Sistem Klasifikasi Genre Musik menggunakan CNN dan MFCC

Pengantar Pemrosesan Data Multimedia

Dosen Pengampu:

Dr. Anak Agung Istri Ngurah Eka Karyawati, S.Si., M.Eng.

Nama Anggota Kelompok B2 :

Ni Luh Putu Happy Nirmala (2208561015)

Kendrick Raphael Ticoalu (2208561097)

I Gusti Bagus Putrawan (2208561133)

I Made Sastra Wiguna (2208561121)

PROGRAM STUDI INFORMATIKA

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

UNIVERSITAS UDAYANA

2024

DAFTAR ISI

BAB I

PENDAHULUAN

1.1 Rumusan Masalah

1.1.1 Bagaimana tingkat akurasi dan performa dari CNN dalam melakukan klasifikasi genre musik?

1.1.2 Bagaimana hasil dari pemilihan model terbaik setelah melakukan *hyper-parameter tuning* menggunakan *k-fold cross validation*?

1.2 Tujuan

Tujuan dari penelitian ini adalah untuk mengembangkan sistem klasifikasi genre musik menggunakan *Convolutional Neural Network* (CNN) dan *Mel-Frequency Cepstral Coefficients* (MFCC) sebagai fitur utama. Penelitian ini bertujuan untuk mengevaluasi efektivitas MFCC dalam proses klasifikasi, mengimplementasikan model CNN, serta mengevaluasi kinerjanya dengan metrik seperti akurasi, presisi, recall, dan F1-score. Selain itu, penelitian ini juga bertujuan untuk mengoptimalkan model melalui eksplorasi arsitektur CNN. Dengan demikian, penelitian ini diharapkan dapat memberikan kontribusi signifikan dalam bidang penelitian musik dan kecerdasan buatan, serta menghasilkan aplikasi praktis yang mampu mengklasifikasikan genre musik secara *real-time.*

1.3 Manfaat

Manfaat dari penelitian ini mencakup peningkatan akurasi dan efisiensi dalam klasifikasi genre musik, yang dapat membantu dalam berbagai aplikasi seperti rekomendasi musik, analisis tren musik, dan pengarsipan digital. Dengan menggunakan *Convolutional Neural Network* (CNN) dan *Mel-Frequency Cepstral Coefficients* (MFCC), penelitian ini dapat memperkenalkan pendekatan baru yang lebih andal dan cepat dibandingkan metode konvensional. Selain itu, hasil penelitian ini juga dapat dimanfaatkan oleh industri musik untuk mengembangkan layanan berbasis kecerdasan buatan yang lebih canggih dan personal, serta oleh komunitas akademik untuk mendorong penelitian lanjutan di bidang pengolahan sinyal audio dan *deep learning*.

1.4 Batasan

Adapun beberapa batasan adalah sebagai berikut:

1.4.1 Data yang digunakan merupakan klasifikasi genre musik oleh GTZAN Dataset yang diperoleh dari *website* penyedia layanan *dataset*, yaitu *kaggle.*

1.4.2 Metode klasifikasi genre musik yang digunakan yaitu CNN dan MFCC.

1.4.3 Genre musik dibagi menjadi sepuluh kategori.

1.4.4 Sistem yang dibangun diimplementasikan menggunakan bahasa pemrograman Python dan beberapa *library*.

1.5 Asumsi

Adapun beberapa asumsi adalah sebagai berikut:

1.5.1 Data yang digunakan dalam penelitian ini akurat dan representatif terhadap jenis-jenis genre musik yang ada. Serta memiliki kualitas yang memadai untuk dianalisis karena data tersebut tidak mengandung terlalu banyak *noise* (data yang tidak relevan atau tidak bermakna) dan memiliki struktur yang konsisten.

1.5.2 Diasumsikan bahwa distribusi genre dalam *dataset* seimbang atau memiliki cukup data untuk masing-masing kategori.

1.5.3 Diasumsikan bahwa langkah-langkah *preprocessing* dapat meningkatkan akurasi model dengan cara mempersiapkan data secara optimal sebelum dimasukkan ke dalam algoritma *deep learning*.

1.5.4 Diasumsikan bahwa model *deep learning* CNN dan *hyperparameter tuning* yang dilakukan akan menghasilkan model terbaik untuk klasifikasi genre musik.

1.5.5 Diasumsikan bahwa model *deep learning* yang dirancang dapat diimplementasikan dengan baik ke dalam *platform* web tanpa mengurangi performa atau akurasi klasifikasi genre musik

BAB II

TINJAUAN PUSTAKA

2.1. Tinjauan Teori

2.1.1 Analisis Genre

Analisis adalah proses memeriksa sesuatu secara mendalam untuk memahami atau menjelaskan komponennya dan bagaimana mereka saling berhubungan. Genre musik merupakan pengelompokkan musik berdasarkan ritme, harmoni, dan berbagai unsur musik lainnya. Genre musik adalah cara paling umum untuk mengatur database musik digital. Upaya pengenalan genre musik dilakukan secara mendalam oleh komunitas Music Information Retrieval (MIR) sejak tahun 2002 (Ramadhan & Widiartha, 2021). Analisis genre terdapat dua tahap yakni dan ekstraksi fitur dan klasifikasi genre. Ekstraksi fitur adalah metode penarikan fitur-fitur tertentu dari sebuah karya musik yang akan dijadikan sebagai pertimbangan klasifikasi genre. Klasifikasi genre adalah tindakan pemeriksaan fitur-fitur yang telah diekstraksi untuk memprediksi kelas genre sebuah karya musik dengan model yang terlatih atas sebuah dataset. Sebelum dilakukan klasifikasi terdapat proses ekstraksi fitur (Soerkarta dkk, 2023).

2.1.2 *Mel Frequency Cepstral Coefficients* (MFCC)

MFCC adalah fitur yang bisa dianalisis untuk menentukan kelas genre sebuah karya musik. Fitur ini merupakan representasi frekuensi dari sinyal audio yang meniru cara telinga manusia mendengar suara. MFCC diekstraksi dari spektrum frekuensi audio dengan menggunakan filter bank mel dan transformasi cepstral. MFCC telah terbukti efektif dalam berbagai tugas pengenalan suara, termasuk klasifikasi genre musik (Bisht dkk, 2022).

Pertama-tama sampel audio dibagi menjadi frame-frame untuk mempertahankan informasi waktu dari suara. Durasi frame dapat dihitung menggunakan rumus berikut:

$$S(m,k) = \sum_{n=0}^{N-1} x(nmH).w(n).e^{-i2\pi nk/N}$$

dimana \(d_f \) adalah durasi frame, \(sr \) adalah laju sampling, dan \(K \) mewakili jumlah total sampel yang ada dalam sinyal audio.

