Received Date:28th November, 2025
Revision Date: 28th February, 2026
Accepted Date: 15th March, 2026

Chord Classification Using Machine Learning

Rohan Bade[1], Susan Thapa[2], Risav Pokhrel[3], Sagar Bhandari[4], Sanjivan Satyal[5*]
[1]Dept of Electronics and Computer Engineering, Pulchowk Campus, TU, Nepal. Email: 077bei036.rohan@pcampus.edu.np
[2]Dept of Electronics and Computer Engineering, Pulchowk Campus, TU, Nepal. Email: 077bei046.susan@pcampus.edu.np
[3]Dept of Electronics and Computer Engineering, Pulchowk Campus, TU, Nepal. Email: 077bei035.risav@pcampus.edu.np
[4]Dept of Electronics and Computer Engineering, Pulchowk Campus, TU, Nepal. E-mail:
[5]Assistant Professor, Dept of Electronics and Computer Engineering, Pulchowk Campus, TU, Nepal.
Email: sanziwan.satyal@pcampus.edu.np

Abstract— Chord classification is a fundamental task that enables automated music transcription and analysis in Music Information Retrieval. This paper presents a comparative study of the machine learning approaches for chord recognition from audio signals. The proposed framework deals with two feature extraction methods– Pitch Class Profile (PCP) and spectrograms, which are evaluated across three classification models: Support Vector Machines (SVM), Multilayer Perceptron (MLP), and Convolutional Neural Networks (CNN), and these models are trained to classify 24 chord classes comprising 12 major and minor triads. Upon experiment, the result demonstrates that PCP- based feature extraction significantly out performs spectrogram-based methods, particularly with limited training data. The MLP classifier achieves the highest performance with 0.99 precision, recall, F1-score, and accuracy, though these metrics indicate potential overfitting. The SVM model yields 0.96 accuracy with better generalization capability on unseen data. The CNN model achieves up to 0.97 training accuracy but exhibits poor generalization. The finding reveals that with PCP-enabled method can be used effectively and develop a robust chord classification system. This work highlights the potential of machine learning in improving chord classification accuracy and sets the stage for future development by refining models and expanding the dataset.

Keywords — Audio signal processing, Chord classification, Convolutional neural networks, Feature extraction, Multilayer perceptron, Music information retrieval, Pitch class profile, Support vector machines

Introduction

Music is a universal language composed of melodies, rhythms, and harmonies. At the core of harmony are chords, which are formed by playing multiple notes simultaneously. Chords provide depth and emotion to a piece of music, making them essential for musicians, composers, and enthusiasts. Among various types of chords, triads, including major and minor chords, are the most fundamental and widely used in music. Identifying chords by ear, particularly for beginners or self-taught musicians who lack formal training in music theory, remains a significant challenge, especially with accuracy when dealing with complex harmonies, subtle variations, or unfamiliar musical styles. This challenge becomes even more pronounced when attempting to transcribe newly released songs or pieces from diverse musical genres. Traditional approaches to chord identification typically involve manual transcription, which is a time-consuming and error-prone process that requires musical expertise. Existing digital tools provide limited chord vocabularies or lack precision. In addition, accurate chord recognition remains a challenging task due to factors like instrumental variations, recording noise, overlapping harmonics, and diverse musical styles. Early computational systems addressed these issues using Pitch-Class Profiles (PCP) or chroma features, which summarize spectral information into twelve pitch classes to represent harmonic content efficiently [1]. Subsequent improvements in feature extraction, such as enhanced spectral smoothing and constant-Q transforms, further increased robustness to timbre and tuning variations. To model temporal continuity in music, Hidden Markov Models (HMMs) were then employed, enabling smoother and more context-aware chord transitions based on probabilistic state sequencing. In recent years, the field of machine learning has demonstrated remarkable potential in automating complex pat tern recognition tasks, including music analysis. For music analysis, we employed two approaches: Pitch Class Profile (PCP) and spectrogram features, to train three different models, Multilayer Perceptron (MLP), Support Vector Machine (SVM), and Convolutional Neural Network (CNN), separately. MLP and SVM utilized PCP as input features, whereas CNN used spectrogram features of

* Corresponding Author

audio clips. The objective of these models was to classify 24 fundamental major and minor chords. Our goal was to compare them across aspects such as accuracy, performance, and generalization to unseen music. To achieve this, we synthesized our dataset using FL Studio under various environments, including both noisy and quiet conditions [2]. These considerations enabled us to fairly juxtapose each method and ultimately integrate the best-performing approach into a mobile application.

