LAPORAN TUGAS BESAR II

IF2123 ALJABAR LINIER DAN GEOMETRI

IMAGE RETRIEVAL DAN MUSIC INFORMATION RETRIEVAL MENGGUNAKAN PCA DAN VEKTOR

Dosen : Ir. Rila Mandala, M.Eng., Ph.D.

Asisten Pembimbing: Suthasoma Mahardika Munthe (13522098)

Disusun Oleh :

Kelompok 3 - Barat Jauh

Alfian Hanif Fitria Yustanto	13523073
Heleni Gratia M Tampubolon	13523107
Ahmad Wicaksono	13523121

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

DESEMBER 2024

DAFTAR ISI

BAB 1

DESKRIPSI MASALAH

1. 1 Pemrosesan Suara dan Gambar

Suara selalu menjadi hal yang paling penting dalam kehidupan manusia. Manusia berbicara mengeluarkan suara dan mendengarkan suatu suara untuk diresap ke otak dan mencari informasi dari suara tersebut. Suara juga bisa dijadikan orang-orang di dunia ini sebuah media untuk membuat karya seni. Contohnya adalah alat mendeteksi lagu. Manusia bisa mendeteksi suara dengan menggunakan indera pendengar dan memberikan kesimpulan akan apa jenis suara tersebut melalui respon dari otak. Sama seperti manusia, teknologi juga bisa mendeteksi suara dan memberikan jawaban mereka melalui algoritma-algoritma yang beragam bahkan bisa melebihi kapabilitas manusia. Dengan menggunakan algoritma apapun, konsep dari pendeteksi dan interpretasi suara itu bisa juga disebut dengan sistem temu balik suara atau bisa disebut juga dengan *audio retrieval system*. Banyak aplikasi yang menggunakan konsep sistem temu balik contohnya adalah Shazam.

Gambar 1.1.1 Shazam sebagai aplikasi *audio retrieval system*

Selain suara, manusia juga memiliki penglihatan sebagai salah satu inderanya dan bisa melihat warna dan gambar yang bermacam-macam. Teknologi komputasi juga memiliki kapabilitas yang sama dan bisa melihat gambar sama seperti kita, tetapi teknologi seperti ini

juga bisa merepresentasikan gambar tersebut sebagai beragam-ragam angka yang bisa disebut juga fitur. Tahun ke tahun, *image processing* selalu menjadi fokus utama dari tugas besar 2 Algeo. Algoritma yang digunakan adalah Eigenvalue, Cosine Similarity, Euclidean Distance, dll.

Anda sudah melewati Tugas Besar 1 yaitu tentang matriks dan implementasi terhadap berbagai hal. Matriks adalah salah satu komponen yang penting dalam aplikasi aljabar vektor. Di dalam Tugas Besar 2 ini, anda diminta untuk membuat semacam aplikasi Shazam yaitu sebuah aplikasi yang meminta input lagu dan aplikasi tersebut mendeteksi apa nama dari lagu tersebut dan beberapa detail lainnya. Pada tugas besar ini, anda akan menggunakan aljabar vektor untuk mencari perbandingan antar satu audio dengan audio yang lain. Anda akan menggunakan konsep yang bernama *Music Information Retrieval* atau MIR untuk mencari dan mengidentifikasi suara berdasarkan fitur-fitur yang dimilikinya. Tidak hanya itu, anda juga akan menggunakan konsep Principal Component Analysis (PCA) untuk mencari kumpulan audio melalui deteksi wajah berbagai orang (anggap saja mereka sebagai seorang penyanyi).

1.2 Information Retrieval

Information Retrieval adalah konsep meminta informasi dari sebuah data dengan memasukkan data tertentu. Pada tugas besar ini, anda akan berkutik dengan 2 jenis Information Retrieval. *Image Retrieval* dan *Music Information Retrieval*. *Image Retrieval* adalah konsep untuk memasukkan sebuah input gambar dan berharap mendapatkan gambar yang ada di data sesuai dengan informasi dan perhitungan yang diinginkan. Sedangkan *Music Information Retrieval* (MIR) adalah konsep untuk memasukkan sebuah input audio dan berharap mendapatkan audio yang ada di data sesuai dengan informasi dan perhitungan yang diinginkan. Pada tugas besar kali ini, kalian akan mengimplementasikan *Image Retrieval* dengan menggunakan Principal Component Analysis dan *Music Information Retrieval* dengan menggunakan humming.

BAB 2
LANDASAN TEORI

2.1 Sistem Temu Balik Suara (MIR)

Sistem temu balik suara atau Music Information Retrieval (MIR) adalah cabang ilmu yang berada di persimpangan antara ilmu komputer, musikologi, dan teknik audio, yang bertujuan untuk mengembangkan teknik untuk menganalisis, mencari, dan mengklasifikasikan musik secara otomatis dari berbagai sumber. Sistem ini sangat relevan dengan kebutuhan masyarakat modern, terutama dengan semakin besarnya jumlah data musik yang tersedia dalam bentuk digital.

