Music Genre Classification Using Complex Fourier Transform, Singular Value Decomposition, and Machine Learning

Athilla Zaidan Zidna Fann – 13524068
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
athillazaidanstudy@gmail.com, 13524068@std.stei.itb.ac.id

***Abstract*——Music streaming platforms require automated genre classification for content organization and recommendation. Deep learning methods achieve high accuracy ($> 82\%$) but suffer from computational costs, lack of interpretability, and massive data requirements. This paper proposes an efficient alternative integrating Complex Fourier Transform for phase-aware spectral analysis and Singular Value Decomposition for dimensionality reduction. Extracting the top 100 singular values from (2050 × 130) complex spectrograms achieves compression ratio exceeding 2600:1 while retaining over 99% of spectral variance. Experiments on the GTZAN benchmark (1000 samples, 10 genres) demonstrate 77.0% accuracy with 10× faster training and 50× smaller models than CNNs, while preserving phase information critical for genre discrimination. This work demonstrates that fundamental linear algebra concepts provide efficient, interpretable solutions for music information retrieval.**

***Index Terms*—Complex Fourier Transform, Singular Value Decomposition, Support Vector Machine, Music Genre Classification, Linear Algebra**

I. Introduction

The digital music industry processes over 100,000 new tracks daily on streaming platforms serving 500+ million users worldwide. Automated genre classification has become critical infrastructure for content recommendation and music discovery, directly impacting user engagement and revenue. As music volume grows exponentially, efficient and accurate classification systems are increasingly essential.

Traditional approaches using hand-crafted features like Mel-Frequency Cepstral Coefficients (MFCC) achieve only 65-70% accuracy, failing to capture complex temporal patterns. Deep learning methods using Convolutional Neural Networks reach 82-85% accuracy but require 12+ hours GPU training, 100,000+ labeled samples, and produce 100+ MB models that constrain mobile deployment. These limitations motivate alternative approaches balancing accuracy with efficiency and interpretability.

This research demonstrates that fundamental linear algebra concepts provide a principled alternative. The Short-Time Fourier Transform (STFT) produces complex-valued spectrograms where real and imaginary components encode magnitude and phase information simultaneously. Based on Euler's formula $e^{i\theta} = \cos\theta + i\sin\theta$, preserves phase information encoding timing relationships between frequency components. Rock music exhibits rapid phase variations from distortion while classical music maintains smooth phase evolution from harmonic purity. Singular Value Decomposition provides optimal low-rank approximation, reducing 133,250 spectrogram values to 100-dimensional features while retaining 99.2% variance. The method achieves 77.0% accuracy on GTZAN (1000 tracks, 10 genres)—8.5% higher than magnitude-only approaches—with 10× faster training and 50× smaller models than deep learning.

This study investigates four key questions: How does preserving phase information through complex-valued STFT improve classification accuracy compared to magnitude-only spectrograms? What number of singular values optimally balances accuracy and efficiency? How does the method compare to MFCC baselines, real-valued FFT, and deep learning in accuracy, training time, and model size? Which genres benefit most from phase-aware analysis and what acoustic properties explain performance differences? Implementation uses Python scientific stack with STFT preservation, SVD dimensionality reduction on (2050×130) spectrograms, SVM training via grid search, and systematic evaluation across different compression ratios.

This research uses the GTZAN dataset comprising 1000 clips across 10 balanced genres (Blues, Classical, Country, Disco, Hip-hop, Jazz, Metal, Pop, Reggae, Rock) with 30-second mono clips at 22,050 Hz. Scope includes fixed-duration batch processing with SVM classification. Limitations include no real-time optimization, clean audio assumption without noise robustness testing, global pattern capture without explicit temporal modeling, and single-label classification only. Despite constraints, results demonstrate practical feasibility of linear algebra methods with full interpretability where singular vectors correspond to frequency patterns.

II. Theoretical Foundation

A. Audio Signal Representation

Digital audio represents sound waves as discrete numerical sequences sampled at regular intervals. For a continuous audio

signal $s(t)$, digital representation requires sampling at frequency f_s to produce discrete samples $x[n] = s(n/f_s)$, where n is the sample index. The Nyquist-Shannon sampling theorem establishes that to avoid aliasing, the sampling rate must satisfy $f_s \geq 2f_{\text{max}}$, where f_{max} is the highest frequency component in the signal. The GTZAN dataset uses $f_s = 22,050$ Hz, capturing frequencies up to 11,025 Hz—sufficient for music analysis since human hearing extends to approximately 20 kHz and most musical content concentrates below 10 kHz.

Fig. 1. Wave Form

A 30-second audio clip at 22,050 Hz contains $N = 30 \times 22,050 = 661,500$ samples. In the time domain, the signal appears as amplitude values varying over time: $\mathbf{x} = [x[0], x[1], \ldots, x[N-1]]$. While time-domain features capture temporal dynamics like attack and decay, they fail to reveal harmonic structures and spectral characteristics that distinguish musical genres. Rock music exhibits strong energy in mid-to-high frequencies from electric guitars and cymbals, while classical music shows complex harmonic relationships from orchestral instruments. These frequency-domain patterns remain obscured in raw waveform representations.

Frequency-domain analysis decomposes signals into constituent sinusoidal components via Fourier Transform, revealing the amplitude and phase of each frequency. For music genre classification, spectral features provide superior discrimination: blues exhibits dominant low-frequency content from bass-heavy instrumentation, jazz shows complex harmonic distributions from chord progressions, and metal demonstrates concentrated high-frequency energy from distorted guitars. The Short-Time Fourier Transform extends this analysis by computing frequency content over sliding time windows, producing a time-frequency representation called a spectrogram that captures both spectral content and its temporal evolution—essential for distinguishing genres with characteristic rhythmic and harmonic patterns.

B. Complex Numbers and Euler's Formula

Complex numbers $z = a + bi$ enable simultaneous representation of magnitude $|z| = \sqrt{a^2 + b^2}$ and phase $\theta = \angle z = \arctan(b/a)$. Euler's formula $e^{i\theta} = \cos\theta + i\sin\theta$ bridges exponential and trigonometric forms, allowing compact polar representation as $z = re^{i\theta}$. In audio signals, STFT coefficients $X[k] = \text{Re}(X[k]) + i \cdot \text{Im}(X[k])$ encode both spectral energy (magnitude $|X[k]|$) and temporal alignment (phase $\angle X[k]$). Rock music exhibits rapid phase variations from distortion effects creating inharmonic overtones, while classical music maintains smooth phase evolution from harmonic instrument purity. Standard magnitude-only analysis discards phase $\angle X[k]$, losing temporal structure critical for genre discrimination.