Proses penerapan fungsi jendela pada masing-masing frame sebelum melakukan Fourier Transform disebut windowing. Fungsi jendela ini menghilangkan sampel di kedua ujung frame, yang dapat menyebabkan hilangnya informasi sinyal. Untuk menghindari masalah ini, digunakan frame yang saling tumpang tindih untuk mempertahankan informasi. Setelah framing dan windowing, dicarikan *Short Time Fourier Transform* (STFT) untuk mendapatkan spektrum daya dari audio. STFT dari sinyal dapat direpresentasikan dengan persamaan berikut:

$$S(m,k) = \sum_{n=0}^{N-1} x(nmH).w(n).e^{-i2\pi nk/N}$$

Setelah STFT, untuk membuat rentang frekuensi linear digunakan skala Mel-liner. Konversi skala Mel dapat dilakukan menggunakan rumus:

$$n = 2595.log\left(1+\frac{f}{500}\right)$$

dimana m adalah frekuensi dalam skala Mel dan f mewakili frekuensi fisik dalam Hz.

Akhirnya, *Discrete Cosine Transform* (DCT) diterapkan untuk mendapatkan MFCC. DCT dapat didefinisikan secara matematis sebagai:

$$Z(n,k) = \sum_{m=0}^{M-1} X(n,m)cos\left[\frac{\pi}{m}\left(m+\frac{1}{2}\right)k\right], here\ k = 0,1 \ldots . M-1$$

dimana n adalah jumlah frame dan m adalah jumlah frekuensi Mel. DCT dari Mel-spectrogram menghasilkan koefisien bernilai nyata yang memberikan informasi tentang timbre dari sinyal audio (Bisht dkk, 2022).

2.1.3 CNN

CNN adalah jenis jaringan saraf tiruan yang sangat efektif untuk tugas-tugas pengenalan pola, seperti klasifikasi gambar dan pengenalan suara. CNN menggunakan arsitektur yang terdiri dari lapisan konvolusi, lapisan *pooling*, dan lapisan *fully-connected*. Lapisan konvolusi mengekstrak fitur dari data input,

lapisan pooling mengurangi dimensi data, dan lapisan *fully-connected* mengklasifikasikan data.CNN menggunakan arsitektur yang terdiri dari tiga lapisan utama yaitu :

A. Lapisan konvolusi

Mengekstrak fitur dari data input dengan menggeser filter konvolusi di atas data. Filter konvolusi berisi bobot yang dipelajari selama proses pelatihan model. Hasil dari proses ini adalah peta fitur yang menunjukkan respons filter konvolusi terhadap data input.

B. Lapisan *pooling*

Digunakan untuk mengurangi dimensi data, membantu mencegah overfitting dan meningkatkan efisiensi komputasi. Ada dua jenis umum lapisan pooling: *max pooling* dan *average pooling*. *Max pooling* mengambil nilai maksimum dari setiap jendela di peta fitur, sedangkan *average pooling* mengambil nilai rata-rata dari setiap jendela.

C. Lapisan *fully-connected*

Menghubungkan semua neuron di lapisan sebelumnya dengan semua neuron di lapisan berikutnya. Lapisan ini digunakan untuk mengklasifikasikan data input. Setiap neuron di lapisan *fully-connected* memiliki bobot yang dipelajari selama proses pelatihan model.

2.1.5 K-Fold Cross Validation

K-fold cross validation adalah teknik untuk mengevaluasi performa model *machine learning* dengan membagi data menjadi k subset atau *fold*. Setiap *fold* digunakan sebagai data uji satu kali, sementara k-1 *fold* lainnya digunakan sebagai data latih. Proses ini diulang k kali sehingga setiap *fold* berfungsi sebagai data uji sekali. Teknik ini membantu dalam mengurangi *overfitting* dan memberikan estimasi performa model yang lebih akurat.

2.1.6 Hyperparameter Tuning

Hyperparameter tuning adalah proses untuk menemukan kombinasi nilai optimal dari parameter-parameter yang tidak dipelajari secara otomatis oleh model

machine learning selama proses pelatihan. Parameter-parameter ini disebut *hyperparameter*, dan mereka memengaruhi bagaimana model belajar dari data dan kemudian membuat prediksi. Adapun beberapa tahapan yang dilakukan dalam melakukan *hyperparameter tuning,* antara lain:

A. *Grid Search*

Metode ini melibatkan pencarian secara sistematis melalui berbagai kombinasi nilai parameter. Setiap kombinasi diuji pada model, dan performa dari setiap kombinasi dievaluasi menggunakan teknik validasi silang atau data validasi terpisah.

B. *Random Search*

Metode ini melakukan pencarian *hyperparameter tuning* yang digunakan untuk menemukan kombinasi nilai *hyperparameter* yang optimal. Metode ini melibatkan pemilihan nilai *hyperparameter* secara acak dari rentang yang ditentukan untuk setiap *hyperparameter*.

C. Bayesian Optimization

Metode ini menggunakan model probabilistik untuk memperkirakan performa model untuk nilai hyperparameter yang berbeda. Bayesian Optimization adalah metode yang lebih canggih dan efisien daripada Grid Search dan Random Search, tetapi membutuhkan lebih banyak pengetahuan tentang model dan data.

2.1.7 Framing dan Windowing

Proses framing adalah membagi sinyal audio menjadi segmen-segmen kecil yang disebut frame, untuk mempertahankan informasi waktu dari suara. Hal ini penting karena sinyal audio adalah sinyal waktu yang kontinu, dan framing membantu dalam analisis sinyal yang lebih rinci.

Setelah framing, langkah selanjutnya adalah menerapkan fungsi jendela pada masing-masing frame, yang dikenal sebagai windowing. Fungsi jendela digunakan untuk mengurangi efek spektral yang disebabkan oleh pemotongan sinyal pada titik tertentu. Fungsi jendela yang umum digunakan adalah jendela Hamming, Hanning,

dan Blackman. Fungsi jendela menghaluskan ujung-ujung frame untuk mengurangi diskontinuitas sinyal, yang dapat menyebabkan artefak dalam analisis spektrum.

Persamaan matematis untuk fungsi jendela Hamming adalah:

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

dimana w(n)w(n)w(n) adalah nilai fungsi jendela pada titik nnn, dan NNN adalah panjang frame.

Setelah proses windowing, dilakukan Short Time Fourier Transform (STFT) untuk mendapatkan spektrum daya dari audio. STFT dari sinyal dapat direpresentasikan dengan persamaan berikut:

$$X(m, \omega) = \sum_{n=-\infty}^{\infty} x(n) w(n - m) e^{-j\omega n}$$

dimana X(m,ω)X(m, \omega)X(m,ω) adalah hasil STFT, x(n)x(n)x(n) adalah sinyal waktu, w(n)w(n)w(n) adalah fungsi jendela, dan ω\omegaω adalah frekuensi sudut.

Setelah STFT, untuk membuat rentang frekuensi linear digunakan skala Mel-liner. Konversi skala Mel dapat dilakukan menggunakan rumus:

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

dimana mmm adalah frekuensi dalam skala Mel dan fff mewakili frekuensi fisik dalam Hz.

