam berbagai aplikasi, seperti pemrosesan sinyal radio frekuensi (RF). Teknik analisis sinyal ini dapat digunakan untuk mendapatkan informasi penting tentang sinyal, seperti frekuensi, amplitudo, dan fase.

D. Spektrogram

Spektrogram adalah berasal dari spektrum dan spektrumnya adalah frekuensi yang dimiliki audio Spektrogram adalah gambar yang merepresentasikan spektrum frekuensi audio yang dideteksi berdasarkan waktu. Spektrogram sendiri adalah aplikasi dari Graf *time-frequency*.

Spektrogram memungkinkan untuk melihat bagaimana berbagai frekuensi berkontribusi terhadap sinyal audio pada setiap waktu tertentu, memungkinkan analisis yang mendalam untuk mengetahui kompleksitas karakteristik audio secara simultan dalam domain frekuensi dan waktu.



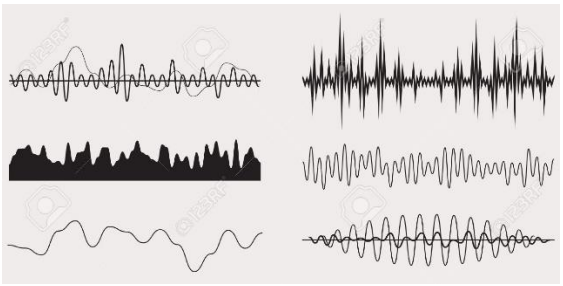
Gambar 5 Spektrogram

Sumber: https://hansenjohnson.org/post/spektrograms-in-r/

E. Audio

Audio adalah suara yang berada dalam rentang frekuensi pendengaran manusia. Ini bagian yang dapat didengar dari spektru frekuensi suara. Suara adalah bentuk energi mekanik yang bergerak dalam gelombang melalui udara dan benda lainnya. Hal ini diukur dalam hertz (Hz), di mana satu Hz sama dengan satu siklus gelombang penuh per detik.

Manusia hanya mampu mendengar sebagian kecil dari spektrum frekuensi suara. Biasanya manusia hanya mampu mendengar rentang frekuensi 20-20.000 Hz.



Gambar 6 Representasi Audio dalam penglihatan manusia

Sumber: https://www.123rf.com/photo_44037116_audio-music-sound-wave-vector-set.html

F. WAV (Waveform Audio Format)

WAV adalah singkatan dari *waveform audio format* yang merupakan format standar yang dikembangkan oleh Microsoft dan IBM. WAV adalah varian dari bitstream RIFF dan mirip dengna format IFF dan AIFF. Wav kompatibel dengna sistem operasi Windows dan Mac. Perbedaannya dengan format mp3 adalah mp3 lebih kecil karena sudah dikompres, sedangkan wav tidak sehingga ukurannya cenderung lebih besar dibandingkan mp3 dan juga menyebabkan wav memiliki kualitas suara yang lebih baik dibandingkan dengan mp3. Dalam kehidupan sehari-hari mp3 cenderung lebih banyak digunakan karena ukuran file-nya lebih kecil sehingga cukup bagus untuk disimpan dengan jumlah besar di perangkat yang memiliki memori terbatas. Penulis akan melakukan analisisnya menggunakan file audio dengan ekstensi wav.

III. IMPLEMENTASI DAN PEMBAHASAN

A. Konversi Audio

Spektrogram audio adalah representasi visual dari frekuensi-suara terhadap waktu. Ini menggambarkan bagaimana energi frekuensi berbeda tersebar dalam rentang waktu tertentu dari sebuah sinyal audio. Secara umum, sumbu vertikal pada spektrogram mewakili frekuensi suara, diukur dalam Hertz (Hz) atau dalam desibel (dB), sementara sumbu horizontal mewakili waktu. Intensitas warna atau kecerahan pada spektrogram menunjukkan seberapa banyak energi frekuensi yang hadir pada waktu dan frekuensi tertentu.