Dalam sistem MIR, beberapa fitur yang dianalisis adalah:

1. Nada (Pitch)

Nada mengacu pada frekuensi dasar dari suara atau musik. Analisis nada digunakan untuk mengenali melodi dari musik.

2. Tempo

Tempo merupakan kecepatan atau ritme dari suatu lagu yang diekstrak dari pola sinyal audio. Analisis ini membantu dalam penentuan genre musik dan deteksi beat.

3. Ritme (Rhythm)

Ritme merupakan pola ketukan dan durasi dalam sebuah lagu. Fitur ini akan membantu dalam memahami struktur musik dan pencocokan lagu serupa.

4. Timbre

Timbre merupakan kualitas suara yang membedakan satu instrumen dari instrumen lainnya meskipun memiliki pitch yang sama.

5. Spektrogram

Spektrogram adalah representasi visual dari spektrum frekuensi suara yang berubah terhadap waktu.

MIR memiliki beragam aplikasi yang luas dan relevan dengan kebutuhan modern, seperti pengenalan musik (*Music Recognition*), klasifikasi genre subgenre, dan sistem rekomendasi musik.

2.2 Metode Ekstraksi Fitur Berdasarkan Humming

Dalam sistem temu balik musik berbasis Query by Humming (QBH) merupakan metode ekstraksi fitur yang memainkan peran penting untuk mencocokkan melodi yang dinyanyikan pengguna dengan database musik. Ekstraksi fitur berfungsi untuk mengidentifikasi dan merepresentasikan informasi penting dari input suara (humming) secara matematis agar dapat dibandingkan dengan fitur-fitur dalam database.

Ekstraksi fitur melibatkan pemrosesan audio untuk menghasilkan representasi numerik yang merepresentasikan karakteristik penting dari suara input. Audio yang digunakan adalah dalam bentuk file MIDI yang selanjutnya digunakan metode windowing untuk membagikan melodi lagu menjadi segmen dalam rentang beat yang diinginkan. Proses ini sekaligus dilakukan normalisasi tempo dan pitch untuk mengurangi variasi humming (menghilangkan noise dan menyesuaikan amplitudo suara).

Pemrosesan audio dalam sistem query by humming menggunakan file MIDI dengan fokus pada track melodi utama, umumnya di Channel 1. Setiap file MIDI diproses menggunakan metode windowing yang membagi melodi menjadi segmen 20-40 beat dengan sliding window 4-8 beat. Teknik ini memungkinkan pencocokan fleksibel dari berbagai bagian lagu yang mungkin diingat pengguna.

Proses windowing disertai normalisasi tempo dan pitch untuk mengurangi variasi humming. Setiap note event dikonversi menjadi representasi numerik yang mempertimbangkan durasi dan urutan nada, memungkinkan sistem membandingkan potongan melodi dengan database. Berikut adalah formula untuk melakukan normalisasi tempo yang dibutuhkan :

$$NP(note) = \frac{(note - \mu)}{\sigma}$$

Dimana μ adalah rata rata dari pitch dan σ adalah standar deviasi dari pitch.

Setelah dilakukannya pemrosesan, maka dapat diperoleh distribusi tone dengan deteksi *pitch:*

1. **Fitur Absolute Tone Based (ATB)** menghitung frekuensi kemunculan setiap nada berdasarkan skala MIDI (0-127). Histogram yang dihasilkan memberikan gambaran distribusi absolut nada dalam data. Hal ini penting untuk menangkap karakteristik statis

melodi dalam sinyal audio. Langkah pertama adalah membuat histogram dengan 128 bin, sesuai dengan rentang nada MIDI dari 0 hingga 127. Kemudian, hitung frekuensi kemunculan masing-masing nada MIDI dalam data. Setelah itu, normalisasi histogram untuk mendapatkan distribusi yang terstandarisasi.

2. **Fitur Relative Tone Based (RTB)** menganalisis perubahan antara nada yang berurutan, menghasilkan histogram dengan nilai dari -127 hingga +127. RTB berguna untuk memahami pola interval melodi, yang lebih relevan dalam mencocokkan humming dengan dataset yang tidak bergantung pada pitch absolut. Dimulai dengan membangun histogram yang memiliki 255 bin dengan rentang nilai dari -127 hingga +127. Selanjutnya, hitung selisih antara nada-nada yang berurutan dalam data. Terakhir, lakukan normalisasi pada histogram yang telah dibuat.
3. **Fitur First Tone Based (FTB)** fokus pada perbedaan antara setiap nada dengan nada pertama, menciptakan histogram yang mencerminkan hubungan relatif terhadap titik referensi awal. Pendekatan ini membantu menangkap struktur relatif nada yang lebih stabil terhadap variasi pitch pengguna. Histogram dibuat dengan 255 bin, juga mencakup rentang nilai dari -127 hingga +127. Kemudian, hitung selisih antara setiap nada dalam data dengan nada pertama. Histogram yang dihasilkan kemudian dinormalisasi untuk menghasilkan distribusi yang seimbang.