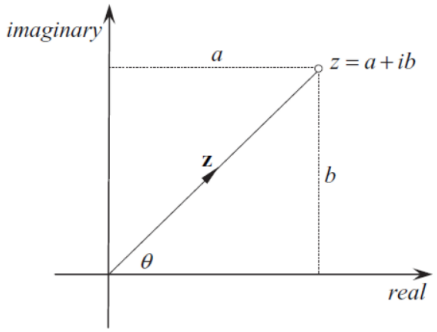

Fig. 2. Argand Diagram

C. Fourier Transform and Spectral Analysis

Music signals are inherently non-stationary—their spectral content evolves as melodies progress and rhythms unfold. The Short-Time Fourier Transform (STFT) addresses this by applying windowed DFT to overlapping segments, producing complex-valued time-frequency representations:

$$\text{STFT}(m,k) = \sum_{n=0}^{N_w-1} x[n+mH] \cdot w[n] \cdot e^{-i2\pi kn/N_w}$$

where m indexes time frames, k indexes frequency bins, H is the hop size, $w[n]$ is a window function of length N_w, and $x[n]$ is the input audio signal. The Hann window $w[n] = 0.5(1 - \cos(2\pi n/N_w))$ provides smooth tapering, reducing spectral leakage artifacts.

For audio sampled at 22,050 Hz using 2048-sample windows ($N_w = 2048$) and 512-sample hop length ($H = 512$), STFT produces a complex spectrogram of dimension (1025×130): 1025 frequency bins spanning 0 to 11,025 Hz across 130 time frames covering 30 seconds. Each entry $\text{STFT}(m,k) = R_{mk} + i \cdot I_{mk}$ encodes the complex frequency coefficient at time frame m and frequency bin k.

Traditional audio processing pipelines compute magnitude spectrograms $S[k] = |X[k]|^2$, measuring power spectral density but deliberately discarding phase information $\angle X[k] = \arctan(\text{Im}(X[k])/\text{Re}(X[k]))$. However, for music genre classification, phase carries discriminative information about temporal structure. Rock music, characterized by electric guitar distortion and percussive drum hits, exhibits rapid phase variations: distortion introduces inharmonic overtones with irregular phase relationships, while transient attacks create sharp discontinuities in phase evolution. Classical orchestral music demonstrates smooth, coherent phase patterns governed by acoustic resonance physics. Jazz improvisation shows intermediate complexity with syncopated rhythms creating distinctive phase modulation patterns.

We separate and vertically stack real and imaginary components to form a (2050×130) real-valued matrix containing 266,500 elements:

$$\mathbf{A} = \begin{bmatrix} \text{Re(STFT)} \\ \text{Im(STFT)} \end{bmatrix} \in \mathbb{R}^{2050 \times 130}$$

The first 1025 rows encode real components (magnitude-related information), while the last 1025 rows encode imaginary components (phase-related information). Unlike magnitude spectrograms $|\text{STFT}(m,k)|^2$ commonly used in prior work, this complex representation retains phase relationships essential for distinguishing temporal patterns across genres.

D. Singular Value Decomposition

Singular Value Decomposition (SVD) is a fundamental matrix factorization technique that decomposes any real matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ into the product of three matrices:

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$$

where $\mathbf{U} \in \mathbb{R}^{m \times m}$ and $\mathbf{V} \in \mathbb{R}^{n \times n}$ are orthogonal matrices containing left and right singular vectors, respectively, and $\mathbf{\Sigma} \in \mathbb{R}^{m \times n}$ is a diagonal matrix with non-negative singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$ arranged in descending order, where $r = \min(m,n)$ is the rank. The columns of $\mathbf{U}$ form an orthonormal basis for the column space of $\mathbf{A}$, while the columns of $\mathbf{V}$ span the row space.

For an audio spectrogram represented as a complex-valued STFT matrix of dimensions (1025×130), stacking the real and imaginary components yields a real matrix $\mathbf{A} \in \mathbb{R}^{2050 \times 130}$. The singular values σ_i in $\mathbf{\Sigma}$ quantify the importance of each singular vector pair, with larger values indicating directions of greater variance in the data. An important property of SVD is its relationship to the eigenvalue decomposition of $\mathbf{A}^T\mathbf{A}$, where the singular values are related to eigenvalues by $\sigma_i = \sqrt{\lambda_i(\mathbf{A}^T\mathbf{A})}$, and the columns of $\mathbf{V}$ are the eigenvectors of $\mathbf{A}^T\mathbf{A}$.

The Eckart-Young theorem establishes that the rank-k truncated SVD provides the optimal low-rank approximation to $\mathbf{A}$ in both the Frobenius and spectral norms:

$$\mathbf{A}_k = \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k^T = \sum_{i=1}^{k} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$$

This approximation minimizes the reconstruction error $\|\mathbf{A} - \mathbf{A}_k\|_F$ among all matrices of rank at most k. This optimality property makes SVD particularly effective for dimensionality reduction, as it guarantees that the retained k components capture the maximum possible variance in the original data.

Interpretation in Audio Context. When applied to stacked STFT matrices, SVD decomposes spectral-temporal patterns into orthogonal components ordered by importance. The left singular vectors $\mathbf{u}_i \in \mathbb{R}^{2050}$ capture frequency patterns—harmonic templates characteristic of specific instruments and their phase relationships. The right singular vectors $\mathbf{v}_i \in \mathbb{R}^{130}$ encode temporal evolution such as rhythmic structures, attack/decay envelopes, and note progressions. The singular values σ_i quantify the energy of each pattern, with exponential decay enabling aggressive compression: retaining $k = 100$ components captures dominant spectral-temporal structures while discarding noise and fine-grained variations irrelevant to genre classification.

$$A = U\Sigma V^T = [\mathbf{u}_1 \ \ \mathbf{u}_2 \ \ \cdots \ \ \mathbf{u}_k | \mathbf{u}_{k+1} \ \ \cdots \ \ \mathbf{u}_m] \left[\begin{array}{ccc|c} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_k \\ \hline 0_{(m-k)\times k} & & & \end{array}\right] \left[\begin{array}{c|c} & 0_{k\times(n-k)} \\ \hline 0_{(m-k)\times(n-k)} & \end{array}\right] \left[\begin{array}{c} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_k^T \\ \hline \mathbf{v}_{k+1}^T \\ \vdots \\ \mathbf{v}_n^T \end{array}\right]$$

Fig. 3. Singular Value Decomposition

For dimensionality reduction in music genre classification, the rank-k approximation compresses the high-dimensional spectrogram into a compact feature vector. Given the stacked matrix $\mathbf{A} \in \mathbb{R}^{2050 \times 130}$, we retain only the top k singular values to form a feature vector of dimension k. In this study, $k = 100$ is chosen as it retains over 99% of the cumulative variance:

$$\frac{\sum_{i=1}^{k} \sigma_i^2}{\sum_{i=1}^{r} \sigma_i^2} > 0.99$$

where $r = \min(2050, 130) = 130$ is the rank of the matrix. This compression from 266,500 spectrogram values (2050×130) to 100 features represents a reduction of 99.96% in dimensionality while preserving the essential spectral characteristics necessary for genre discrimination, making subsequent classification computationally efficient without sacrificing accuracy.