Area dengan warna yang lebih gelap atau lebih terang menunjukkan kekuatan atau intensitas yang lebih tinggi pada frekuensi tersebut pada waktu tertentu. Spektrogram audio sangat berguna dalam analisis sinyal audio, seperti dalam identifikasi pola suara, pengenalan ucapan, dan juga dapat membantu mendeteksi sumber suara atau gangguan dalam rekaman. Ini adalah alat visual yang berguna untuk memahami komposisi frekuensi suara dari waktu ke waktu.

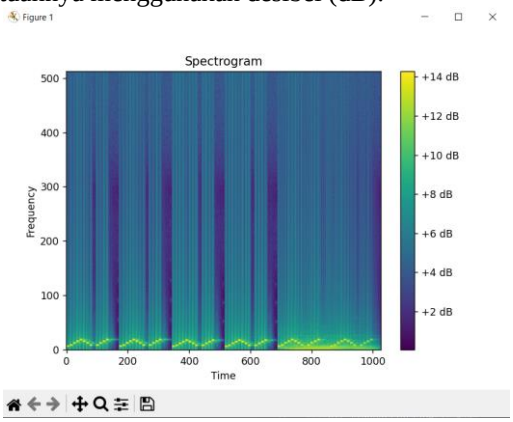
Mengubah Audio menjadi Spektrogram bisa dilakukan dengan menggunakan Bahasa pemrograman, berikut cuplikan code yang Mengubah audio dalam bentuk wav menjadi sebuah spektrogram. Untuk mengubah audio menjadi dalam bentuk data graf akan digunakan library bawaan dari python numpy, matplotlib, dan scipy.io. Hasil dari konversinya akan divisualisasikan dengan library matplotlib.

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.io import wavfile

def spectrogram(audio_file):
    rate, data = wavfile.read(audio_file)
    if len(data.shape) > 1:
        data = data.mean(axis=1)

    window_size = 1024
    hop_size = 512

    n_samples = len(data)
    n_overlap = window_size - hop_size
    n_windows = (n_samples - n_overlap) // hop_size

    spectrogram = np.zeros((n_windows, window_size // 2 + 1))

    for i in range(n_windows):
        start = i * hop_size
        end = start + window_size
        segment = data[start:end]
        windowed_segment = segment * np.hamming(window_size)
        spectrum = np.abs(np.fft.rfft(windowed_segment, window_size))
        spectrogram[i] = spectrum

    plt.figure(figsize=(7, 5))
    plt.imshow(np.log1p(spectrogram.T), aspect='auto', origin='lower')
    plt.colorbar(format='%+2.0f dB')
    plt.title('Spectrogram')
    plt.xlabel('Time')
    plt.ylabel('Frequency')
    plt.show()

audio_file_path = './data/file_example_WAV_2MG.wav'
spectrogram(audio_file_path)
```

Gambar 7 Code convert audio ke spektrogram

Sumber: dokumen penulis

Hasil yang didapat dan divisualisasikan adalah terdapat sumbu-y yang merupakan *frequency*, Sumbu-x merupakan waktu dan terdapat indikator untuk kegelapannya yang satuannya menggunakan desibel (dB).



Gambar 8 Hasil convert audio ke spektrogram

Sumber: dokumen penulis

B. Hash Function

Fungsi Hash di python mengubah input menjadi sebuah key yang unik contohnya seperti ini. Kode ini Mengubah string “Hello, world!” menjadi sebuah key yang hasilnya adalah sebuah integer.

```
def simple_hash(input_string):
    return hash(input_string)


print(simple_hash("Hello, world!"))
```

Gambar 9 simple hash function

Sumber: dokumen penulis

```
D:\Indra's\Programming\Spektrogram>py simple-hashfunc.py
-8521675969404604366
```

Gambar 10 hasil simple hash function

Sumber: dokumen penulis

Namun, fungsi hash ini menghasilkan nilai yang berbeda-beda setiap dijalankan sehingga tidak memiliki keunikan dalam *key*-nya. Kita tidak mau spektrogram yang dihasilkan dari audio memiliki nilai yang berbeda-beda setiap prosesnya sehingga digunakan teknik hashing sha256 yang setiap input yang dimasukkan akan memiliki nilai yang unik. Hal ini dilakukan agar setiap audio memiliki *key*-nya masing-masing sehingga saat mencari *key*, *key* tersebut sesuai dengan lagu yang dicari dan tidak terjadi miss dalam *comparing* atau menghasilkan audio yang tidak sesuai dengan audio yang dicek.