Hasil tersebut akan dilakukan normalisasi untuk memastikan bahwa semua nilai dalam histogram berada dalam skala probabilitas. Berikut adalah formula umum dari normalisasi yang digunakan:

$$Hnorm = \frac{H[d]}{\sum_{d}^{127} H[d]}$$

Dimana H adalah Histogram dan d adalah bin dari histogram tersebut.

Kemudian histogram tersebut diubah menjadi sebuah vektor dan dilakukan perhitungan kemiripan menggunakan cosine similarity. Cosine Similarity adalah ukuran untuk menentukan seberapa mirip dua vektor dalam ruang berdimensi tinggi, dengan menghitung sudut cosinus di

antara keduanya. Semakin kecil sudutnya (semakin dekat ke 1 hasilnya), semakin mirip kedua vektor tersebut.

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

2.3 Image Retrieval dengan PCA

PCA (Principal Components Analysis) adalah salah satu teknik statistik yang digunakan untuk mereduksi dimensionalitas data. PCA digunakan untuk mengatasi masalah yang muncul saat bekerja dengan data berdimensi tinggi. Seiring dengan bertambahnya jumlah dimensi, jumlah kemungkinan kombinasi fitur meningkat juga secara eksponensial yang membuat sulit secara komputasi untuk memperoleh sampel data yang representatif. Disertai juga peningkatan waktu dan kompleksitas.

Reduksi dimensionalitas pada PCA hadir yang bertujuan untuk mengurangi jumlah fitur input sambil mempertahankan sebanyak mungkin informasi asli dengan mencari komponen utama (*principal components*) yang merupakan kombinasi linier dari variabel asli. Komponen utama tersebut menangkap varians maksimum dalam data yang kemudian diurutkan berdasarkan variansi terbesar. PCA terdiri dari beberapa langkah dalam penerapannya.

Untuk memudahkan dalam pengimplementasian PCA, maka dilakukan terlebih dahulu pemetaan nama gambar dan gambarnya dengan file .txt atau .json. Kemudian setiap gambar baik untuk dataset dan query akan dilakukan pemrosesan. Pemrosesan tersebut diawali dengan menginterpretasikan informasi dalam gambar menjadi matriks piksel (Image Processing and Loading). Proses yang terjadi di dalamnya adalah mengubah gambar menjadi grayscale untuk mengurangi kompleksitas gambar dan membuat fokus menjadi bagian terang dan gelap gambar. Sehingga setiap gambar direpresentasikan dalam intensitas pixel saja tanpa informasi warna. Proses tersebut dapat dilakukan dengan mengalikan komponen gambar:

$$I(x,y) = 0.2989 \cdot R(x,y) + 0.5870 \cdot G(x,y) + 0.1140 \cdot B(x,y)$$

Hasil pemrosesan tersebut akan diubah besarnya sehingga ukurannya sama untuk seluruh gambar agar perhitungan dapat lebih akurat. Kemudian, vektor grayscale pada gambar diubah menjadi 1 dimensi (flattening) untuk membuat dimensi vektor dapat dilakukan pemrosesan data. Jika gambar memiliki dimensi $M \times N$, maka hasilnya adalah vektor dengan panjang $M \cdot N$:

$$I = [I1, I2, ..., IM \cdot N]$$

Selanjutnya, normalisasi data agar setiap piksel memiliki skala yang seragam yaitu rata-rata 0 dan deviasi standar 1. Proses tersebut dalam dilakukan dengan:

$$\mu_j = \frac{1}{N} \sum_{i=1}^{N} x_{ij}$$

di mana:

- $x$ij: nilai piksel ke-j pada gambar ke-i,
- N: jumlah total gambar dalam dataset.

Lalu dilakukan pengurangan piksel tersebut dengan rata-rata yang sudah dihitung untuk melakukan standarisasi.

$$xij' = xij - \mu j$$

Ketiga, membuat matriks kovarians untuk menangkap hubungan antar piksel. Kovarians mengukur derajat hubungan linier atau variabilitas gabungan antar piksel yang menunjukkan seberapa besar perubahannya dalam kaitannya satu sama lain (ukurannya adalah sejumlah piksel).

$$C = \frac{1}{N} X'^T X'$$

di mana:

- X': matriks data yang sudah distandarisasi.

Dari matriks kovarians yang didapat, dilakukan pemrosesan untuk menemukan *eigenvalues* dan *eigenvectors*. Nilai eigen akan mengindikasikan besarnya variansi yang dijelaskan oleh setiap komponen utama dan vektor eigen memberikan arah dari komponen utama.

$$A X = \lambda X$$

λ adalah nilai eigen dari matriks A dan X adalah vektor eigen. Hal ini didapat dengan dekomposisi matriks, seperti SVD (Singular Value Decomposition). Vektor eigen yang didapat akan menjadi basis baru untuk memproyeksikan data. Dari basis tersebut hanya dipilih sejumlah k komponen berdasarkan kumulatif varians. Terakhir, dilakukan proyeksi data ke ruang komponen utama dengan mengalikan matriks vektor eigen untuk k komponen utama dengan data gambar yang sudah dinormalisasi.