Automatic chord classification has evolved from rule-based and statistical methods to modern deep learning approaches. Early systems employed the Pitch-Class Profile (PCP) representation [1] to map audio spectra into twelve pitch classes, forming the basis for most feature extraction pipelines. Improvements in chroma feature computation, including spectral smoothing and Constant-Q Transform (CQT) representations, enhanced robustness to tuning and timbral variations. Hidden Markov Models (HMMs) were later used to model temporal dependencies between chords, enabling smoother transitions and more musically consistent predictions. More recent works have leveraged Convolutional Neural Networks (CNNs) and hybrid CNN–RNN architectures to learn features directly from spectrograms, achieving state-of-the-art performance on benchmark datasets such as the Beatles and Isophonics collections.

A. *Our Proposed Approach on Chord Classification*

Despite the availability of multiple approaches to chord classification, in our assessment, our study serves as the first venture into comprehensive musical chord recognition using a two-phase approach that combines advanced signal processing and machine learning. Our work addresses the critical challenge of accurately identifying all 24 major and minor chords from raw audio and optimizing classification accuracy and model generalization across diverse instruments and acoustic conditions. In the first phase of our research, we developed a feature extraction pipeline trained on a custom dataset to transform raw audio signals into Pitch Class Profile (PCP) representations for chord classification. This data-driven approach allows for octave-independent representations, with a more informed and efficient harmonic characterization strategy, considering various factors such as frequency components, pitch class energy distributions, that capture the essential harmonic content regardless of loudness or recording quality. Building upon this feature extraction foundation, the second phase of our work focuses on the comprehensive classification of all 24 chord types using multiple machine learning architectures. Here, we employ three distinct classification models—Support Vector Machines (SVM), Multilayer Perceptron (MLP), and Convolutional Neural Networks (CNN)—to model and optimize the complex patterns within musical harmony during the classification process. This approach enables us to balance competing objectives such as maximizing classification accuracy, ensuring robust generalization to unseen data, and maintaining computational efficiency across the evolving landscape of multi-instrument audio inputs. By integrating advanced signal processing for harmonic feature extraction and machine learning for intelligent classification, our research contributes a novel methodology to the field of automated music analysis. This two-phase strategy not only addresses the immediate challenges of accurate chord recognition but also lays the groundwork for future music information retrieval applications by providing a flexible and adaptable framework for audio-to-musical-notation transformation. Our work aims to provide musicians, music learners, and educators with a data-driven approach to navigate the complexities of music transcription, potentially accelerating the adoption of automated music analysis technology while minimizing manual effort and maximizing transcription accuracy across diverse musical contexts.

I. Materials And Methods

In this section, we discuss our two-phased procedure for classification using feature extraction and machine learning. The workflow outlines the sequence of steps involved in the entire procedure from raw audio processing to final chord prediction. Phase I deals with feature extraction to transform raw audio signals into meaningful harmonic representations for chord classification. Initially, audio samples are collected and relevant acoustic features are extracted by processing the audio through a Discrete Fourier Transform (DFT) pipeline. The audio signal is converted from the time domain to the frequency domain using DFT, which decomposes the audio into its constituent frequency components. Each frequency bin from the DFT is then mapped to one of the 12 musical pitch classes (C, C#, D, D#, E, F, F#, G, G#, A, A#, B) using a logarithmic formula that determines which semitone each frequency corresponds to, regardless of the octave. The energy of all frequency bins belonging to the same pitch class is added together—for example, all frequencies corresponding to "C" notes (C1, C2, C3, etc.) are aggregated into a single bin representing the pitch class "C". This process is repeated for all 12 pitch classes, creating a raw