Akhirnya, Discrete Cosine Transform (DCT) diterapkan untuk mendapatkan MFCC. DCT dapat didefinisikan secara matematis sebagai:

$$C(k) = \sum_{n=0}^{N-1} x(n) \cos\left[\frac{\pi k}{N}\left(n + \frac{1}{2}\right)\right]$$

dimana nnn adalah jumlah frame dan kkk adalah jumlah koefisien Mel. DCT dari Mel-spectrogram menghasilkan koefisien bernilai nyata yang memberikan informasi tentang timbre dari sinyal audio (Bisht dkk, 2022).

2.2 Tinjauan Empiris

Berikut merupakan beberapa penelitian sebelumnya terkait dengan klasifikasi genre musik yang menggunakan metode *Convolutional Neural Network* (CNN) dan *Mel-Frequency Cepstral Coefficients* (MFCC).

2.2.1 *Classification of Pop And RnB (Rhythm And Blues) Songs With MFCC Feature Extraction And K-NN Classifier* (Ramadhan dan Widiartha, 2021)

Penelitian ini melakukan klasifikasi musik untuk menentukan apakah kelas atau genre musik adalah pop atau RnB (*Rhythm and Blues*) dengan menggunakan MFCC sebagai metode ekstraksi fitur dan K-NN sebagai metode klasifikasi. Dalam penelitian ini, data audio yang akan diklasifikasikan adalah data sekunder yang diperoleh dari dataset GTZAN. Dua genre yang diambil adalah RnB dan pop dengan masing-masing 100 lagu per genre. Beberapa langkah diperlukan untuk ekstraksi fitur MFCC, yaitu *Frame Blocking, Windowing, Fast Fourier Transform (FFT), Mel Frequency Wrapping, dan Cepstrum.* Data baru yang diperoleh dari ekstraksi fitur MFCC akan digunakan sebagai data untuk klasifikasi selanjutnya. Beberapa langkah dalam klasifikasi dengan K-NN adalah *Input Dataset, Preprocessing, Euclidean Distance Calculation, Sorting Closest distance,* dan *Class Determination.* Skenario pengujian dilakukan untuk menentukan akurasi terbaik yang dihasilkan dari beberapa nilai K berbeda yang akan diuji dengan metode K-NN terhadap klasifikasi genre musik. Nilai atau parameter K yang akan diuji adalah 3, 7, 11, 21, 31, 41, 51, dan 61. Proporsi data pelatihan dan data uji dalam penelitian ini adalah 80:20. Berdasarkan hasil penelitian, metode ekstraksi fitur MFCC dan K-NN *Classifier* menghasilkan akurasi tertinggi sebesar 77,5% dengan nilai k = 31.

2.2.2 MUSIC RECOMMENDATION SYSTEM BASED ON COSINE SIMILARITY AND SUPERVISED GENRE CLASSIFICATION (Alyza, dkk., 2023)

Penelitian ini menentukan teknik pembelajaran mesin yang paling efektif untuk memprediksi genre lagu dengan akurat menggunakan algoritma *K-Nearest Neighbors* (K-NN) dan *Support Vector Machine* (SVM) dengan dimensi dipertimbangkan dan tanpa menggunakan *Principal Component Analysis* (PCA)

untuk pengurangan dimensi. MFCC digunakan untuk mengumpulkan data dari dataset. Adapun tahapan metode yang dilakukan adalah *data collection, preprocessing* yang terdiri dari *Fourier Transform, Spectrogram, dan Mel Spetrogram*, dilanjutkan dengan *feature extraction* yang terdiri dari *Harmonics and Perceptual,* dan *Spectral Centroid and Roll off*. Kemudian dilanjutkan dengan algoritma *K-Nearest Neighbors* (K-NN) dan *Support Vector Machine* (SVM) serta *Confusion Matrix.* Penelitian ini mengungkapkan bahwa *K-Nearest Neighbors* dan *Support Vector Machine* menawarkan hasil yang lebih presisi tanpa mengurangi dimensi dibandingkan dengan hasil PCA. Akurasi menggunakan metode PCA adalah 58% dan memiliki potensi untuk menurun. Akurasi *K-Nearest Neighbors* adalah 64,9%, dan akurasi *Support Vector Machine* (SVM) adalah 77%.

2.2.3 Hyperparameter Optimization of CNN Classifier for Music Genre Classification. (Soerkarta, dkk., 2021)

Penelitian ini mengevaluasi hiperparameter dalam proses klasifikasi genre musik menggunakan CNN pada dataset GTZAN dengan data berdurasi 30 detik yang dioptimalkan menggunakan ekstraksi fitur MFCC. Model yang dibentuk dengan waktu 3 (tiga) detik mengklasifikasikan genre musik dalam 3 detik pertama musik. Model ini memiliki potensi kesalahan yang tinggi karena 3 detik pertama musik sangat bervariasi dan tidak dapat digunakan sebagai patokan dalam menentukan genre musik. Penelitian ini melakukan pengaturan hiperparameter pada variabel ukuran batch, epoch, dan pembagian dataset dengan berbagai skenario. Hasil akurasi tertinggi diperoleh sebesar 72% dengan pembagian data 85%:15%, ukuran batch 32, dan 500 epoch.

BAB III

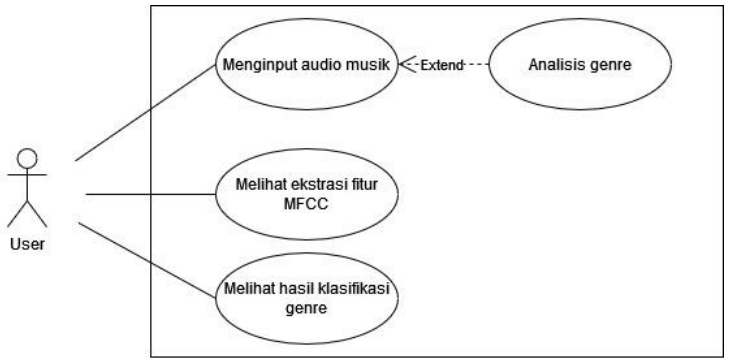
ANALISIS DAN DESAIN

3.1 Data dan Pengumpulan Data

Data yang digunakan pada penelitian ini merupakan data sekunder yaitu data yang memang sudah ada di lapangan. Adapun data yang kami gunakan adalah *Music Genre Classification* dari GTZAN Dataset. Data tersebut diambil dari kaggle dengan *link* berikut ini https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification/data Data yang digunakan yaitu sepuluh jenis genre musik dengan masing-masing 100 file audio, seluruh file berdurasi 30 detik yang terdiri dari genre *blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, dan rock.*