Pencarian kesamaan dapat dilakukan dengan mencari jarak euclidean dari tiap dataset dengan gambar query. Kemudian dilakukan pengurutan kecocokan dari data yang paling tinggi. Representasikan gambar query dalam ruang komponen utama dengan proyeksi yang sama:

$$\mathbf{q} = (\mathbf{q}' - \mu)\mathbf{U}_k$$

Dimana:

- q = Vektor proyeksi dari gambar query ke ruang komponen utama (PCA).
- q′: Gambar query dalam format vektor (setelah grayscale, resize, dan flattening).
- μ: Rata-rata piksel dari dataset (per piksel).
- U_k: Matriks eigenvector dengan k dimensi utama dari PCA.

Kemudian, perhitungan jarak Euclidean antara gambar query dengan semua gambar dalam dataset:

$$d(\mathbf{q}, \mathbf{z}_i) = \sqrt{\sum_{j=1}^{k} (q_j - z_{ij})^2}$$

- $d(q,z_i)$ = Jarak antara gambar query q dan gambar ke-i dalam ruang komponen utama.
- zi = Vektor proyeksi dari gambar ke-i dalam dataset ke ruang komponen utama.
- q_j: Elemen ke-j dari vektor proyeksi query q.
- z_{ij}: Elemen ke-j dari vektor proyeksi gambar ke-i, yaitu z_i.
- k: Jumlah dimensi ruang komponen utama yang dipilih.

Lalu, Urutkan hasil berdasarkan jarak terkecil.

BAB 3
ARSITEKTUR FRONTEND DAN BACKEND

3.1 Sistem Information Retrieval

*Sistem Information Retrieva*l (IR) adalah sistem yang dirancang untuk mencari dan mengidentifikasi informasi yang paling relevan dari koleksi data yang besar berdasarkan input atau query dari pengguna. Sistem ini bertujuan untuk mengolah data secara efisien dan memberikan hasil pencarian yang relevan sesuai dengan kebutuhan pengguna. Dalam konteks tertentu, sistem ini dapat diterapkan untuk menganalisis gambar album dan lagu, serta menemukan hubungan atau tingkat kesamaan (similaritas) antara data query (gambar atau lagu yang diberikan pengguna) dengan data yang ada dalam database. Proses kerja sistem IR terdiri dari 3 tahapan:

1. Ekstraksi Fitur

   Proses pencarian dimulai dengan ekstraksi fitur. Untuk gambar album, fitur yang diekstraksi berupa warna dan tekstur. Sedangkan untuk lagu (audio), fitur yang digunakan mencakup frekuensi nada (pitch) dan melodi.
2. Perhitungan Similaritas

Langkah selanjutnya adalah menghitung tingkat kesamaan (similaritas) antara data query dengan setiap data dalam database. Tingkat kesamaan antara gambar dihitung menggunakan jarak euclidean yang mengukur geometris antara fitur gambar query dan gambar dalam database di ruang fitur. Semakin kecil jarak euclidean maka semakin mirip dua gambar tersebut. Konsep ini juga diterapkan pada audio tetapi menggunakan cosine similarity. Nilai cosine similarity ini berkisar antara -1 (sangat tidak mirip) hingga 1 (sangat mirip).

3. Proses Pencarian

   Sistem kemudian akan mengurutkan hasil berdasarkan tingkat similaritas tertinggi ke terendah. Hasil tersebutlah yang akan disajikan pada website.

3. 2 Arsitektur FrontEnd

Front-end aplikasi ini dirancang untuk memberikan antarmuka pengguna yang sederhana, interaktif, dan fungsional. Proses rendering halaman dilakukan melalui fungsi `render_template` yang disediakan oleh Flask. Dengan pendekatan ini, file HTML di render di sisi server, kemudian dikirimkan kepada pengguna untuk ditampilkan di browser mereka. Pendekatan ini memungkinkan penggabungan dinamis antara elemen-elemen visual pada front-end dengan data yang disediakan oleh back-end.

Untuk mendukung estetika dan pengalaman pengguna yang responsif, aplikasi memanfaatkan Tailwind CSS melalui Content Delivery Network (CDN). Tailwind CSS adalah framework CSS berbasis utility-first yang menawarkan berbagai kelas siap pakai, sehingga pengembang dapat dengan mudah mengatur layout, warna, ukuran, dan elemen-elemen desain lainnya tanpa harus menulis CSS dari awal. Penggunaan CDN juga mempercepat proses pengembangan karena tidak memerlukan instalasi atau konfigurasi tambahan.

Selain styling, aplikasi menggunakan MIDI.js melalui CDN untuk menangani pemutaran file MIDI. MIDI.js adalah library JavaScript yang memungkinkan file MIDI diproses dan dimainkan langsung di browser, tanpa memerlukan perangkat lunak tambahan. Hal ini memberikan fleksibilitas bagi pengguna untuk mendengarkan hasil query dan file MIDI lainnya secara langsung melalui antarmuka aplikasi.