```
concatenated_hash = b''
    for row in spectrogram_array:
        row_bytes = row.tobytes()
        sha256_hash = hashlib.sha256(row_bytes).hexdigest().encode('utf-8')
        concatenated_hash += sha256_hash

    unique_hash = concatenated_hash.hex()
    truncated_Hash = unique_hash[:keyLength]
```

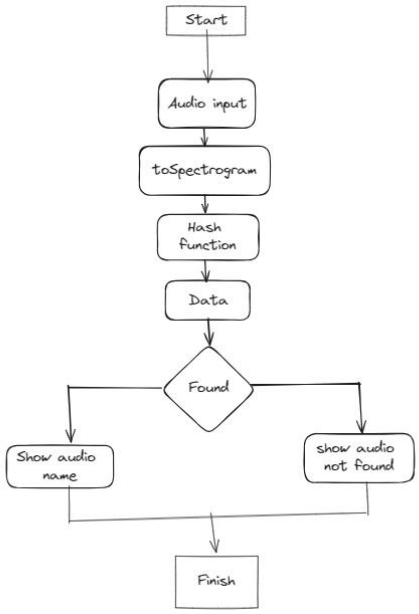
Gambar 11 Cuplikan kode hash dengan teknik sha256

Sumber: dokumen penulis

C. Hashing Value Spektrogram

Dalam python akan diubah *spektrogram* yang sudah dihasilkan dari audio yang hasil dari spektrogram tadi akan diubah menjadi sebuah hashkey, untuk mendapatkan hashkey yang unik akan digunakan library bawaan dari python yaitu hashlib. Bisa dilihat dari spektrogramHash akan dihasilkan sebuah tuple yang berisi str dan str yang mana str pertama tadi adalah hashkey sedangkan str kedua adalah value-nya yang mana itu adalah nama file. Namun, disini dilakukan pemotongan dari key dari hash yang dihasilkan karena terlalu panjang. Jadi, hanya akan digunakan 100 angka paling awal dari key yang dihasilkan. Pada program ini tidak digunakan teknik *chaining* atau teknik yang digunakan untuk menghindari *collision* pada hash, tetapi digunakan sha256 agar masing-masing input memiliki *key* yang unik. Teknik hashing langsung menggunakan library yang sudah disediakan oleh python.

Alur dari programnya sendiri seperti yang ada pada gambar. User memasukkan input berupa path yang menunjukkan audio itu berada di mana. Lalu Mengubah audio tadi menjadi bentuk spektrogram. Setelah itu, hasil dari spektrogram tadi dimasukkan ke dalam hash function yang nantinya akan menghasilkan key dan value dari spektrogram tadi. *Key*-nya sendiri berupa string angka yang panjangnya 100 dan *value*-nya adalah nama file itu sendiri.



Gambar 12 alur program utama

Sumber: dokumen penulis

Gambar 13 Alur checking key

Sumber: dokumen penulis

Dalam mengecek audio, pengecekannya akan dibagi menjadi dua mengecek terlebih dahulu global database yang sudah ada sebelumnya dan yang kedua mengecek dataset dari user yang dimasukkan secara manual. Pada bagian ini tidak diperlukan untuk menampilkan graf *time-frequency* seperti pada bagian sebelumnya karena di-bagian ini kita hanya memperhatikan *key* yang sudah dihasilkan.