12-dimensional vector called PCP*, where each element represents the total energy of one pitch class. Finally, the vector is normalized by dividing each pitch class energy by the total energy across all 12 bins, ensuring that the PCP represents relative proportions rather than absolute loudness. The dataset is expanded through recordings from multiple instruments (guitar, piano, violin, accordion) and diverse acoustic conditions to reinforce it. This enhanced dataset is then utilized to train classification models, which identify the relationship between the PCP vectors and the corresponding chord labels. Phase II focuses on comprehensive chord classification, deploying multiple machine learning architectures to assess different classification approaches. At first, three distinct models are developed: Support Vector Machines (SVM) for efficient linear separation in high-dimensional PCP space, Multilayer Perceptron (MLP) for capturing non-linear relationships between pitch class distributions and chord types, and Convolutional Neural Networks (CNN) for learning hierarchical patterns from spectrogram representations. Now, comparative experiments are conducted across these three architectures with different hyperparameter configurations. During the training process, chronological behaviors of accuracy, loss, precision, and recall are observed while continuously updating model weights through backpropagation. Using the trained models, the chord classification readiness for different audio inputs is predicted. The predictions provide accurate chord labels for all 24 major and minor chord classes. Finally, the performance patterns and outcomes of each model are interpreted through confusion matrices, classification reports, and cross-validation results. This interpretation illustrates the optimal classification strategy, with SVM demonstrating superior generalization (96% accuracy) and MLP achieving the highest training performance (99% accuracy). Figure 1 summarizes both Phase I and Phase II of our work. Phase I contains the feature extraction part: audio pre-processing, DFT transformation, PCP calculation, and normalization. Phase II contains model development, training and validation, and performance evaluation for chord prediction. These two phases are linked by the normalized PCP vector, which, for Phase I, acts as the output representation of harmonic content and, for Phase II, acts as the input feature vector for machine learning models (the PCP vector serves as the bridge between acoustic signal processing and intelligent classification). The combination of advanced signal processing and ma chine learning in these two phases ensures an automated chord classification. By utilizing PCP-based feature extraction for harmonic representation and comparative machine learning analysis across multiple architectures, our approach handles both the acoustic complexity and classification accuracy aspects of the transition from raw audio to musical chord identification.

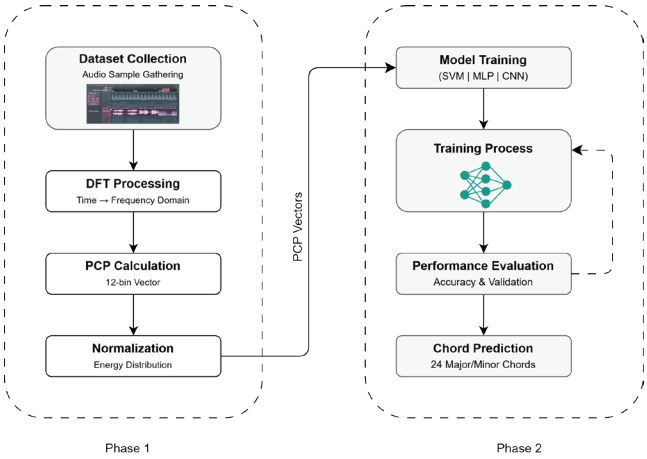

Fig. 1. Summarization of Phase I and Phase II of the chord classification framework.

A. *Dataset Generation*

The dataset used in this study was constructed from two primary sources. A subset of chord samples corresponding to ten chord classes namely A, Am, Bm, C, D, Dm, E, Em, F, and G was obtained from the publicly available collection curated by Jos Malsy [3], accessible at https://people. montefiore.uliege.be/josmalskyj/research.php. To expand the chord vocabulary and ensure balanced class representation, the remaining chord samples were generated using FL Studio, a digital audio workstation (DAW). Each chord was synthesized using a consistent virtual instrument and recording setup to maintain uniformity in timbre and volume across all samples. The generated audio files were then exported in WAV format and subsequently processed for feature extraction and model training.