Untuk menyimpan dan mengelola data sementara, seperti file yang diunggah oleh pengguna dan nama-nama dataset yang tersedia, aplikasi menggunakan file konfigurasi `config.json`. File ini berfungsi sebagai penghubung antara front-end dan back-end, sehingga data yang diperlukan dapat diakses dengan mudah dan diperbarui secara dinamis sesuai aktivitas pengguna.

Fitur utama yang dihadirkan pada front-end mencakup:

1. Menampilkan album: Aplikasi membuat dan menampilkan daftar album yang tersedia dalam dataset, lengkap dengan informasi terkait.
2. Memutar lagu: Pengguna dapat memutar lagu dalam format MIDI menggunakan MIDI.js, yang terintegrasi langsung di antarmuka aplikasi.
3. Menampilkan hasil query: Setelah pengguna melakukan query, hasil pencocokan akan ditampilkan secara visual, termasuk informasi terkait lagu yang ditemukan.
4. Mengunggah dataset: Pengguna dapat menambahkan dataset baru untuk memperkaya basis data aplikasi.
5. Mengunggah query: Front-end menyediakan form untuk menerima input file query dari pengguna, yang kemudian dikirim ke back-end untuk diproses.

Integrasi antara front-end dan back-end dilakukan melalui **Flask API**, yang memungkinkan komunikasi data secara real-time. API ini mengelola semua request dari front-end, seperti pengunggahan file, pemrosesan query, dan pengambilan data untuk ditampilkan. Dengan struktur ini, aplikasi memastikan alur kerja yang efisien dan responsif untuk memenuhi kebutuhan pengguna.

3. 3 Arsitektur BackEnd

Back-end aplikasi ini dibangun menggunakan Flask, sebuah framework Python yang ringan dan fleksibel. Flask dipilih karena kemampuannya untuk menyediakan endpoint yang cepat dan mudah untuk dikembangkan, serta dukungan kuat untuk pengintegrasian dengan berbagai pustaka Python.

Aplikasi ini menggunakan Python sebagai bahasa pemrograman utama. Python dipilih karena sifatnya yang mudah dipelajari, memiliki sintaks yang sederhana, dan ekosistem library

yang luas, sehingga sangat mendukung pengembangan aplikasi berbasis web. Selain itu, Python sangat cocok untuk kebutuhan pemrosesan data dan algoritma karena memiliki banyak pustaka yang dirancang untuk tugas-tugas tersebut.

Tidak menggunakan basis data, arsitektur back-end ini dirancang sebagai controller yang menangani permintaan dari front-end dan menjalankan berbagai tugas, termasuk:

1. Render template: Back-end menggunakan fungsi bawaan Flask untuk merender file HTML dengan render_template.
2. Pengunggahan data: Endpoint Flask dirancang untuk menerima data dari pengguna, seperti file yang diunggah. File ini kemudian diproses atau disimpan secara sementara untuk kebutuhan algoritma.
3. Mengembalikan data ke front-end: Back-end menyediakan API yang mengembalikan hasil query atau data lainnya kepada front-end dalam format yang mudah diakses, seperti JSON atau melalui template HTML.
4. Menjalankan algoritma: Menggunakan pustaka seperti librosa dan numpy, back-end bertugas untuk memproses data audio, seperti melakukan normalisasi, ekstraksi fitur, atau operasi matematis lainnya. Library tambahan seperti mido digunakan untuk mengelola data MIDI, sementara scipy mendukung operasi matematis yang lebih kompleks.

Back-end terintegrasi dengan front-end sepenuhnya melalui Flask. Semua permintaan yang dilakukan pengguna melalui antarmuka diarahkan ke Flask API. Data hasil pemrosesan di back-end dikembalikan dalam bentuk yang dapat langsung dimanfaatkan oleh antarmuka front-end, seperti JSON atau elemen visual yang sudah dirender.

Untuk menunjang berbagai fungsi utama, beberapa pustaka digunakan dalam pengembangan back-end:

- Librosa: Digunakan untuk pemrosesan audio, termasuk analisis fitur dan manipulasi data suara.
- Numpy dan Scipy: Membantu dalam komputasi numerik, seperti operasi matriks dan pengolahan data audio.

- Mido: Digunakan untuk memproses file MIDI, termasuk pembacaan, manipulasi, dan pembuatan kembali file MIDI.
- Pillow: Dimanfaatkan untuk menangani data gambar jika diperlukan.
- Python-dotenv: Untuk mengelola konfigurasi sensitif seperti variabel lingkungan secara aman.

Dengan pendekatan ini, back-end mampu mengelola proses-proses inti aplikasi secara efisien, termasuk pemrosesan data, integrasi dengan algoritma khusus, dan pengelolaan file. Tanpa ketergantungan pada basis data, sistem ini lebih ringan namun tetap cukup fleksibel untuk menjalankan fungsionalitas yang dibutuhkan aplikasi.