```python
def spectrogramHash(audio_file: str, keyLength: int = 100) -> dict[str, str]:
    data, _ = sf.read(audio_file)

    if len(data.shape) > 1:
        data = data.mean(axis=1)

    window_size = 1024
    hop_size = 512

    n_samples = len(data)
    n_overlap = window_size - hop_size
    n_windows = (n_samples - n_overlap) // hop_size

    spectrogram = []

    for i in range(n_windows):
        start = i * hop_size
        end = start + window_size
        segment = data[start:end]
        windowed_segment = segment * np.hamming(window_size)
        spectrum = np.abs(np.fft.rfft(windowed_segment, window_size))
        spectrogram.append(spectrum)

    spectrogram_array = np.array(spectrogram)

    concatenated_hash = b''
    for row in spectrogram_array:
        row_bytes = row.tobytes()
        sha256_hash = hashlib.sha256(row_bytes).hexdigest().encode('utf-8')
        concatenated_hash += sha256_hash

    unique_hash = concatenated_hash.hex()
    truncated_Hash = unique_hash[:keyLength]

    return truncated_Hash, audio_file
```

Gambar 14 Cuplikan code spektrogramHash

Sumber: dokumen penulis

Key dengan Panjang 100 string ini bisa untuk menyimpan sekitar

$$10! = 10 \times 9 \times 8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1$$
$$10! = 3.628.800 \textit{ audio}$$

key-nya sendiri bisa diatur panjangnya dengan maksimal panjangnya adalah 4300, tapi kita tidak butuh sebanyak itu untuk makalah ini cukup 100 saja. Berikut adalah hasil kompilasi dari programnya untuk folder audio yang mana akan dihasilkan key dan value yang mana key-nya adalah 100 angka yang dihasilkan dari spektrogram dan *value*-nya adalah nama file dengan ekstensi wav.

Gambar 15 key dan value

Sumber: dokumen penulis

D. Comparing

Audio yang telah diproses menjadi spektrogram dan diubah menjadi hash akan dilakukan comparing berdasarkan key yang telah didapat. Jika key ketemu dengan key yang sama maka akan dihasilkan nama file dengan audio yang sama.

```python
def compareToDatabase(file: str, folderPath: str, databaseGlobal: list[dict[str, str]]) -> bool:
    checker_hash, fileValue = spectrogramHash(file)
    isFound: bool = False
    if databaseGlobal is not None:
        for item in databaseGlobal:
            if item['key'] == checker_hash:
                playAudio(item['file_name'])
                playAudio(fileValue)
                print("Found a matching entry in the database:", item['file_name'])
                isFound = True
                return True

    database: list[dict[str, str]] = toDatabase(folderPath)
    saveDatabase(databaseGlobal, database)
    for entry in database:
        if entry['key'] == checker_hash:
            playAudio(entry['file_name'])
            playAudio(fileValue)
            print("Found a matching entry in the database:", entry['file_name'])
            return True