3.2 Desain Sistem

3.2.1 Use Case Diagram

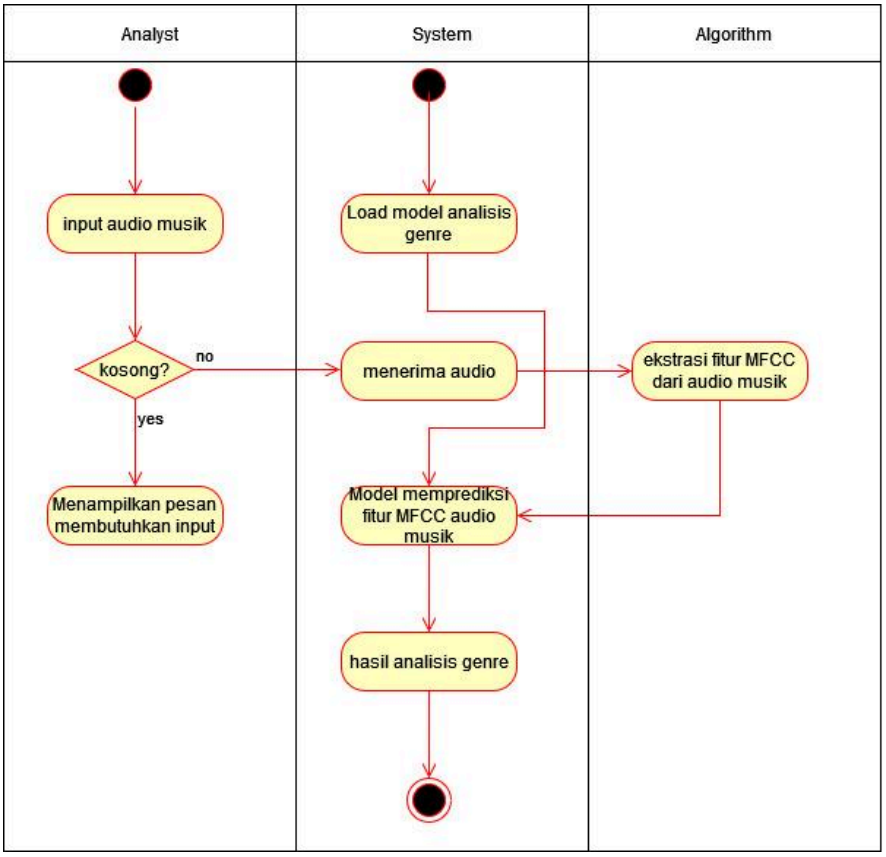


Gambar 3.2 *Use Case Diagram*

Adapun *use case* yang sudah disiapkan dalam proses pembuatan *website* klasifikasi genre musik, antara lain:

1. *User* atau analis dapat melakukan *input audio musik* dan sistem akan memprediksi genre dari musik yang diberikan.
2. Analis dapat melihat ekstraksi fitur MFCC dari musik yang sudah melalui tahap ekstraksi fitur.
3. Analis dapat melihat hasil dari klasifikasi genre musik yang diberikan oleh sistem berupa salah satu dari sepuluh jenis genre musik yaitu genre *blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, dan rock.*

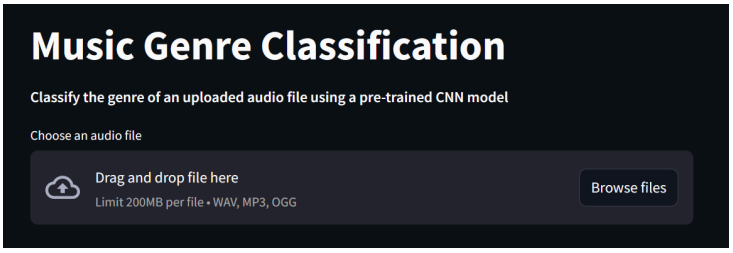
3.2.2 Activity Diagram



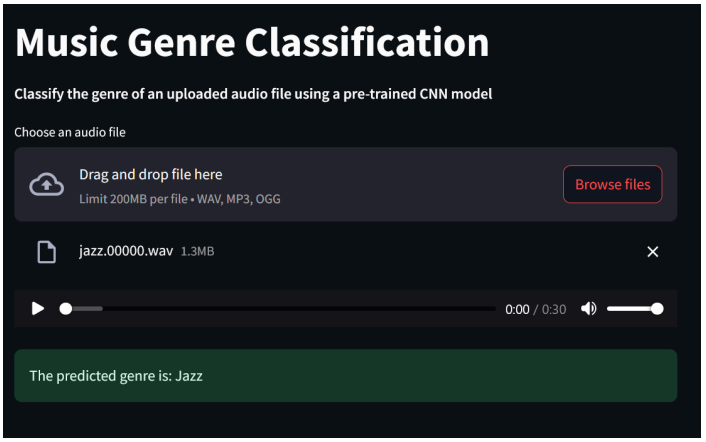
Gambar 3.3 *Activity Diagram*

User atau analis akan bertugas untuk menginputkan data musik dalam bentuk *file*. Sistem akan melakukan load model klasifikasi genre yang sebelumnya sudah dilakukan *training*. Jika *user* belum menginputkan file maka akan muncul pesan bahwa sistem membutuhkan *input*. *Input* yang telah dimasukkan oleh *user* akan melalui ekstraksi fitur MFCC terlebih dahulu sebelum diprediksi oleh model yang sudah dibuat. Kemudian model akan memprediksi audio musik tersebut dan mengklasifikasikannya ke salah satu jenis genre musik. Prediksi yang muncul pada musik adalah berupa salah satu dari sepuluh jenis genre musik yaitu genre *blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, dan rock.*

3.3 Desain UI/UX



Gambar 3.4 Desain *GUI Website Input Audio* dengan Menggunakan Figma



Gambar 3.5 Desain *GUI Website* Klasifikasi Genre dengan Menggunakan Figma

Pada bagian GUI web dengan input file, setelah analis atau *user* memasukkan file audio musik, sistem akan menampilkan audio yang diinputkan dan setelah itu akan muncul *loading screen* untuk sistem melakukan olah data dan menunggu hasil prediksi klasifikasi genre musik. Setelah itu akan muncul hasil klasifikasinya pada field prediksi genre apakah musik tersebut bergenre *blues, classical, country, disco, hiphop, jazz, metal, pop, reggae,* atau *rock.*

3.4 Skenario Eksperimen untuk Evaluasi Model

3.4.1 Persiapan Data

Data yang digunakan yaitu sepuluh jenis genre musik dengan masing-masing 100 file audio, seluruh file berdurasi 30 detik yang terdiri dari genre *blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, dan rock.* Selanjutnya dataset diupload ke sebuah Google Drive untuk memudahkan proses klasifikasi di Google Collab.