BAB 4
IMPLEMENTASI DAN UJI COBA

4.1 Implementasi Fungsi Program

4.1.1 Album Picture Finder (imageProcessing.py)

Fungsi	Parameter Input	Parameter Output	Penjelasan
process_image	1. image_path: path dari 2. target_size: ukuran piksel target dari file	1. flattened_image: vektor hasil dalam bentuk 1 dimensi	Pemrosesan setiap gambar dataset dan query menjadi grayscale, resize, dan menjadi vektor 1 dimensi
pca	1. X: numpy array hasil pemrosesan pada fungsi process_image 2. variance _threshold: banyaknya variansi dari data yang ingin	1. Mean_pixel: hasil standarisasi data di sekitar nilai 0 2. Uk: matriks eigenvector dengan n-dimensi (n =	Pemrosesan gambar dengan melakukan dekomposisi SVD. Kemudian diambil n jumlah komponen utama teratas dari

	dipertahankan	jumlah komponen utama). 3. X_projected: vektor proyeksi dari gambar dataset dan query ke ruang komponen utama(PCA)	matriks eigenvector dan dilakukan proyeksi pada komponen tersebut
projection_query_image	1. query_image: hasil dari fungsi process_image 2. mean_pixel: hasil rata-rata dari fungsi load_pca_result 3. Uk: matriks transpose dari SVD	1. query_projected: hasil proyeksi gambar query	Dilakukan memproyeksikan gambar query ke ruang komponen utama
save_pca_results	1. mean_pixel: hasil rata-rata dari fungsi load_pca_result 2. Uk: matriks transpose dari SVD 3. X_projected: hasil proyeksi gambar dataset	-	Prosedur untuk menyimpan hasil PCA.
load_pca_results	1. prefix: nama awal dari hasil pca	1. mean_pixel: hasil rata-rata dari fungsi load_pca_result 2. Uk: matriks transpose dari SVD 3. X_projected: hasil proyeksi gambar	Memuat hasil PCA yang sudah dihitung.

		dataset	
read_json_and_process	1. album_data: hasil load file json	-	Prosedur untuk membaca file JSON, melakukan proses dan membuat PCA
imageRetrieval	1. json_path: path letak file map.json 2. pca_result_path: path letak file hasil PCA 3. query_path: path file gambar query	1. ordered_results: hasil pengurutan jawak euclidean dan informasinya pada json juga 2. processing_time: waktu pemrosesan	Fungsi untuk melakukan seluruh langkah pada image retrieval

4.1.2 Music Information Retrieval (audioProcessing.py)

Fungsi	Parameter Input	Parameter Output	Penjelasan
normalize_pitch	pitches (array), epsilon(float, optional)	res (array ter-normalisasi)	Menormalisasi array pitch agar memiliki distribusi dengan mean 0 dan standar deviasi 1.
filter_pitches	pitches (array), threshold (int)	Array pitches yang sudah difilter	Menghapus pitch dengan nilai di bawah threshold.
filter_sparse_notes	notes (array)	Notes yang sudah difilter	Menghapus catatan MIDI yang jarang muncul.
extract_pitches	y (signal audio), sr(sampling rate)	detected_pitches, peaks	Mengekstraksi pitch dari signal audio menggunakan analisis puncak pada onset environment.

process_wav	file_path, sr (sampling rate), window_size	windows (array pitch yang sudah diproses)	Memproses file WAV untuk mengekstraksi pitch, menormalisasi, dan membuat window.
process_midi	file_path, window_size	windows (array pitch yang sudah diproses)	Memproses file MIDI untuk mengekstraksi dan menormalisasi notes.
compute_normalized_histogram	data (array), bins (int), range_ (tuple)	Histogram ter-normalisasi	Menghitung histogram ter-normalisasi dari data input.
compute_atb	notes (array)	Histogram ATB	Menghitung distribusi ATB (After-To-Before).
compute_rtb	notes (array)	Histogram RTB	Menghitung distribusi RTB (Relative-To-Base).
compute_ftb	notes (array)	Histogram FTB	Menghitung distribusi FTB (Feature-To-Base).
extract_tone_distribution	notes (array)	Dictionary distribusi ATB, RTB, FTB	Mengekstraksi fitur distribusi pitch dari array notes, seperti ATB, RTB, dan FTB.
cosine_similarity	vec1 (array), vec2 (array)	Nilai cosine similarity (float)	Menghitung cosine similarity antara dua vektor.
compute_weighted_similarity	midi_distribution, wav_distribution, weights	Total similarity	Menggabungkan distribusi ATB, RTB, FTB untuk menghitung similarity berbasis bobot.
find_similarities	query_file, json_file_path, base_dir, weights, window_size	Array hasil similarity antara file MIDI dan WAV	Menggabungkan semua distribusi untuk menemukan kesamaan antara query dan dataset.

4.2 Struktur *Website*

4.2.1 Halaman Utama (Home)

Halaman ini berisi seluruh informasi terkait judul lagu, lagu, dan gambar album. Tersedia juga button untuk melakukan upload dataset dan query.

4.2.2 Halaman Upload Dataset

Halaman tersebut merupakan lokasi untuk melakukan input dataset musik, album, dan mapper. Input (masukan) tersebut dapat berupa multiple image, folder, .zip, .rar, dan lainnya.