B. *Data Preprocessing*

The preprocessing pipeline employed in this study follows established methodologies in music information retrieval (MIR) and adapts them for chord classification tasks. Two distinct preprocessing approaches were implemented to extract necessary features from audio files: spectrogram-based classification and Pitch Class Profile (PCP)-based classification. Each method involves specific computational steps optimized for different model architectures and feature representations.

1) *Spectrogram-Based Preprocessing*

The audio data preprocessing for spectrogram-based classification follows a multi-stage pipeline that transforms raw audio signals into time-frequency representations suitable for convolutional neural networks. It involves the steps:

Audio Loading and Signal Acquisition: The preprocessing pipeline begins with audio file loading using Librosa. The time-domain signal y(t) is extracted with its native sampling rate fs:

$$y(t) = \text{librosa.load(file\_path,sr = None)} \quad (1)$$

where y(t) represents the time-domain audio signal and fs denotes the sampling rate.

Time-Frequency Transformation: The Short-Time Fourier Transform (STFT) is applied to the time-domain signal to obtain its time-frequency representation:

$$\text{STFT}(y(t)) = X(t,f) = \int_{-\infty}^{\infty} y(\tau) \cdot w(t-\tau) \cdot e^{-j2\pi f\tau} d\tau \quad (2)$$

where w (t − τ) is the window function and X (t, f) denotes the complex-valued time-frequency bins. This is implemented as:

$$D = \text{librosa.stft(y)} \quad (3)$$

where D is the complex STFT matrix.

Logarithmic Amplitude Scaling: The magnitude spectrum is converted to decibel (dB) scale to enhance dynamic range and approximate human auditory perception:

$$SdB(t, f) = 20 \log10(|X(t, f)|) \quad (4)$$

where SdB(t, f) represents the decibel-scaled value and |X (t, f) | denotes the magnitude of the complex STFT value.

Feature Normalization: The decibel-scaled spectrogram undergoes min-max normalization to standardize the input range for neural network processing:

$$Snorm(t, f) = [SdB(t, f) - \min(SdB) / \max(SdB) - \min(SdB)] \times 255 \quad (5)$$

ensuring values reside within [0, 255], compatible with standard image processing conventions.

Spatial Standardization: The final stage involves resizing spectrograms through symmetric padding or center cropping to maintain consistent input dimensions. For undersized spectrograms:

$$Padwidth = targetwidth - currentwidth \quad (6)$$

with padding distributed as:

$$leftpad = | Padwidth / 2|$$

$$rightpad = Padwidth - leftpad \quad (7)$$

For oversized spectrograms, center cropping is performed:

$$Cropstart = | (currentwidth - targetwidth) / 2 | \quad (8)$$

The complete pipeline produces standardized spectrogram images capturing time-frequency characteristics essential for CNN-based chord recognition. The detailed implementation of the Algorithm for Spectrogram Feature Extraction Pipeline Algorithm 1.

Algorithm 1: Spectrogram Feature Extraction Pipeline

Input: Audio_file, target_size (1025 × 192)

Output: Preprocessed spectrogram Sf inal

1: y(t), fs ← Load audio signal from Audio_file

2: D ← STFT(y(t)) ▷ Apply Short-Time Fourier Transform

3: SdB ← 20 log10(|D|) ▷ Convert to decibel scale

4: Snorm ← MinMaxScale(SdB, range=[0, 255]) ▷ Normalize to [0, 255]