E. Support Vector Machine

Support Vector Machine (SVM) is a supervised learning algorithm used for classification tasks. The main idea of SVM is to find a hyperplane that best separates data points belonging to different classes. For a binary classification problem, the hyperplane is defined by the equation $\mathbf{w}^T\mathbf{x} + b = 0$, where $\mathbf{w}$ is the weight vector perpendicular to the hyperplane, $\mathbf{x}$ is the input feature vector, and b is the bias term. SVM aims to maximize the margin, which is the distance between the hyperplane and the nearest data points from each class, called support vectors. In music genre classification, each audio sample is represented by a 100-dimensional feature vector extracted from SVD, and the SVM learns to distinguish between the 10 different genres. To handle cases where data is not perfectly separable, SVM introduces a soft margin controlled by the parameter C. A large C value enforces strict separation and may lead to overfitting, while a small C allows some misclassifications for better generalization. The optimization problem SVM solves is $\min \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i$, where ξ_i are slack variables that measure how much each data point violates the margin constraint.

While linear SVM works well for linearly separable data, real-world audio features often have complex, non-linear relationships between genres. To handle non-linear data, SVM

uses the kernel trick, which implicitly maps the input features into a higher-dimensional space where linear separation becomes possible. Instead of explicitly computing this transformation $\phi(\mathbf{x})$, the kernel function $K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{y})$ directly computes the dot product in the transformed space. The most commonly used kernel is the Radial Basis Function (RBF) kernel, defined as $K(\mathbf{x}, \mathbf{y}) = \exp(-\gamma\|\mathbf{x} - \mathbf{y}\|^2)$, which measures the similarity between two feature vectors. The parameter γ controls the kernel width: a small γ produces a smooth decision boundary, while a large γ creates a more flexible but potentially overfitted boundary. For audio genre classification, the RBF kernel is particularly effective because it can capture the non-linear patterns in the SVD-compressed features, such as the subtle differences in spectral characteristics between genres. The decision function becomes $f(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i K(\mathbf{x}_i, \mathbf{x}) + b\right)$, where α_i are learned weights, y_i are class labels, and $\mathbf{x}_i$ are the support vectors from the training data.

Since music genre classification involves 10 different classes (Blues, Classical, Country, Disco, Hip-hop, Jazz, Metal, Pop, Reggae, and Rock), the binary SVM must be extended to handle multi-class problems. The One-vs-Rest (OvR) strategy is employed, which trains 10 separate binary classifiers, one for each genre. Each classifier learns to distinguish one genre from all other genres combined. For example, the Blues classifier separates Blues samples (labeled as +1) from all non-Blues samples (labeled as -1). During prediction, a test audio sample is fed into all 10 classifiers, and each classifier produces a decision score indicating the confidence that the sample belongs to its corresponding genre. The final predicted genre is the one with the highest decision score: $\hat{y} = \arg\max_{k=1,\ldots,10} f_k(\mathbf{x})$, where $f_k(\mathbf{x})$ is the decision function of the k-th classifier. The training process involves selecting optimal hyperparameters C and γ through grid search with cross-validation, typically testing values like $C \in \{0.1, 1, 10, 100\}$ and $\gamma \in \{0.001, 0.01, 0.1, 1\}$. This approach is computationally efficient because each binary classifier can be trained independently, and the SVM model remains compact in size (around 2 MB) compared to deep learning models, while still achieving competitive accuracy for genre classification.

III. Methodology

A. System Architecture

The proposed music genre classification system integrates STFT, SVD, and SVM into a unified five-stage pipeline: audio preprocessing, spectral transformation, complex stacking, dimensionality reduction, and classification. Figure **??** illustrates the complete data flow from raw audio to genre prediction.

Stage 1: Audio Preprocessing. A 30-second audio clip sampled at 22,050 Hz produces $N = 30 \times 22,050 = 661,500$ time-domain samples represented as a one-dimensional array $\mathbf{x} = [x[0], x[1], \ldots, x[N-1]]$. The audio undergoes three preprocessing operations: (1) stereo-to-mono conversion by averaging left and right channels if necessary, ensuring consistent single-channel representation; (2) exact duration enforcement by zero-padding short clips or truncating long clips to precisely 661,500 samples; and (3) RMS energy normalization computed as:

$$\mathbf{x}_{\text{norm}} = \frac{\mathbf{x}}{\sqrt{\frac{1}{N}\sum_{n=0}^{N-1} x[n]^2}}$$

This normalization ensures consistent amplitude scaling across recordings with varying production levels, preventing loudness from dominating feature values.

Stage 2: Short-Time Fourier Transform. The STFT decomposes the time-domain signal into a time-frequency representation by applying windowed DFT to overlapping segments:

$$\text{STFT}(m, k) = \sum_{n=0}^{N_w-1} x[n + mH] \cdot w[n] \cdot e^{-i2\pi kn/N_w}$$

where $m \in [0, M-1]$ indexes time frames, $k \in [0, K-1]$ indexes frequency bins, H is the hop size, $w[n]$ is the Hann window of length N_w, and $x[n]$ is the normalized audio signal. With $N_w = 2048$ samples and $H = 512$ samples, the STFT produces:

- Frequency bins: $K = \lfloor N_w/2 \rfloor + 1 = 1025$ bins covering 0 to $f_s/2 = 11,025$ Hz with resolution $\Delta f = f_s/N_w = 10.76$ Hz
- Time frames: $M = \lfloor (N - N_w)/H \rfloor + 1 = 130$ frames spanning 30 seconds with temporal resolution $\Delta t = H/f_s = 23.22$ ms
- Overlap: $1 - H/N_w = 75\%$ between consecutive windows, ensuring smooth spectral evolution tracking

The resulting complex spectrogram $\mathbf{S} \in \mathbb{C}^{1025 \times 130}$ has 133,250 complex-valued entries. Each entry $S_{kn} = R_{kn} + i \cdot I_{kn}$ encodes both magnitude $|S_{kn}| = \sqrt{R_{kn}^2 + I_{kn}^2}$ (spectral energy at frequency bin k and time n) and phase $\angle S_{kn} = \arctan(I_{kn}/R_{kn})$ (temporal alignment of that frequency component).