    print("404 not found")
    return False
```

Gambar 16 code comparing audio

Sumber: dokumen penulis

Gambar 17 audio ada

Sumber: dokumen penulis

Jika tidak ditemukan audio di dataset yang diinput oleh user maupun yang berada di global database yang sesuai dengan yang dicari akan dimunculkan pesan yang menunjukkan audio tidak ada yang sesuai.

Gambar 18 audio tidak ada

Sumber: dokumen penulis

Perlu diketahui bahwa pengecekan ini hanya sebatas pada file audio dengan format .wav karena jika pencarian dan pengecekan menggunakan file mp3 *hashkey*-nya akan sama antar file sehingga menyebabkan miss atau salah mengecek lagu. Hal ini juga tetap terjadi meskipun sudah dilakukan konversi dari file audio dengan ekstensi mp3 ke file audio dengan ekstensi wav

Gambar 19 hashkey mp3

Sumber: dokumen penulis

E. Database

Database dibutuhkan untuk menyimpan input dari user saat sehingga pemrosesan nantinya bisa lebih cepat tidak perlu lagi menunggu pengubahan dari audio ke spektrogram dan perhitungan spekrogram menjadi hash. Saat pertama kali dijalankan program memerlukan dataset input dari user karena di dalam global database yang ada di file *csv* atau *config* tidak memiliki apapun sehingga perlu prosesnya mungkin lebih lama dibandingkan jika sudah terdapat data di dalam file config.

Gambar 20 Database audio

Sumber: dokumen penulis

IV. KEKURANGAN

Kekurangan dari topik makalah ini adalah makalah ini lebih condong ke hash daripada spektrogram itu sendiri karena spektrogram disini digunakan untuk Mengubah audio tadi menjadi sebuah objek yang bisa dimanipulasi secara matematika sehingga didapatkan sebuah key dari hash yang unik sehingga dapat dilakukan perbandingan untuk mencari audio yang mirip. Program ini juga terbatas untuk mendeteksi file audio yang memiliki ekstensi wav saja. Penulis sudah mencoba melakukan *converting* dari file mp3 ke wav. Namun, hasilnya gagal. Bisa dilihat hashkey yang dihasilkan sama semua, sehingga saat *comparing* lagu terjadi *miss* sehingga audio yang dimaksudkan justru gagal untuk dideteksi.

V. KESIMPULAN

Graf *Time-Frequency* memiliki kegunaan yang dapat digunakan untuk persoalan mencari audio atau lagu yang tidak kita ketahui judulnya menggunakan spektrogram. Apalagi saat ini didukung dengan berkembangnya berbagai Bahasa pemrograman yang dapat digunakan untuk mengimplementasikan aplikasi dari Graf *Time-Frequency*. Dengan menggunakan bahasa python sebagai alat untuk mengimplementasikan graf *time-frequency* atau spektrogram hasil yang didapatkan adalah sebuah audio dengan format wav dapat diubah menjadi spektrogram yang nantinya akan dimasukkan ke sebuah hash function dan didapatkan *key* lalu disimpan di sebuah file *config* atau *csv*, *key* dan *value*-nya sehingga nantinya bisa dicari audio yang sama berdasarkan *key* yang didapatkan dari sample audio yang akan dicari audionya yang sama. Jadi, mengubah audio menjadi bentuk spektrogram dan mengubahnya dengan hash function untuk mencari kemiripan antar audio dapat dilakukan dengan aplikasi graf *time-frequency* atau spektrogram.

VI. PENUTUP

Pada akhir makalah ini, saya ingin menyampaikan rasa Syukur saya kepada Tuhan Yang Maha Esa atas kelancaran penyelesaian makalah IF2120 Matematika Diskrit ini. Saya juga mengucapkan terima kaasih kepada Ibu Fariska Zakhralativa Ruskanda, S.T., M.T., atas bimbingannya selama perkuliahan sehingga saya bisa menyelesaikan makalah ini. Saya juga mengucapkan terima kasih kepada keluarga, teman, dan seluruh pihak terkait yang telah memberikan kontribusi dalam penyelesaian makalah ini. Saya berharap makalah ini memberikan manfaat tidak hanya pada civitas akademika ITB, tetapi juga berguna untuk pembaca-pembaca lainnya.

REFERENSI

[1] Fulop, Sean A., and Kelly Fitz. "A spectrogram for the twenty-first century." *Acoustics today* 2.3 (2006): 26-33, https://acousticslab.org/learnmoresra/files/fulopfitz2006acousticstoday2-3.pdf, diakses pada 8 Desember 2023.

[2] Begué, Matthew. "On a time-frequency a pproach to translation on finite graphs." *2015 International Conference on Sampling Theory and Applications (SampTA)*. IEEE, 2015, https://ieeexplore.ieee.org/abstract/document/7148839, diakses pada 8 Desember 2023.

[3] Merkle, Ralph C. "A fast software one-way hash function." *Journal of Cryptology* 3 (1990): 43-58, https://link.springer.com/article/10.1007/BF00203968, diakses pada 8 Desember 2023.

[4] https://scicoding.com/how-to-do-spektrogram-in-python/, diakses pada 8 Desember 2023.

[5] https://hansenjohnson.org/post/spektrograms-in-r/, diakses pada 8 Desember 2023.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 11 Desember 2023

Indraswara Galih Jayanegara (13522119)

Lampiran

1. Source Code: https://github.com/Indraswara/Spectrogram.