5: if width(Snorm) < target_width then ▷ Check spectrogram width

6: Sresized ← SymmetricPad(Snorm, target_size) ▷ Apply symmetric padding

7: else if width(Snorm) > target_width then

8: Sresized ← CenterCrop(Snorm, target_size) ▷ Apply center cropping

9: else

10: Sresized ← Snorm

11: end if

12: Sf inal ← Resize(Sresized, target_size) ▷ Final resize to target dimensions

13: return Sf inal

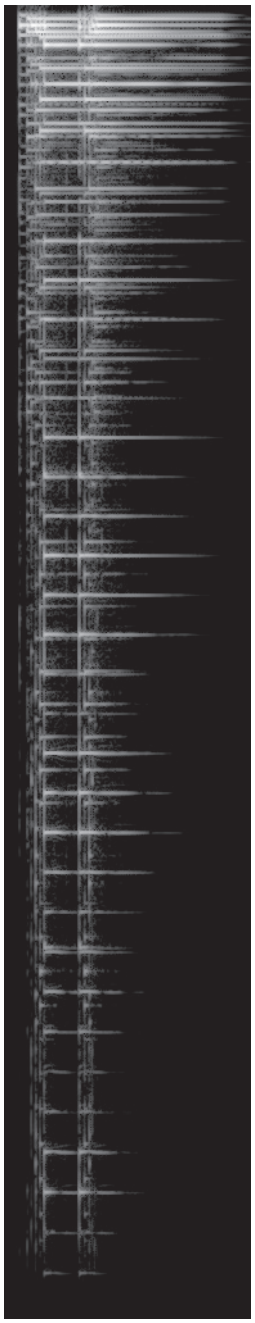

Fig. 2. Spectrogram Image of A minor Chord after Preprocessing

2) *PCP-Based Preprocessing*

The Pitch Class Profile (PCP) preprocessing pipeline extracts harmonic features representing the relative intensity of each semitone in the chromatic scale, providing a compact representation optimized for chord classification [1].

PCP Vector Computation: The PCP vector, denoted as PCP*(p), is derived from the Discrete Fourier Transform (DFT) spectrum using energy aggregation across pitch classes, following established methodologies in harmonic analysis [20]. The computational formulation is expressed

$$\mathrm{PCP}^*(p) = \sum_{l} |X(l)|^2 \delta(M(l), p) \tag{9}$$

where δ (·,·) represents the Kronecker delta function, and M(l) maps frequency bins to pitch classes according to the equal-tempered scale:

$$M(l) = \mathrm{round}\left(12 \log_2 \left(\frac{f_s \cdot l}{N \cdot f_{\mathrm{ref}}}\right)\right) \mod \tag{10}$$

where fs denotes the sampling frequency, N represents the number of DFT bins, and fref=130.81 Hz corresponds to the reference frequency of C3. This mapping ensures accurate assignment of spectral energy contributions to their respective pitch classes, forming a 12-dimensional PCP vector.

Energy Normalization: To mitigate variations in recording intensity and performance dynamics, the raw PCP vector undergoes energy normalization, emphasizing relative harmonic distributions over absolute amplitude values [2]. The normalization is computed as:

$$\mathrm{PCP}(p) = \frac{\mathrm{PCP}^*(p)}{\sum_{j=0}^{11} \mathrm{PCP}^*(j)} \tag{11}$$

This transformation ensures classification models focus on pitch class relationships rather than absolute energy levels, enhancing robustness to performance variations.

Feature Scaling: Min-Max scaling is applied to map PCP values to a [0, 1] range, preventing dominance by specific pitch classes and ensuring numerical stability during model training [11]. This preprocessing step aligns with standard feature scaling practices in machine learning and facilitates convergence during optimization.

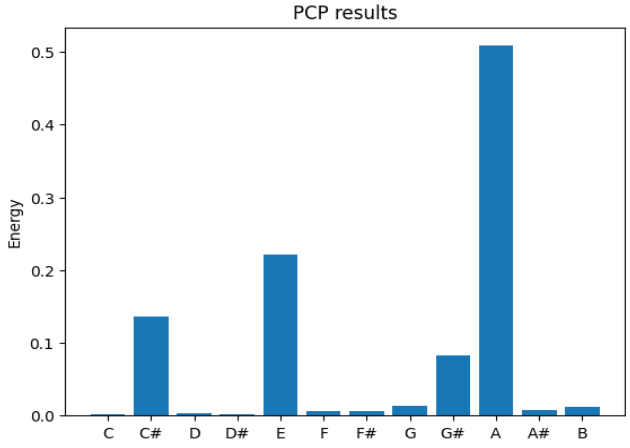

Fig. 3. PCP of an A major chord after Preprocessing

C. *Model Training*

This section delineates the specific pipeline and parameters for each of the three models: Support Vector Machine (SVM), Multilayer Perceptron (MLP), and Convolutional Neural Network (CNN), while assessing their unique audio analysis capabilities.