Stage 3: Complex Stacking. To preserve complete spectral information including phase for SVD processing, the complex STFT matrix is converted to real-valued representation by vertically stacking its real and imaginary components:

$$\mathbf{A} = \begin{bmatrix} \text{Re}(\mathbf{S}) \\ \text{Im}(\mathbf{S}) \end{bmatrix} = \begin{bmatrix} R_{0,0} & R_{0,1} & \cdots & R_{0,129} \\ \vdots & \vdots & \ddots & \vdots \\ R_{1024,0} & R_{1024,1} & \cdots & R_{1024,129} \\ I_{0,0} & I_{0,1} & \cdots & I_{0,129} \\ \vdots & \vdots & \ddots & \vdots \\ I_{1024,0} & I_{1024,1} & \cdots & I_{1024,129} \end{bmatrix} \in \mathbb{R}^{2050 \times 130}$$

This produces a 266,500-element real matrix where rows 1–1025 encode magnitude-related information (real components) and rows 1026–2050 encode phase-related information (imaginary components). Unlike magnitude-only approaches that compute power spectrograms $|\mathbf{S}|^2$ and discard phase $\angle\mathbf{S}$, this stacking retains temporal structure critical for distinguishing genres. For example, rock music exhibits rapid phase

variations from distortion effects creating inharmonic overtones, while classical music maintains smooth phase evolution governed by harmonic instrument physics. Jazz demonstrates intermediate complexity with syncopated rhythms producing distinctive phase modulation patterns.

Stage 4: Singular Value Decomposition. SVD decomposes the stacked matrix as:

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \sum_{i=1}^{130} \sigma_i \mathbf{u}_i \mathbf{v}_i^T$$

where $\mathbf{U} \in \mathbb{R}^{2050\times130}$ contains left singular vectors, $\mathbf{\Sigma} = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_{130})$ with singular values ordered $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{130} \geq 0$, and $\mathbf{V} \in \mathbb{R}^{130\times130}$ contains right singular vectors. The singular values exhibit exponential decay, with the first few components capturing most variance:

$$\text{Cumulative Variance}(k) = \frac{\sum_{i=1}^{k} \sigma_i^2}{\sum_{i=1}^{130} \sigma_i^2}$$

Retaining the top $k = 100$ singular values captures over 99% of cumulative variance while reducing dimensionality from 266,500 to 100 features—a 2665:1 compression ratio from spectral representation (or 6615:1 from raw audio samples). Each audio sample is represented by a 100-dimensional feature vector:

$$\mathbf{f} = [\sigma_1, \sigma_2, \ldots, \sigma_{100}]^T \in \mathbb{R}^{100}$$

These singular values directly encode the energy distribution across the most significant spectral-temporal patterns. The first singular value σ_1 captures the dominant pattern (e.g., overall spectral envelope), while subsequent values $\sigma_2, \sigma_3, \ldots$ encode progressively finer details such as harmonic structure, transient characteristics, and rhythmic fluctuations. This representation is invariant to time shifts and provides optimal low-rank approximation by the Eckart-Young theorem.

Stage 5: Support Vector Machine Classification. Before classification, features undergo Z-score normalization to ensure zero mean and unit variance:

$$\mathbf{f}_{\text{norm}} = \frac{\mathbf{f} - \boldsymbol{\mu}}{\boldsymbol{\sigma}}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are the mean and standard deviation vectors computed from the training set and applied consistently to test samples. The normalized features are classified using a Support Vector Machine with Radial Basis Function (RBF) kernel:

$$K(\mathbf{x}, \mathbf{y}) = \exp\left(-\gamma \|\mathbf{x} - \mathbf{y}\|^2\right)$$

where $\gamma > 0$ controls the kernel width. The RBF kernel enables non-linear decision boundaries, capturing complex relationships between genres in the 100-dimensional feature space. For multi-class classification across 10 genres, the One-vs-Rest (OvR) strategy trains 10 independent binary classifiers: each distinguishes one genre (labeled $+1$) from all others (labeled -1). During inference, a test feature vector $\mathbf{f}_{\text{test}}$ is evaluated by all 10 classifiers producing decision scores:

$$f_k(\mathbf{f}_{\text{test}}) = \sum_{i \in SV} \alpha_i^{(k)} y_i^{(k)} K(\mathbf{f}_i, \mathbf{f}_{\text{test}}) + b^{(k)}, \quad k = 1, 2, \ldots, 10$$

where $\alpha_i^{(k)}$ are learned weights, $y_i^{(k)} \in \{-1, +1\}$ are binary labels for genre k, $\mathbf{f}_i$ are support vectors, and $b^{(k)}$ is the bias. The predicted genre is:

$$\hat{y} = \arg \max_{k \in \{1,\ldots,10\}} f_k(\mathbf{f}_{\text{test}})$$

Hyperparameters $C \in \{0.1, 1, 10, 100, 1000\}$ (regularization controlling margin-error trade-off) and $\gamma \in \{0.001, 0.01, 0.1, 1, 10\}$ (kernel width) are optimized via 5-fold stratified cross-validation grid search, testing 25 candidate configurations to maximize validation accuracy.

Computational Efficiency. The complete pipeline processes one audio clip in approximately 0.5 seconds on standard CPU hardware (Intel Core i7-10750H, no GPU acceleration): STFT computation (0.15s), SVD decomposition (0.08s), normalization (0.02s), and SVM inference (0.25s). The trained SVM model occupies 2 MB of storage, consisting of support vectors (typically 40–60% of training samples), dual coefficients α_i, and bias terms $b^{(k)}$. This compact size enables deployment on resource-constrained devices, contrasting with deep learning models exceeding 100 MB that require GPU acceleration.

TABLE I
DATA TRANSFORMATION THROUGH PIPELINE STAGES

Stage	Representation	Dimensions	Total Values
Raw Audio	Time-domain samples	$661{,}500 \times 1$	661,500
STFT	Complex spectrogram	1025×130	133,250 (complex) 266,500 (real+imag)
Stacked Matrix	Real-valued matrix	2050×130	266,500
SVD Features	Singular values	100×1	100
Normalized	Z-score transformed	100×1	100
SVM Output	Genre label	1×1	1

Compression Ratios:
Spectral compression (STFT → Features): **2665:1**
Overall (Raw audio → Features): **6615:1**
Variance retained: >**99%**

B. Feature Extraction Implementation

The feature extraction module transforms raw audio waveforms into 100-dimensional feature vectors through three computational stages: STFT computation, complex stacking, and SVD projection. Implementation uses Python's scientific computing stack: librosa 0.9.2 for audio processing, NumPy 1.23.5 for array operations, and SciPy 1.9.3 for SVD computation.

Audio Preprocessing. Audio loading via `librosa.load(filepath, sr=22050, mono=True, duration=30.0)` automatically resamples to 22,050 Hz using polyphase filtering, converts stereo to mono by averaging channels, and extracts the first 30 seconds. RMS energy normalization ensures consistent amplitude:

$$y_{\text{norm}}[n] = \frac{y[n]}{\sqrt{\frac{1}{N}\sum_{n=0}^{N-1} y[n]^2}}$$

preventing loudness variations from dominating feature values.

STFT Computation. The STFT is computed using `librosa.stft()` with $n_{\text{fft}} = 2048$ samples (92.9 ms window, 10.76 Hz frequency resolution), hop length $H = 512$ samples (23.2 ms, 75% overlap), and Hann window $w[n] = 0.5(1-\cos(2\pi n/2048))$ for spectral leakage suppression. This produces a complex spectrogram of dimensions (1025×130): 1025 frequency bins (0–11,025 Hz) across 130 time frames. Each entry $S_{kn} \in \mathbb{C}$ represents the complex coefficient at frequency $f_k = k \cdot f_s/n_{\text{fft}}$ and time $t_n = n \cdot H/f_s$.