3.4.2 Representasi Data

Setelah itu dataset yang ada di Google Drive akan diinput untuk melalui proses ekstraksi fitur MFCC (*Mel-Frequency Cepstral Coefficients*) dari file audio dan menyimpan hasil ekstraksi ke dalam file CSV. Tujuan dari proses ekstraksi fitur MFCC (*Mel-Frequency Cepstral Coefficients*) adalah untuk menangkap karakteristik akustik penting dari file audio yang mewakili ciri-ciri yang berbeda dari setiap genre musik. MFCC memetakan sinyal audio mentah ke dalam representasi yang lebih kompak dan informatif, yang mencerminkan cara telinga manusia mendengar dan memproses suara. Dengan mengubah data audio menjadi serangkaian koefisien, MFCC menyediakan fitur yang lebih mudah dianalisis oleh algoritma *deep learning, Convolutional Neural Network (CNN)*.

3.4.3 Pengembangan Model

Kami akan menggunakan CNN sebagai algoritma klasifikasi. Untuk meningkatkan kinerja model, *tuning hyper-parameter* akan dilakukan menggunakan *grid search* dalam *k-fold cross validation*. *Hyper-parameter* yang akan disetel meliputi parameter *kernel* SVM, parameter regularisasi (C), dan parameter *gamma* (γ).

3.4.4 Evaluasi Model

Setiap kombinasi *hyper-parameter* akan dievaluasi menggunakan *k-fold cross validation*. *Dataset* akan dibagi menjadi 10 bagian, di mana 8 bagian digunakan untuk pelatihan dan 2 bagian digunakan untuk pengujian, secara bergiliran. Metode ini memastikan bahwa setiap data digunakan sebagai data pengujian sekali, sehingga memberikan evaluasi yang lebih akurat terhadap kinerja model. Hasil evaluasi akan diukur menggunakan metrik akurasi, presisi, recall, dan *F1-score*. Kombinasi *hyper-parameter* yang memberikan performa terbaik berdasarkan metrik-metrik ini akan dipilih sebagai model akhir.

3.5 Skenario Eksperimen untuk Evaluasi Sistem

3.5.1 Implementasi Sistem

Model terbaik yang telah dipilih akan diimplementasikan dalam web klasifikasi genre musik yang dirancang untuk mengklasifikasi genre musik secara otomatis. Web ini akan mencakup komponen untuk *pre-processing* data, ekstraksi fitur MFCC, dan klasifikasi

genre musik menggunakan model CNN hingga mendapatkan hasil presentasi genre musik yang sesuai.

3.5.2 Uji Coba Sistem

Sistem akan diuji menggunakan *dataset* ulasan baru yang tidak termasuk dalam data pelatihan atau pengujian sebelumnya. *Dataset* ini akan mencakup musik dengan berbagai genre dan kompleksitas untuk menguji generalisasi model. Musik ini akan diproses oleh sistem dan hasil klasifikasi akan dibandingkan dengan label genre untuk mengukur akurasi sistem.

3.5.3 Evaluasi Kinerja Sistem

Kinerja sistem akan dievaluasi berdasarkan beberapa hal utama yaitu akurasi sistem yang menunjukkan persentase genre musik yang diklasifikasikan dengan benar oleh sistem; waktu pemrosesan yang dibutuhkan sistem untuk memproses dan mengklasifikasikan setiap musik; serta emampuan sistem untuk tetap akurat meskipun terjadi variasi dalam data musik.

BAB IV

HASIL DAN PEMBAHASAN

https://github.com/bgsptr/B2_MusicGenreClassification_MFCC_CNN.git

4.1 Implementasi Tahap *Preprocessing*

4.1.1 *Framing*

Proses *preprocessing* pada penelitian ini melibatkan beberapa langkah penting dalam pengolahan sinyal audio. Pertama, file audio dimuat dan dikonversi menjadi *array* amplitudo menggunakan librosa.load. Kemudian, sinyal audio ini dibagi menjadi frame-frame kecil dalam proses yang disebut *framing*. Panjang setiap frame ditentukan oleh parameter n_fft, yang dalam kode ini diatur menjadi 2048, menunjukkan jumlah sampel dalam setiap frame. Proses framing ini membagi sinyal audio menjadi potongan-potongan kecil yang lebih mudah diolah dan dianalisis.

4.1.2 *Windowing*

Setelah *framing*, setiap frame diterapkan fungsi *windowing*, biasanya menggunakan *Hamming window* secara *default* di librosa. *Windowing* digunakan untuk meredam efek bocoran (*leakage*) dengan meredam tepi-tepi dari frame. Ini penting karena tanpa *windowing*, perpotongan sinyal di tepi frame dapat menghasilkan artefak yang tidak diinginkan dalam analisis frekuensi. Selain itu, parameter hop_length digunakan untuk menentukan jarak antara titik awal frame-frame yang berurutan, memungkinkan frame-frame tersebut untuk saling tumpang tindih sebagian besar.

4.2 Hasil dan pembahasan Tahap *Preprocessing*

4.2.1 *Framing*

Tahap *framing* dalam pengolahan sinyal audio menghasilkan potongan-potongan kecil dari sinyal asli yang lebih mudah dianalisis. Dengan membagi sinyal audio menjadi frame-frame kecil dengan panjang yang ditentukan oleh parameter n_fft, mendapatkan sejumlah frame yang masing-masing berisi sejumlah sampel dari sinyal audio. Hal ini memungkinkan untuk melakukan analisis lebih detail pada setiap segmen kecil dari sinyal tersebut. *Framing* ini penting karena karakteristik frekuensi dari sinyal audio dapat berubah seiring waktu, sehingga dengan memecahnya menjadi frame-frame kecil, kita dapat menangkap variasi temporal ini.

4.2.2 Windowing

Tahap windowing dalam pengolahan sinyal audio menghasilkan frame-frame yang telah diterapkan fungsi *window*, seperti *Hamming window*, untuk meredam tepi-tepi frame. Hasil dari *windowing* adalah sinyal yang lebih halus pada tepi frame, yang mengurangi efek bocoran atau *leakage* yang bisa terjadi saat melakukan transformasi Fourier. Dengan menerapkan *windowing*, dapat memastikan bahwa transisi antara frame-frame yang berurutan lebih mulus, yang membantu dalam mendapatkan representasi frekuensi yang lebih akurat dan mengurangi artefak yang tidak diinginkan dalam analisis spektral.

4.3 Implementasi Ekstraksi Fitur

Setelah *framing* dan *windowing*, fitur MFCC diekstraksi menggunakan fungsi librosa.feature.mfcc. Fungsi ini menghitung koefisien MFCC untuk setiap frame dengan

mengonversi sinyal dari domain waktu ke domain frekuensi menggunakan transformasi Fourier dan kemudian memetakannya ke skala Mel yang lebih sesuai dengan persepsi manusia terhadap frekuensi. Jumlah koefisien MFCC yang diekstraksi ditentukan oleh parameter n_mfcc, yang dalam kode ini diatur menjadi 13. Koefisien ini memberikan informasi tentang konten spektral dari sinyal audio dan digunakan untuk klasifikasi genre musik.