1) *Support Vector Machine*

Support Vector Machine (SVM) is a classic model that operates by creating decision boundaries through hyperplanes in the feature space, making it effective for distinguishing harmonic structures between different chords [3]. For our investigation, the input features comprised Pitch Class Profile (PCP) vectors extracted from audio clips, a feature vector established as effective for automatic chord recognition [1], [2]. The model was developed through a step-by-step process involving data preprocessing, model configuration, training, and evaluation. For data preparation, audio files from different sources were sliced into 1-second clips, from which PCP features were generated and stored in JSON files in a specific format. Each file contained a 12-element PCP vector and a 24- element one-hot encoded label corresponding to individual chord classes. To prepare the labels for SVM training, each sample's one-hot encoded vector was converted to a single class index using the argmax function. A linear kernel was selected due to its efficiency with high-dimensional data of moderate size [3]. The aggregated dataset was partitioned in an 80-20 ratio, with 80% used for training and the remaining 20% for evaluation. Model performance was evaluated using standard accuracy metrics and a confusion matrix to identify which of the 24 chords were commonly confused.

2) *Multilayer Perceptron*

Multilayer Perceptron (MLP) is recognized for its capability to model non-linear relationships between input variables, making it suitable for our analysis of PCP features. Consequently, it was selected as our second comparative model, following the precedent of using neural networks for chord recognition from PCP vectors [2]. We used the same aggregated JSON dataset as for the SVM, converting it into TensorFlow-compatible format before dividing it into three subsets: 80% for training, 10% for validation during training, and 10% for final testing. Batching with a batch size of 8 and prefetching were implemented to accelerate the training process. The MLP architecture consisted of an input layer with 12 neurons (for the PCP vector), three hidden layers with 256, 256, and 128 neurons respectively, and finally an output layer with 24 neurons corresponding to the 24 chord classes. Each hidden layer was followed by batch normalization, a ReLU activation function to stabilize the learning process, and a dropout layer with a 20% rate for regularization. The output layer utilized the softmax activation function to produce a probability distribution across the chord classes. Model training was performed using the Adam optimizer with a learning rate of 0.001 and a categorical cross-entropy loss function with a label smoothing factor of 0.1 to improve generalization. Top-K categorical accuracy (K=8) served as the primary evaluation metric. Although training was configured for 200 epochs, early stopping (patience=10) and learning rate reduction on plateau (factor=0.5, patience=5, minimum learning rate=1e-6) caused training to terminate after 25 epochs. With a training set of 4672 samples and a batch size of 8, the number of steps per epoch was 584.

Algorithm 1: Spectrogram Feature Extraction Pipeline

Input: Audio_file, target_size (1025 × 192)

Output: Preprocessed spectrogram Sf inal

1: y(t), fs ← Load audio signal from Audio_file

2: D ← STFT(y(t)) ▷ Apply Short-Time Fourier Transform

3: SdB ← 20 log10(|D|) ▷ Convert to decibel scale

4: Snorm ← MinMaxScale(SdB, range=[0, 255]) ▷ Normalize to [0, 255]

5: if width(Snorm) < target_width then ▷ Check spectrogram width

6: Sresized ← SymmetricPad(Snorm, target_size) ▷ Apply symmetric padding

7: else if width(Snorm) > target_width then

8: Sresized ← CenterCrop(Snorm, target_size) ▷ Apply center cropping

9: else

10: Sresized ← Snorm

11: end if

12: Sf inal ← Resize(Sresized, target_size) ▷ Final resize to target dimensions

13: return S final

Table 1

Model Architecture And Training Configuration

Layer Type / Parameter	Details / Value
Model	Multilayer Perceptron (MLP)
Input Layer	12-dimensional PCP features
Dense Layer 1	256 neurons, BatchNorm, ReLU, Dropout (0.2)
Dense Layer 2	256 neurons, BatchNorm, ReLU, Dropout (0.2)
Dense Layer 3	128 neurons, BatchNorm, ReLU, Dropout (0.2)
Output Layer	24 neurons, Softmax activation
Optimizer	Adam (lr = 0.001)
Loss Function	Categorical Cross-Entropy with label smoothing
Metrics	Top-K Categorical Accuracy (K = 8)