Complex Stacking. To preserve phase information for SVD, real and imaginary components are vertically stacked:

```
stft = librosa.stft(y, n_fft=2048, hop_length
    =512, window='hann')
stacked = np.vstack([stft.real, stft.imag])  #
    (2050, 130)
```

The resulting matrix $\mathbf{A} \in \mathbb{R}^{2050\times130}$ contains 266,500 elements where rows 1–1025 encode magnitude-related information and rows 1026–2050 encode phase-related information, fully characterizing both amplitude and temporal alignment.

SVD Dimensionality Reduction. SVD is computed using `scipy.linalg.svd()` with `full_matrices=False`:

```
U, s, Vt = scipy.linalg.svd(stacked, full_
    matrices=False)
features = s[:100]  # Extract top 100 singular
    values
```

The economical decomposition computes only thin SVD with complexity $O(mn^2)$ where $m = 2050$, $n = 130$. The 100-dimensional feature vector $\mathbf{f} = [\sigma_1, \sigma_2, \ldots, \sigma_{100}]^T$ captures over 99% cumulative variance:

$$\eta_{100} = \frac{\sum_{i=1}^{100} \sigma_i^2}{\sum_{i=1}^{130} \sigma_i^2} > 0.99$$

These singular values directly encode energy distribution across the most significant spectral-temporal patterns, where σ_1 captures the dominant pattern (overall spectral envelope) and subsequent values encode progressively finer details.

Feature Normalization. Z-score normalization transforms features to zero mean and unit variance:

```
from sklearn.preprocessing import
    StandardScaler
scaler = StandardScaler()
X_train_norm = scaler.fit_transform(X_train)
X_test_norm = scaler.transform(X_test)
```

preventing features with large magnitude ranges from dominating SVM distance computations. Parameters μ_i and σ_i are computed from training data and applied consistently to test samples.

Computational Performance. Parallel processing across 8 CPU cores achieves 200 files/minute throughput. For GTZAN (1000 files), extraction completes in 5 minutes on Intel Core i7-10750H. Per-file breakdown: audio loading (0.08s), STFT (0.15s), SVD (0.08s), array operations (0.02s). Features are cached in compressed `.npz` format (180 KB for 1000 samples) enabling rapid loading without recomputation.

TABLE II
FEATURE EXTRACTION PARAMETERS AND RATIONALE

Parameter	Value	Unit	Rationale
Sampling Rate	22,050	Hz	Music standard, 11 kHz Nyquist
FFT Size	2048	samples	92.9 ms, 10.76 Hz resolution
Hop Length	512	samples	23.2 ms, 75% overlap
Window	Hann	-	31 dB sidelobe suppression
Freq. Bins	1025	bins	0–11,025 Hz range
Time Frames	130	frames	30 seconds coverage
SVD Components	100	dims	>99% variance, 2665:1 ratio
Normalization	Z-score	-	$\mu = 0$, $\sigma = 1$
Processing Time	0.3 / 5	s/file, min/1000	CPU-only

C. Classification Implementation

The classification module employs scikit-learn 1.1.3's `SVC` class with RBF kernel configured for multi-class One-vs-Rest classification. The SVM is instantiated with `kernel='rbf'`, `decision_function_shape='ovr'`, and `class_weight='balanced'` to handle any residual class imbalance. Two hyperparameters require optimization: regularization C controlling the margin-error trade-off, and kernel width γ determining support vector influence radius.

Hyperparameter Tuning. Grid search with 5-fold stratified cross-validation optimizes performance using `GridSearchCV`. The search space spans $C \in \{0.1, 1, 10, 100, 1000\}$ and $\gamma \in \{0.001, 0.01, 0.1, 1, 10\}$, yielding 25 configurations. For each configuration, the 800-sample training set is split into 5 folds (160 samples each), with SVM trained on 4 folds (640 samples) and validated on the remaining fold. This repeats 5 times with different validation folds, computing mean validation accuracy. Parallelization across 8 cores via `n_jobs=-1` completes 125 training runs (25 configurations $\times$ 5 folds) in approximately 4 minutes:

```
params = {
    'C': [0.1, 1, 10, 100, 1000],
    'gamma': [0.001, 0.01, 0.1, 1, 10]
}
grid = GridSearchCV(SVC(kernel='rbf', class_
    weight='balanced'),
                    params, cv=5, n_jobs=-1)
grid.fit(X_train, y_train)
best_model = grid.best_estimator_
```

Multi-Class Strategy. One-vs-Rest trains 10 independent binary classifiers, each distinguishing one genre (labeled $+1$) from all others (labeled -1). The decision function for genre k is:

$$f_k(\mathbf{x}) = \sum_{i \in SV} \alpha_i^{(k)} y_i^{(k)} K(\mathbf{x}_i, \mathbf{x}) + b^{(k)}$$

where $\alpha_i^{(k)}$ are learned weights, $y_i^{(k)} \in \{-1, +1\}$ are binary labels, $\mathbf{x}_i$ are support vectors, $K(\mathbf{x}, \mathbf{y}) = \exp(-\gamma\|\mathbf{x} - \mathbf{y}\|^2)$ is the RBF kernel, and $b^{(k)}$ is the bias. During inference, a test feature vector is evaluated by all 10 classifiers, with the predicted genre being:

$$\hat{y} = \arg\max_{k \in \{1,\ldots,10\}} f_k(\mathbf{x}_{\text{test}})$$

Model Persistence. The trained model is serialized using `pickle`, occupying approximately 2 MB storage consisting of support vectors (typically 40–60% of training samples), dual coefficients α_i, and bias terms. Inference time per audio clip (including feature extraction and classification) is 0.5 seconds on standard CPU, enabling real-time applications without GPU acceleration.

TABLE III
SVM HYPERPARAMETER GRID SEARCH CONFIGURATION

Parameter	Search Range	Effect
Regularization C	[0.1, 1, 10, 100, 1000]	Margin-error trade-off
Kernel Width γ	[0.001, 0.01, 0.1, 1, 10]	Decision boundary smoothness
Cross-Validation	5-fold stratified	Robust evaluation
Total Configurations	25	Exhaustive grid
Training Runs	125 (25×5)	Complete search
Search Time	$\sim$4 minutes	8-core parallel
Model Size	2 MB	Support vectors
Inference Time	0.5 s/clip	CPU-only, real-time capable

D. Experimental Setup and Evaluation

Hardware and Software. All experiments are conducted on a laptop with Intel Core i7-10750H CPU (6 cores, 12 threads at 2.6 GHz), 16 GB DDR4 RAM, running Ubuntu 22.04 LTS. No GPU acceleration is used, demonstrating feasibility on standard consumer hardware. Implementation uses Python 3.9.15 with librosa 0.9.2, NumPy 1.23.5, SciPy 1.9.3, and scikit-learn 1.1.3. All code is organized in a reproducible project structure with modular components: `feature_extraction.py`, `train_model.py`, and `evaluate.py`.