Setelah fitur *Mel-Frequency Cepstral Coefficients* (MFCC) diekstraksi dari setiap frame, langkah selanjutnya adalah memastikan bahwa semua frame MFCC memiliki panjang yang konsisten. Ini dilakukan dengan menghitung panjang maksimum dari semua frame dan menambahkan *padding* dengan nilai nol pada frame yang lebih pendek dari panjang maksimum tersebut. Langkah ini penting untuk memenuhi persyaratan input model *deep learning* yang biasanya memerlukan dimensi input yang seragam. Setelah *padding*, frame-frame MFCC tersebut diberi dimensi tambahan sehingga memiliki format input yang sesuai untuk model *Convolutional Neural Network* (CNN). Akhirnya, semua frame yang telah diproses disusun ke dalam satu *array* besar, menghasilkan data dengan dimensi yang konsisten untuk digunakan dalam pelatihan model.

Berikut keseluruhan code untuk tahap preprocessing dan ekstraksi fitur dengan MFCC:

```
import os

import pandas as pd

import librosa

import numpy as np

# Fungsi untuk mengekstrak MFCC dari file audio

def extract_mfcc(file_path, n_mfcc=13, hop_length=512, n_fft=2048):

    y, sr = librosa.load(file_path, sr=None)

        mfccs = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=n_mfcc, hop_length=hop_length, n_fft=n_fft)

    return mfccs
```

```
    # Menyimpan path, mood, mfcc
    df = pd.DataFrame(columns=["path", "mood_label", "mfcc"])
    drive_path = 'drive/MyDrive/genres_original'
    folders = os.listdir(drive_path)
    for f in folders:
        emotion_path = os.path.join(drive_path, f)
        files = os.listdir(emotion_path)
        for data in files:
            file_path = os.path.join(emotion_path, data)
            mfcc = extract_mfcc(file_path)
                df = pd.concat([df, pd.DataFrame({"path": [file_path],
"mood_label": [f], "mfcc": [mfcc]})], ignore_index=True)
    print(df.head())
    df.to_csv('mfcc-library-genre.csv')
```

4.4 Implementasi *Deep Learning*

Implementasi *deep learning Convolutional Neural Network* (CNN) dalam penelitian ini bertujuan untuk mengklasifikasikan sinyal audio berdasarkan fitur *Mel-Frequency Cepstral Coefficients* (MFCC). Model ini dimulai dengan menentukan beberapa *hyperparameters* penting, seperti jumlah baris (`num_rows`), kolom (`num_columns`), channel (`num_channels`), dan jumlah label (`num_labels`). Struktur model dibangun menggunakan kelas `Sequential` dari Keras, yang memungkinkan penyusunan layer secara berurutan.

Model CNN ini terdiri dari beberapa lapisan konvolusi dan *pooling*. Pada lapisan pertama, sebuah `Conv2D` dengan 32 filter dan kernel berukuran 3x3 diterapkan pada input data yang memiliki dimensi `(13, 1320, 1)`, diikuti oleh aktivasi `LeakyReLU` dengan parameter alpha 0.1. Untuk mengurangi *overfitting* dan meningkatkan generalisasi, lapisan ini juga diikuti oleh `BatchNormalization` dan `SpatialDropout2D` dengan tingkat dropout sebesar 0.07. Lapisan konvolusi kedua mengikuti struktur yang

mirip, tetapi dengan tambahan `MaxPooling2D` untuk mengurangi dimensi spasial dari fitur map.

Lapisan konvolusi ketiga menerapkan filter dengan ukuran 64 dan kernel 3x3, diikuti lagi oleh `LeakyReLU`, `BatchNormalization`, dan `SpatialDropout2D` untuk mengurangi *overfitting*. Meskipun ada komentar yang menunjukkan kemungkinan penambahan lapisan konvolusi keempat, lapisan tersebut dinonaktifkan dalam kode ini. Untuk menggabungkan fitur dari semua filter, model ini menggunakan `GlobalAveragePooling2D`, yang mengurangi dimensi setiap fitur map menjadi satu angka dengan mengambil rata-rata dari semua nilai dalam fitur map tersebut.

Terakhir, model ini diakhiri dengan lapisan `Dense` yang memiliki jumlah neuron sesuai dengan jumlah label klasifikasi (`num_labels`) dan menggunakan fungsi aktivasi `softmax` untuk menghasilkan probabilitas kelas dari setiap input. Model ini dirancang untuk klasifikasi dengan mengubah representasi fitur MFCC dari sinyal audio menjadi prediksi kelas, memanfaatkan kekuatan lapisan konvolusi dan *pooling* untuk ekstraksi fitur yang efektif.

Berikut keseluruhan code untuk tahap pembangunan model *deep learning* dengan CNN:

```
import pandas as pd

import numpy as np

import tensorflow as tf

from sklearn.model_selection import train_test_split

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, BatchNormalization, LeakyReLU, SpatialDropout2D, GlobalAveragePooling2D

from tensorflow.keras.utils import to_categorical

from keras.regularizers import l2

# Hyperparameters

num_rows = 13

num_columns = 1320
```

```
num_channels = 1

num_labels = 10

defcreate_model(spatial_dropout_rate_1=0.07,
spatial_dropout_rate_2=0.14, l2_rate=0.0005):

    model = Sequential()

    # Conv 1

    model.add(Conv2D(filters=32,

                         kernel_size=(3, 3),

                         kernel_regularizer=l2(l2_rate),

                                      input_shape=(num_rows,  num_columns,
num_channels)))

    model.add(LeakyReLU(alpha=0.1))

    model.add(BatchNormalization())

    model.add(SpatialDropout2D(spatial_dropout_rate_1))

    # Conv 2

    model.add(Conv2D(filters=32,

                         kernel_size=(3, 3),

                         kernel_regularizer=l2(l2_rate)))

    model.add(LeakyReLU(alpha=0.1))

    model.add(BatchNormalization())

    model.add(MaxPooling2D(pool_size=(2, 2)))

    # Conv 3

    model.add(SpatialDropout2D(spatial_dropout_rate_1))

    model.add(Conv2D(filters=64,

                         kernel_size=(3, 3),

                         kernel_regularizer=l2(l2_rate)))

    model.add(LeakyReLU(alpha=0.1))

    model.add(BatchNormalization())
```

```
    model.add(Dense(num_labels, activation='softmax'))

    return model

model = create_model()
```

4.4 Hasil dan Pembahasan Tahap Training dan Testing Model

4.4.1 Hasil dan Pembahasan Tahap Training (Tuning Hyper-Parameter untuk Pemilihan Model Terbaik dengan K-Fold Cross Validation)

Pada tahap *training* model, label kelas dari data *training* (y_train) diubah menjadi format *one-hot encoding* menggunakan LabelEncoder dan to_categorical dari Keras. LabelEncoder mengonversi label kelas yang berupa *string* atau kategori menjadi angka integer, kemudian to_categorical mengubah angka-angka ini menjadi representasi one-hot, yang diperlukan untuk klasifikasi multi-kelas pada model *deep learning*. Ini menghasilkan matriks binar dari label yang digunakan sebagai target dalam pelatihan model.