4.2.3 Halaman Humming Result

Halaman yang berisi untuk menunjukkan hasil audio *processing*. Akan ditampilkan waktu proses dan juga similaritas audio yang terurut.

4.2.4 Halaman Album Result

Halaman yang berisi untuk menunjukkan hasil *image processing* . Akan tertampil waktu proses dan juga similaritas gambar.

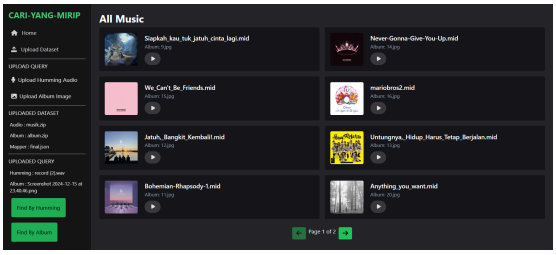
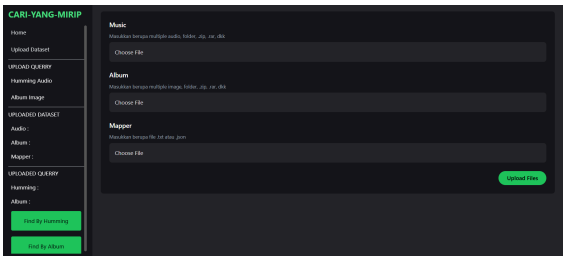
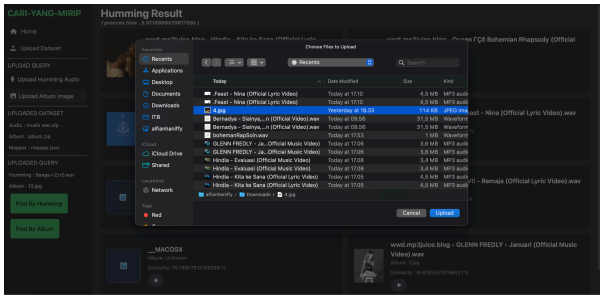
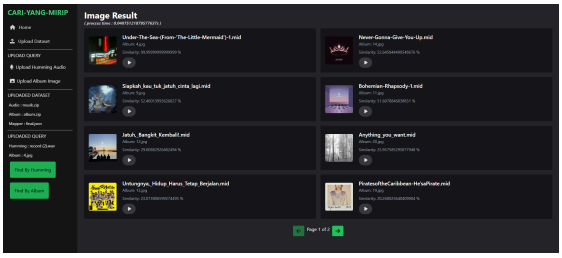
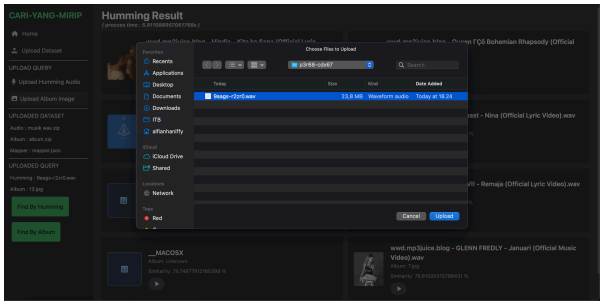
4.3 Tata Cara Penggunaan Program

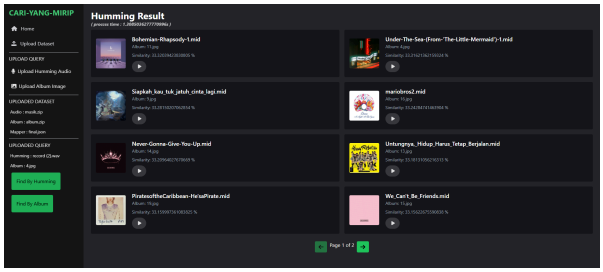
Berikut merupakan tata cara penggunaan *website* program:

1. Navigasikan *website* menggunakan tombol pada bagian kiri berupa home, upload dataset, upload humming audio, upload album audio, find by humming, dan find by album.
2. Untuk melakukan pengunggahan (upload) dataset, tekan tombol ‘Upload Dataset’ kemudian pilih dataset musik, album, dan mapper pada form yang bersesuaian lalu klik ‘Upload Files’
3. Untuk melakukan pengunggahan query, maka pilih pada navigasi yang di sisi kiri sesuai jenis query yang ingin dicek (audio ataupun image).
4. Pilih file yang sesuai kemudian klik tombol ‘Find by Humming’ jika ingin memproses audio dan ‘Find by Album’ jika ingin memproses gambar album.