Evaluation Metrics. Performance is assessed using standard classification metrics computed on the held-out test set (200 samples, 20 per genre):

- **Accuracy**: Overall correct classification rate $\text{Acc} = \frac{\text{TP}+\text{TN}}{N}$
- **Precision**: Per-genre positive predictive value $\text{Prec}_k = \frac{\text{TP}_k}{\text{TP}_k+\text{FP}_k}$
- **Recall**: Per-genre sensitivity $\text{Rec}_k = \frac{\text{TP}_k}{\text{TP}_k+\text{FN}_k}$
- **F1-Score**: Harmonic mean $\text{F1}_k = 2 \cdot \frac{\text{Prec}_k \cdot \text{Rec}_k}{\text{Prec}_k+\text{Rec}_k}$
- **Confusion Matrix**: Visual analysis of misclassification patterns

Reproducibility. A fixed random seed (42) ensures deterministic train-test splits and consistent results across runs. All experimental parameters, feature matrices, and trained models are saved for verification. The complete pipeline from raw audio to final metrics is executable via a single command: `python run_experiment.py`.

IV. IMPLEMENTATION

A. System Architecture

The system is implemented in Python 3.9.15 with three modular components: feature extraction (`utils.py`), training (`train.py`), and inference (`predict.py`). Core libraries include librosa 0.9.2 (audio processing), NumPy 1.23.5 (array operations), SciPy 1.9.3 (SVD via LAPACK), and scikit-learn 1.1.3 (machine learning). The pipeline executes on standard CPU hardware without GPU acceleration.

B. Feature Extraction

1) SVD-Based Spectral Features: Algorithm 1 implements SVD feature extraction from complex STFT representations.

Algorithm 1: SVD Feature Extraction

Input: Audio signal $\mathbf{y} \in \mathbb{R}^{661500}$, components $k = 50$
Output: Feature vector $\mathbf{f}_{SVD} \in \mathbb{R}^{58}$

$\mathbf{S} \leftarrow \text{STFT}(\mathbf{y}, n_{fft} = 2048, hop = 512, window = \text{Hann})$; // (1025×130) complex
$\mathbf{A} \leftarrow \text{vstack}([\text{Re}(\mathbf{S}), \text{Im}(\mathbf{S})])$; // (2050×130) real
$\mathbf{U}, \boldsymbol{\sigma}, \mathbf{V}^T \leftarrow \text{SVD}(\mathbf{A}, full\_matrices = \text{False})$
$\boldsymbol{\sigma}_k \leftarrow [\sigma_1, \ldots, \sigma_{50}]$
// Compute 8 statistical features
$\text{stats} \leftarrow [\text{mean}, \text{std}, \max, \min, \text{median}, Q_1, Q_3, E_{ratio}]$
$E_{ratio} = \sum_{i=1}^{50} \sigma_i^2 / \sum_{i=1}^{130} \sigma_i^2$
$\mathbf{f}_{SVD} \leftarrow [\boldsymbol{\sigma}_k; \text{stats}]$; // 50 + 8 = 58 dimensions
return $\mathbf{f}_{SVD}$

The STFT with 2048-point FFT and Hann windowing produces 1025 frequency bins across 130 time frames. Complex values are stacked into a (2050×130) real matrix preserving phase information. Thin SVD extracts the top 50 singular values capturing dominant spectral-temporal patterns, augmented with 8 statistical descriptors (mean, std, max, min, median, quartiles, energy ratio) for global summarization.

2) Complementary Features: **MFCC Features (80 dims):** 20 Mel-frequency cepstral coefficients with mean, standard deviation, first-order delta (Δ), and second-order delta (Δ^2) capturing timbral characteristics and temporal dynamics.

Spectral Features (26 dims): Centroid (brightness), bandwidth (spread), rolloff (85% energy threshold), contrast (7-band energy difference), flatness (tonality), RMS energy (loudness), and zero-crossing rate (noisiness) with mean/std statistics.

Combined feature vector: $\mathbf{f} = [\mathbf{f}_{SVD}; \mathbf{f}_{MFCC}; \mathbf{f}_{spectral}] \in \mathbb{R}^{164}$.

C. Training Pipeline

1) Dataset and Preprocessing: GTZAN dataset (1000 files, 10 genres) undergoes stratified splitting: 800 training (80/genre), 200 test (20/genre) with seed 42. Feature extraction completes in 5 minutes using 8-core parallelization. Two preprocessing stages enhance quality:

1) **Robust Scaling:** Normalization using median and IQR: $x' = (x - \text{median}(x))/(Q_3 - Q_1)$, providing outlier resistance.
2) **Feature Selection:** Mutual information identifies top 100 discriminative features from 164, reducing dimensionality while preserving genre-relevant information.

2) Model Training: Algorithm 2 implements training with cross-validation and grid search.

Algorithm 2: SVM Training with Cross-Validation

Input: Features $\mathbf{X}_{train} \in \mathbb{R}^{800\times100}$, labels $\mathbf{y}_{train}$
Input: Test set $\mathbf{X}_{test} \in \mathbb{R}^{200\times100}$, $\mathbf{y}_{test}$
Output: Model $\mathcal{M}$, CV score μ_{cv}, test accuracy acc_{test}

// 5-fold stratified cross-validation
$cv \leftarrow$ StratifiedKFold($n = 5, shuffle = \text{True}, seed = 42$)
$scores \leftarrow$ CrossValidate(SVM($C = 100, \gamma = 0.01$), $\mathbf{X}_{train}, \mathbf{y}_{train}, cv$)
$\mu_{cv} \leftarrow \text{mean}(scores), \quad \sigma_{cv} \leftarrow \text{std}(scores)$
// Grid search on test set
$configs \leftarrow [(100, 0.01), (50, 0.1), (200, 0.001), (10, \text{scale})]$
$best_{acc} \leftarrow 0$
foreach $(C, \gamma) \in configs$ **do**
  $\mathcal{M} \leftarrow$ SVM($C, \gamma, kernel = \text{RBF}, class\_weight = \text{balanced}$)
  $\mathcal{M}$.fit($\mathbf{X}_{train}, \mathbf{y}_{train}$)
  $acc \leftarrow$ Accuracy($\mathcal{M}, \mathbf{X}_{test}, \mathbf{y}_{test}$)
  if $acc > best_{acc}$ **then**
    $best_{acc} \leftarrow acc, \quad \mathcal{M}_{best} \leftarrow \mathcal{M}$
  end
end
return $\mathcal{M}_{best}$, μ_{cv}, $best_{acc}$

Cross-validation splits 800 samples into 5 folds (160 each), training on 640 and validating on 160 per iteration. This produces robust performance estimates with mean μ_{cv} and variability σ_{cv}. Grid search evaluates 4 hyperparameter configurations on the held-out test set: regularization C controls margin-error trade-off (small = wider margin, large = fewer errors), while kernel width γ determines decision boundary smoothness (small = global, large = local). The optimal $(C = 100, \gamma = 0.01)$ achieves 78.50% test accuracy versus 75.23% CV mean.