Setelah itu, model dikompilasi menggunakan optimizer Adam dengan *learning rate* sebesar 0.0001, serta loss function categorical_crossentropy yang cocok untuk klasifikasi multi-kelas. metrics=['accuracy'] digunakan untuk mengevaluasi kinerja model selama pelatihan dan *testing*. Proses kompilasi ini mengonfigurasikan model untuk pelatihan dengan menetapkan parameter-parameter penting seperti *optimizer* dan *loss function.*

Pelatihan model dilakukan dengan memanggil metode fit pada model, menggunakan data *training* (X_train dan y_train_encoded). Model dilatih selama 100 epoch dengan *validation split* sebesar 1/12 dari data *training*, yang berarti sebagian kecil dari data training digunakan untuk validasi pada setiap epoch. *Batch size* ditetapkan sebesar 32, yang menentukan jumlah sampel yang diproses sebelum pembaruan parameter model. Selama pelatihan, model mempelajari pola dari data *training* dan menyesuaikan bobotnya untuk meminimalkan *loss function*. Hasil dari proses ini adalah model yang terlatih, yang diharapkan mampu mengklasifikasikan data baru dengan akurasi yang baik.

4.4.2 Hasil dan Pembahasan Tahap Testing Model

Pada tahap *testing* model, data *testing* (X_test dan y_test_encoded) digunakan untuk mengevaluasi performa model yang telah dilatih. Label kelas dari data *testing* juga

diubah menjadi format *one-hot encoding* menggunakan LabelEncoder dan to_categorical. Model yang sudah terlatih ini diuji dengan data testing untuk mengukur seberapa baik model dapat memprediksi label dari data *testing* dibandingkan dengan label sebenarnya. Akurasi dan performa model dievaluasi berdasarkan metrik akurasi yang sudah ditentukan saat kompilasi model.

Hasil testing ini memberikan gambaran tentang kemampuan generalisasi model terhadap data yang belum pernah dilihat selama pelatihan. Evaluasi performa model pada data testing penting untuk menilai kinerja model secara keseluruhan dan memastikan bahwa model tidak hanya bekerja baik pada data *training*, tetapi juga pada data yang belum pernah dilihat sebelumnya. Ini membantu dalam mengidentifikasi *overfitting* dan memastikan bahwa model dapat diaplikasikan pada data nyata dengan tingkat akurasi yang memadai.

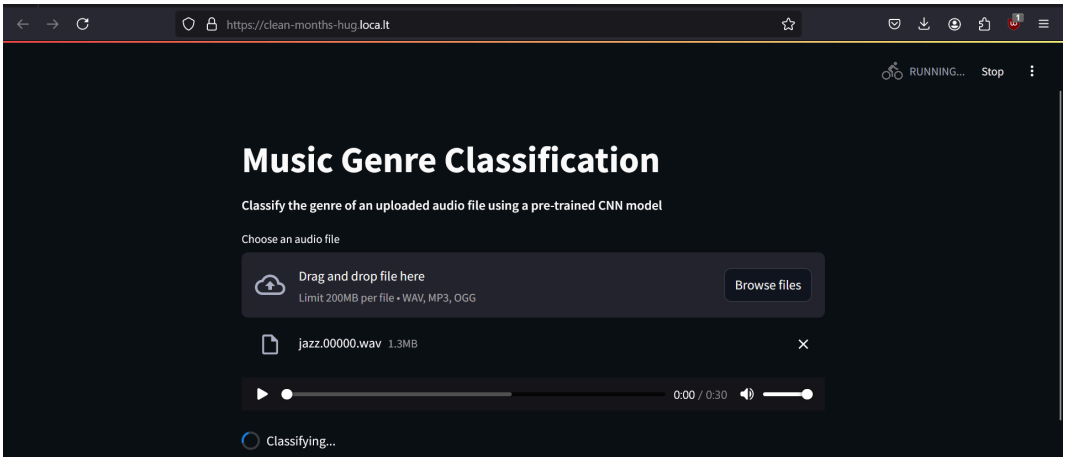
4.5 Hasil Antarmuka Fitur Aplikasi

Kami berhasil menyusun sebuah aplikasi web berbasis streamlit yang mengaplikasikan mesin yang dilatih untuk melakukan klasifikasi genre musik. Dalam aplikasi tersebut, pengguna bisa menginput sebuah file audio musik. Sistem aplikasi akan mengolah file audio musik tersebut melalui tahap *pre-processing* yang serupa dengan pemrosesan yang dilakukan pada sistem pelatihan model. Dari hasil *pre-processing*, akan dilakukan ekstraksi fitur dengan MFCC kemudian sistem akan melakukan klasifikasi genre musik menggunakan model yang telah dibuat dengan *deep learning* CNN. Untuk setiap file audio musik hasil klasifikasi yaitu berupa salah satu jenis genre. Klasifikasi tersebut mengaplikasikan model terlatih yang telah diekspor dalam *Hierarchical Data Format* (.h5).

Berikut adalah hasil antarmuka aplikasi yang kami susun:

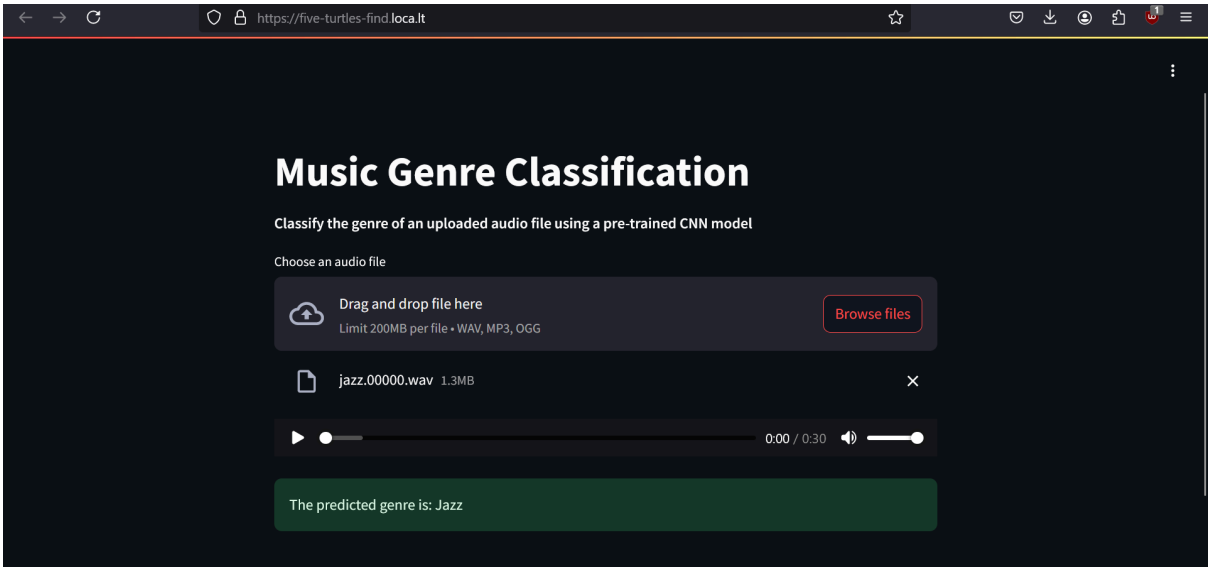
Gambar 4.8 Aplikasi Menginput File Audio

Setelah pengguna mengunggah file audio ke aplikasi. Setelah beberapa saat, aplikasi menampilkan audio yang telah diinputkan Aplikasi menjalankan proses ekstraksi fitur MFCC dan klasifikasi genre dengan model CNN.