4.4 Hasil Pengujian

Pengujian yang Dilakukan	Hasil Pengujian

Halaman utama *website* disertai beberapa button untuk mengarahkan pada laman upload dataset, upload query, humming result, dan album result.	 Gambar 4.4.1
Dilakukan upload dataset	 Gambar 4.4.2
Dilakukan upload image query	 Gambar 4.4.3
Pengujian dan diperoleh hasil persentase kemiripan dan waktu proses *image*	 Gambar 4.4.4
Dilakukan upload audio query	 Gambar 4.4.5

Pengujian dan diperoleh hasil persentase kemiripan dan waktu proses audio	 Gambar 4.4.5

4.4 Analisis Solusi Algoritma

Setelah dilakukan uji coba pada perhitungan yang dibuat, algoritma image processing masih menunjukkan kekurangan dalam hal presisi, khususnya dalam menentukan jarak Euclidean terdekat antara query dan dataset yang tersedia. Salah satu pengujian yang dilakukan adalah dengan menggunakan query berupa hasil *resize* dari gambar yang terdapat di dataset. Namun, hasil yang diperoleh tidak sesuai dengan harapan. Hal ini kemungkinan disebabkan oleh keterbatasan PCA sebagai teknik reduksi dimensi. PCA memperlakukan data gambar sebagai vektor datar (*flattened vector*), sehingga informasi penting mengenai hubungan spasial antar piksel dalam gambar menjadi hilang. Akibatnya, PCA tidak mampu memahami posisi relatif piksel dalam gambar asli, yang menyebabkan distribusi varians dari query di ruang PCA menjadi sangat berbeda dengan data di dataset.

Selain itu, pada algoritma *audio processing*, hasil penghitungan *similarity* juga seringkali kurang akurat. Hal ini dapat disebabkan oleh beberapa faktor, seperti keterbatasan dalam analisis data, kualitas input audio yang tidak memadai (misalnya, terdapat noise atau kualitas rekaman rendah), serta kompleksitas sinyal audio itu sendiri. Sinyal audio memiliki struktur temporal dan frekuensi yang kompleks, yang tidak selalu dapat ditangkap sepenuhnya oleh algoritma yang digunakan. Faktor-faktor ini memengaruhi performa algoritma dalam menghitung kesamaan antara query audio dan dataset.

BAB 5
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dalam sistem temu balik gambar, Principal Component Analysis dapat diterapkan untuk melakukan pencarian gambar dengan mereduksi dimensi data dengan tetap mempertahankan sebanyak mungkin informasi yang ada. PCA akan mengubah data berdimensi tinggi menjadi beberapa dimensi yang lebih kecil (principal component) tanpa kehilangan esensi atau pola utama dalam data tersebut. Pada program yang diterapkan, PCA dilakukan dengan dekomposisi matriks cara SVD (Singular Value Decomposition) kemudian dapat diperoleh eigenvector dan proyeksi data, baik gambar query maupun database lalu dilakukan perhitungan jarak euclidean. Untuk pemrosesan audio, dilakukan dengan memproses pitch yang dinormalisasi kemudian disamakan dengan file yang tersedia di dataset dengan menggunakan cosine similarity.

Lewat pembelajaran pada mata kuliah aljabar linier dan geometri, pemahaman akan dekomposisi matriks dapat diterapkan dan kemudian menghasilkan pemrosesan ini kemudian dituangkan pada bentuk website juga.

5.2 Saran

Penambahan library pada python untuk pengoptimalan waktu dan proses pada image dan audio processing. Selain itu, adanya perbaikan pada algoritma image processing agar lebih presisi dalam membandingkan gambar query dengan gambar dataset. Contohnya, bila gambar query adalah hasil potongan kecil atau hasil resize dari salah satu dataset.

5.3 Komentar

Tugas ini memberi kami kesempatan untuk mencari tahu lebih banyak tentang bagaimana audio dan image diproses dalam bentuk numerik. Hal ini membuat kami untuk lebih mendalami mengenai penerapan pada salah satu materi pada aljabar linear dan geometri, yaitu dekomposisi matriks.

5.4 Refleksi

Tugas besar ini berhasil memberikan kesempatan bagi kami untuk bekerja sama di bawah tekanan karena harus mengerjakan tugas-tugas lain di waktu yang bersamaan. Melalui tugas besar ini kami dapat meningkatkan performa dalam mengimplementasikan materi perhitungan seperti dekomposisi dalam bentuk program/kode.

LAMPIRAN

1. **Referensi**

Frost, J. (2024). *Principal Component Analysis Guide & Example*. Statistics By Jim. Retrieved December 8, 2024, from https://statisticsbyjim.com/basics/principal-component-analysis/

Homepage Rinaldi Munir. (n.d.). Informatika. Retrieved December 3, 2024, from https://informatika.stei.itb.ac.id/~rinaldi.munir/

Jain, S. (2024, September 10). *Principal Component Analysis(PCA)*. GeeksforGeeks. Retrieved December 8, 2024, from https://www.geeksforgeeks.org/principal-component-analysis-pca

Jain, S. (2024, September 10). *Principal Component Analysis(PCA)*. GeeksforGeeks. Retrieved December 1, 2024, from https://www.geeksforgeeks.org/principal-component-analysis-pca/

Mallick, S. (2018, January 7). *Principal Component Analysis*. LearnOpenCV. Retrieved December 1, 2024, from https://learnopencv.com/principal-component-analysis/

2. **Tautan Repository**

https://github.com/AlfianHanifFY/Algeo02-23073

3. **Tautan Video**

https://www.youtube.com/watch?v=QBXQrlCRqnw