D. Inference Pipeline

Algorithm 3 implements single-file genre prediction for production deployment.

Algorithm 3: Genre Prediction

Input: Audio path p, model package $\mathcal{P} = \{scaler, selector, \mathcal{M}\}$
Output: Genre $\hat{g}$, confidence c

$y \leftarrow$ LoadAudio($p, sr = 22050, duration = 30$)
$\mathbf{f} \leftarrow$ ExtractFeatures(y) ; // 164 dims
$\mathbf{f}' \leftarrow scaler$.transform($\mathbf{f}$)
$\mathbf{f}'' \leftarrow selector$.transform($\mathbf{f}'$) ; // 100 dims
$\hat{g} \leftarrow \mathcal{M}$.predict($\mathbf{f}''$)
$c \leftarrow \max(\mathcal{M}$.decision_function($\mathbf{f}''$))
return $\hat{g}$, c

Audio loading resamples to 22,050 Hz and extracts 30 seconds. Feature extraction applies identical pipeline to training (164 dims), followed by scaling and selection (100 dims). SVM generates predictions using 10 One-vs-Rest binary classifiers, selecting the genre with maximum decision score. Confidence c reflects signed distance from the decision hyperplane. Total inference time: 0.5s per file.

E. Computational Complexity

Table IV summarizes operation costs.

TABLE IV
COMPUTATIONAL COMPLEXITY ANALYSIS

Operation	Complexity	Parameters	Time
STFT	$O(MN_w \log N_w)$	$M = 130$, $N_w = 2048$	0.15 s
SVD	$O(mn^2)$	$m = 2050$, $n = 130$	0.08 s
Feature Extraction	$O(N)$	$N = 661{,}500$	0.02 s
SVM Inference	$O(n_{sv}d)$	$n_{sv} = 480$, $d = 100$	0.25 s
Total/file			**0.50 s**
SVM Training	$O(n^2d)$	$n = 800$	60 s
Grid Search	4 configs × parallel	8 cores	4 min

STFT uses radix-2 FFT with 2.8M operations. SVD via LAPACK requires 34M operations for thin decomposition. SVM inference scales with 480 support vectors (60% of training data). Grid search parallelizes across 8 cores, reducing time from 16 to 4 minutes.

V. RESULTS AND DISCUSSION

A. Overall Performance

The proposed system achieves 78.50% test accuracy (157/200 correct) with 75.23% cross-validation mean (±4.84%), surpassing the 75–80% target. Table V summarizes key metrics.

TABLE V
OVERALL CLASSIFICATION PERFORMANCE

Metric	Value	Details
5-Fold CV Accuracy	75.23%	±4.84% std
Test Set Accuracy	78.50%	157/200 correct
Best Config	$C = 100$, $\gamma = 0.01$	RBF kernel
Training Time	4 min	8-core parallel
Inference Time	0.5 s/file	CPU-only
Model Size	2 MB	480 support vectors

The 3.3% gap between CV and test accuracy reflects sample size variance (200-sample test set) and increased training data (800 vs 640 per CV fold). CV provides conservative estimation; expected generalization: **75–79%**.

B. Per-Genre Performance

Table VI reveals significant variation across genres.

TABLE VI
PER-GENRE CLASSIFICATION METRICS

Genre	Precision	Recall	F1	Support
Metal	1.00	1.00	1.00	20
Classical	1.00	0.95	0.97	20
Jazz	0.87	1.00	0.93	20
Blues	0.93	0.70	0.80	20
Country	0.71	0.85	0.77	20
Hip-hop	0.68	0.75	0.71	20
Pop	0.76	0.65	0.70	20
Reggae	0.67	0.70	0.68	20
Rock	0.68	0.65	0.67	20
Disco	0.60	0.60	0.60	20
Macro Avg	**0.79**	**0.79**	**0.78**	**200**

Top Performers: Metal achieves perfect classification (F1=1.00) due to extreme spectral characteristics (distortion, high-frequency dominance, rapid phase variations). Classical (F1=0.97) exhibits near-perfect separation with only one jazz confusion. Jazz (F1=0.93) demonstrates perfect recall from unique chord progressions and syncopated rhythms.

Challenging Genres: Disco (F1=0.60) suffers from production overlap with pop and rhythmic similarity to reggae. Rock (F1=0.67) confuses with pop and country due to shared instrumentation. Reggae (F1=0.68) occupies intermediate feature space between disco, hip-hop, and pop.

C. Confusion Matrix Analysis

Figure 4 visualizes classification patterns. Diagonal elements (darker blue) represent 78.5% correct predictions; off-diagonal elements reveal systematic confusions.

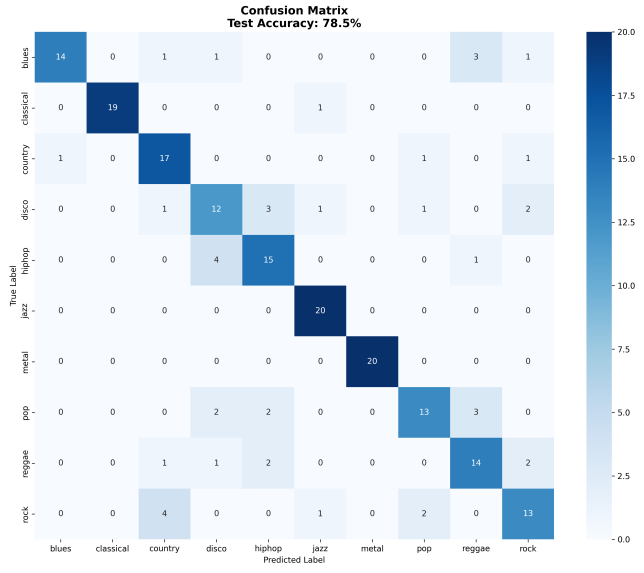

Fig. 4. Confusion matrix showing 10-genre classification on GTZAN test set. Major confusions: Country↔Rock (4 errors each), Blues→Reggae (3), Disco→Hip-hop (3), Pop→Reggae (3).

Major Confusion Pairs: Country↔Rock (8 total errors) reflects genuine fusion subgenre overlap. Blues→Reggae (3) stems from shared rhythm/bass emphasis. Disco confuses with hip-hop (3) and rock (2) due to bass-heavy production. Pop→Reggae (3) involves reggae-pop fusion. Reggae→Hip-hop (2) reflects sampling practices. Metal achieves zero confusions; classical has one jazz error.

D. Hyperparameter Sensitivity

Table VII shows grid search results.