Gambar 4.10 *Loading* Klasifikasi Genre Musik

Setelah tahap klasifikasi selesai, aplikasi menampilkan hasil klasifikasi genre musik



Gambar 4.11 Klasifikasi Genre Musik Selesai

4.6 Hasil Evaluasi Sistem Perangkat Lunak (*Black Box Testing*)

Pengujian *blackbox* pada sistem klasifikasi genre musik dilakukan dengan menginput 1 file audio musik bergenre salah satu dari 10 genre yang ada di penelitian kali ini. Adapun file audio musik yang kami gunakan yaitu musik dengan genre jazz.

Hasil *blackbox testing* sistem klasifikasi genre musik yang kami input yaitu seperti berikut

BAB V

KESIMPULAN

Penelitian ini menggunakan data sekunder dari GTZAN Dataset yang diambil dari Kaggle, yang berisi 100 file audio berdurasi 30 detik dari sepuluh genre musik. Proses *preprocessing* data melibatkan beberapa langkah penting, termasuk memuat dan mengonversi file audio menjadi array amplitudo, membagi sinyal audio menjadi frame-frame kecil (*framing*), dan menerapkan *windowing* untuk meredam efek bocoran. Fitur *Mel-Frequency Cepstral Coefficients* (MFCC) diekstraksi dari setiap frame untuk mendapatkan representasi karakteristik frekuensi yang relevan dengan persepsi manusia terhadap suara. Selanjutnya, semua frame MFCC diproses untuk memastikan panjang yang konsisten dengan menambahkan padding pada frame yang lebih pendek.

Model *deep learning* yang digunakan adalah *Convolutional Neural Network* (CNN), yang terdiri dari beberapa lapisan konvolusi dan pooling. Tujuan utama dari model ini adalah mengklasifikasikan sinyal audio berdasarkan fitur MFCC yang telah diekstraksi. Proses pelatihan model dilakukan dengan mengubah label kelas menjadi format one-hot encoding dan mengompilasi model menggunakan optimizer Adam dengan learning rate 0.0001. Model dilatih selama 100 epoch dengan validation split 1/12 dari data training dan batch size 32. Setelah pelatihan, model diuji dengan data testing untuk mengevaluasi performa dan kemampuan generalisasi terhadap data yang belum pernah dilihat sebelumnya. Hasil testing menunjukkan akurasi model dan membantu dalam mengidentifikasi overfitting, memastikan bahwa model tidak hanya bekerja baik pada data training tetapi juga dapat diaplikasikan pada data nyata dengan tingkat akurasi yang memadai.

Daftar Pustaka

Bisht, A. S., Negi, C. M. S., Singh, R., dkk. Classification of Indian Classical Music (Hindustani Music) Genres through MFCCs Features using RNN-LSTM Model, 08 December 2022, PREPRINT (Version 1) available at Research Square [https://doi.org/10.21203/rs.3.rs-2348537/v1]

Chandani, V., Wahono, R. S., & Purwanto, . (2015). Komparasi Algoritma Klasifikasi Machine Learning Dan Feature Selection pada Analisis Sentimen Review Film. *Journal of Intelligent Systems*, *1*(1), 56–60.

Gifari, O. I., Adha, Muh., Hendrawan, I. R., & Durrand, F. F. S. (2022). Analisis Sentimen ReviewFilm Menggunakan TF-IDF dan Support Vector Machine. *JIFOTECH: Journal of Information Technology,2*(1), 36–40.

Hermawan, L., & Ismiati, M. B. (2020). Pembelajaran TextPreprocessing berbasis Simulator Untuk Mata Kuliah Information Retrieval. *Jurnal Transformatika*, *17*(2), 188–199.

Karo, I. M. K., Karo, J. A. K., Yunianto, , Hariyanto, , Falah, M., & Ginting, M. (2023). Analisis Sentimen Ulasan Aplikasi Info BMKG di Google Play Menggunakan TF-IDF dan Support Vector Machine. *Journal of Information System Research (JOSH)*, *4*(4), 1423–1430. https://doi.org/ 10.47065/josh.v4i4.3943

Khairunnisa, S., Adiwijaya, , & Faraby, S. A. (2021). Pengaruh Text Preprocessing terhadap Analisis Sentimen Komentar Masyarakat pada Media Sosial Twitter (Studi Kasus Pandemi COVID-19). *Jurnal Media Informatika Budidarma*, *5*(2), 406–414. https://doi.org/10.30865/mib.v5i2.2835

Praghakusma, A. Z., & Charibaldi, N. (2021). Komparasi Fungsi Kernel Metode Support Vector Machine untuk Analisis Sentimen Instagram dan Twitter (Studi Kasus : Komisi Pemberantasan Korupsi) . *Jurnal Sarjana Teknik Informatika*, *9*(2), 33–42.

Riyaddulloh, R., & Romadhony, A. (2021). Normalisasi Teks Bahasa Indonesia Berbasis Kamus Slang Studi Kasus: Tweet Produk Gadget Pada Twitter. *E-Proceeding of Engineering*, *8*(4), 4216–4228.

Sierra, D. (2019, February 13). *Algoritma TF — IDF*. Medium. https://dltsierra.medium.com/algoritma-tf-idf-633e17d10a80

Surya T, Y., Faraby, S. A., & Dwifebri, M. (2021). Analisis Sentimen Terhadap Ulasan Film

Menggunakan Word2Vec dan SVM. *E-Proceeding of Engineering*, *8*(4), 4136–4144.

Thomas, V. W. D., & Rumaisa, F. (2022). Analisis Sentimen Ulasan Hotel Bahasa Indonesia Menggunakan Support Vector Machine dan TF-IDF. *Jurnal Media Informatika Budidarma*, *6*(3), 1767–1774. https://doi.org/10.30865/mib.v6i3.4218

Verawati, I., & Audit, B. S. (2022). Algoritma Naïve Bayes Classifier Untuk Analisis Sentiment Pengguna Twitter Terhadap Provider By.u . *Jurnal Media Informatika Budidarma*, *6*(3), 1411–1417. https://doi.org/10.30865/mib.v6i3.4132