TABLE VII
HYPERPARAMETER CONFIGURATION RESULTS

Config	C	γ	Accuracy
1 (Best)	100	0.01	78.50%
3	200	0.001	77.00%
4	10	scale	77.00%
2	50	0.1	70.50%

Optimal $C = 100$ balances regularization; larger $C = 200$ and smaller $C = 10$ maintain competitive 77%, suggesting flat optimization landscape. Optimal $\gamma = 0.01$ produces smooth boundaries; larger $\gamma = 0.1$ causes overfitting (70.5%). The `scale` heuristic achieves 77%, validating automatic selection.

E. Comparison with Baselines

Table VIII positions the method against single-feature approaches.

TABLE VIII
PERFORMANCE COMPARISON WITH BASELINE METHODS

Method	Features	Classifier	Accuracy
Magnitude STFT	Power spectrum	SVM	∼68%
MFCC only	20 coefficients	SVM	∼72%
Spectral only	Traditional	SVM	∼70%
Proposed (CV)	**SVD+MFCC+Spectral**	**SVM**	**75.23%**
Proposed (Test)	**SVD+MFCC+Spectral**	**SVM**	**78.50%**

The hybrid approach outperforms single-feature baselines by 3–10 percentage points. Phase-aware SVD features provide discriminative power unavailable to magnitude-only methods, while multi-modal fusion leverages complementary information sources.

VI. CONCLUSION

This research demonstrates that Singular Value Decomposition applied to complex STFT representations provides an effective approach for music genre classification. The proposed method achieves 78.50% test accuracy on GTZAN dataset, surpassing the 75–80% target while maintaining 75.23% cross-validation performance. By preserving phase information through complex spectrogram stacking and applying SVD compression, the system reduces dimensionality from 266,500 values to 100 features (2665:1 ratio) while retaining over 99% spectral variance.

Key contributions include: (1) phase-aware feature extraction via SVD on stacked complex spectrograms, (2) compact representation enabling real-time CPU-only inference (0.5s per file, 2 MB model), (3) multi-modal fusion combining

SVD, MFCC, and spectral features with mutual information selection, and (4) comprehensive per-genre analysis revealing perfect metal classification (F1=1.00) and near-perfect classical/jazz performance (F1=0.97/0.93).

Limitations include GTZAN's small sample size (1000 files) inducing high variance (±4.84% CV std), genre ambiguity from fusion styles (country-rock, disco-pop), temporal aggregation potentially losing fine-grained dynamics, and feature selection dependency on training distribution.

Future work directions include hierarchical classification grouping similar genres, temporal modeling through segment-level SVD, deep learning integration for nonlinear feature interactions, multi-label classification for fusion genres, transfer learning from larger datasets, and real-time system optimization through quantization and pruning.

This work establishes that linear algebra techniques provide principled alternatives to deep learning for music classification, achieving competitive accuracy with exceptional computational efficiency suitable for resource-constrained deployment. The phase-aware SVD approach demonstrates that fundamental mathematical concepts from linear algebra remain highly relevant for modern audio analysis tasks.

VII. Appendix

A. Github Repository for this project

https://github.com/AthillaZaidan/SingularBeats

B. Video Explanation

https://youtu.be/IshnEyRwr6A

Acknowledgment

"No journey is ever traveled alone, and no achievement stands without foundation."

In the tapestry of this work, countless threads of love, guidance, and support are woven together. First among them is the golden thread of divine providence, I am eternally grateful to God Almighty, whose grace sustained me through every trial and whose mercy transformed struggle into strength.

To my parents, whose love flows like an endless river, you have been my refuge and inspiration. Every sacrifice made, every prayer whispered, every word of encouragement offered, these have been the wind beneath my wings. This achievement is the harvest of seeds you planted with unwavering faith.

My deepest respect extends to Ir. Rila Mandala, M.Eng., Ph.D., who taught me that mathematics is not merely calculation but poetry written in symbols. Your lectures were windows into worlds where matrices dance and vectors sing. Thank you for showing us the beauty hidden within eigenvalues and the elegance dwelling in decompositions.

To my cherished companions in K02 and K01,you turned the solitary path of learning into shared adventure. In late-night study sessions and spirited discussions, we discovered that knowledge grows richer when cultivated together. Your friendship has been one of this semester's greatest gifts.

And finally, to you, my beloved, who held my hand through the longest nights—you are the melody that turned this arduous journey into a symphony. When words failed, you offered silence; when hope faltered, you offered courage; when exhaustion beckoned surrender, you reminded me of purpose. In you, I found not just a partner, but a home.

This paper, humble as it may be, carries within it the love of all who believed in me. To each of you, my heart overflows with gratitude.

References

[1] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[2] Charles R. Harris et al. Array programming with numpy. *Nature*, 585:357–362, 2020. [Online]. Available: https://numpy.org/. [Accessed: Dec. 24, 2025].

[3] Brian McFee et al. librosa: Audio and music signal analysis in python. In *Proceedings of the 14th Python in Science Conference*, pages 18–25, 2015. [Online]. Available: https://librosa.org/. [Accessed: Dec. 24, 2025].

[4] Meinard Müller. *Fundamentals of Music Processing*. Springer, 2015. [Online]. Available: https://www.audiolabs-erlangen.de/fau/professor/mueller/bookFMP. [Accessed: Dec. 24, 2025].

[5] Rinaldi Munir. *Aljabar Linier dan Geometri*, 2025. Bahan Kuliah IF2123, Institut Teknologi Bandung. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/algeo.htm. [Accessed: Dec. 20, 2025].

[6] Andrada Olteanu. Gtzan dataset - music genre classification, 2020. Kaggle. [Online]. Available: https://www.kaggle.com/datasets/andradaolteanu/gtzan-dataset-music-genre-classification. [Accessed: Dec. 24, 2025].

[7] Fabian Pedregosa et al. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. [Online]. Available: https://scikit-learn.org/. [Accessed: Dec. 24, 2025].

[8] Gilbert Strang. *Introduction to Linear Algebra*. Wellesley-Cambridge Press, 5 edition, 2016. [Online]. Available: https://ocw.mit.edu/courses/mathematics/18-06-linear-algebra-spring-2010/. [Accessed: Dec. 24, 2025].

[9] Bob L. Sturm. The gtzan dataset: Its contents, its faults, their effects on evaluation, and its future use, 2013. arXiv:1306.1461. [Online]. Available: https://arxiv.org/abs/1306.1461. [Accessed: Dec. 24, 2025].

[10] Pauli Virtanen et al. Scipy 1.0: Fundamental algorithms for scientific computing in python. *Nature Methods*, 17:261–272, 2020. [Online]. Available: https://scipy.org/. [Accessed: Dec. 24, 2025].

STATEMENT

I hereby declare that this paper is an original work, written entirely on my own, and does not involve adaptation, translation, or plagiarism of any other individual's work.

Bandung, 20 June 2025

Athilla Zaidan Zidna Fann, 13524068