3) *Convolutional Neural Network*

As a complementary approach to the PCP-based methods, a Convolutional Neural Network (CNN) was implemented to learn hierarchical patterns directly from spectrogram images. This method provides a distinct perspective on the chord classification problem by analyzing the time-frequency representations of audio signals, rather than relying on preextracted harmonic features like PCP. The dataset consisted of grayscale spectrogram images representing eight chord classes: Am, B, Bm, C, Dm, Em, F, and G. All images were resized to a uniform shape of 1025 × 192 pixels. Preprocessing involved rescaling pixel values to the range [0, 1] by applying a factor of 1/255, followed by an 80-20 training-validation split. A batch size of 32 was used to enhance generalization and training speed. The CNN model processes input spectrograms of shape (1025, 192, 1). It consists of four convolutional blocks, each containing a Conv2D layer with ReLU activation and a 2×2 MaxPooling layer. The number of filters increases from 64 in the first block to 512 in the fourth. The resulting feature maps are flattened and passed through two fully connected layers, with the final output layer using a softmax activation for classification. The model was compiled with the Adam optimizer (learning rate=0.0001) and used categorical cross-entropy as its loss function. It was trained for 20 epochs, with 50 steps per epoch, based on 1600 training samples and a batch size of 32. A model checkpoint callback was used to save the version with the highest validation accuracy.

Table 2

CNN Model Architecture and Training Configuration

Layer Type / Parameter	Details / Value
Model	CNN (Convolutional Neural Network)
Input Layer	1025×192 spectrogram image (grayscale)
Conv2D (Layer 1)	64 filters, 3×3 kernel, ReLU activation
MaxPooling2D (Layer 1)	2×2 pool size, 2 strides
Conv2D (Layer 2)	128 filters, 3×3 kernel, ReLU activation
MaxPooling2D (Layer 2)	2×2 pool size, 2 strides
Conv2D (Layer 3)	256 filters, 3×3 kernel, ReLU activation
MaxPooling2D (Layer 3)	2×2 pool size, 2 strides
Conv2D (Layer 4)	512 filters, 3×3 kernel, ReLU activation
MaxPooling2D (Layer 4)	2×2 pool size, 2 strides
Flatten	Reshapes the output to a 1D vector
Dense (Layer 1)	1024 neurons, ReLU activation
Dense (Output Layer)	8 neurons, Softmax activation
Optimizer	Adam (lr=0.0001)
Loss Function	Categorical Crossentropy
Metrics	Accuracy

Results

This study evaluates the performance of three chord classification models—Support Vector Machine (SVM), Multilayer Perceptron (MLP), and Convolutional Neural Network (CNN)—using Pitch Class Profiles (PCP) as input features [4]. The models were assessed based on accuracy, loss curves, confusion matrices, and additional metrics, including precision, recall, and F1-score. The following sections detail the results for each model, followed by a comparative analysis.

A. *Convolutional Neural Network (CNN) Model*

The CNN model was designed to capture hierarchical patterns from spectrogram representations, making it particularly suitable for analyzing audio signals where local frequency structures are critical for chord recognition [5]. The model achieved a training accuracy of 97%, demonstrating its capacity to learn complex features from the input data. However, the model exhibited signs of overfitting when tested on an independent dataset. The epoch-versus-loss graph indicated a consistent decrease in training loss,

while the validation loss stagnated or increased in later epochs. Similarly, training accuracy increased rapidly, but validation accuracy showed limited improvement beyond a certain point. This performance gap suggests that the model memorized dataset-specific features rather than learning generalized representations of chord structures. A primary factor contributing to overfitting is the limited size and diversity of the training dataset [6]. In music information retrieval, models trained on small datasets often fail to generalize to real-world audio variations [7]. To address this, future work could incorporate data augmentation techniques such as pitch shifting, time stretching, and additive noise, which have been shown to improve robustness in audio classification tasks [8]. Additionally, regularization methods like dropout and weight decay could further mitigate overfitting [9].

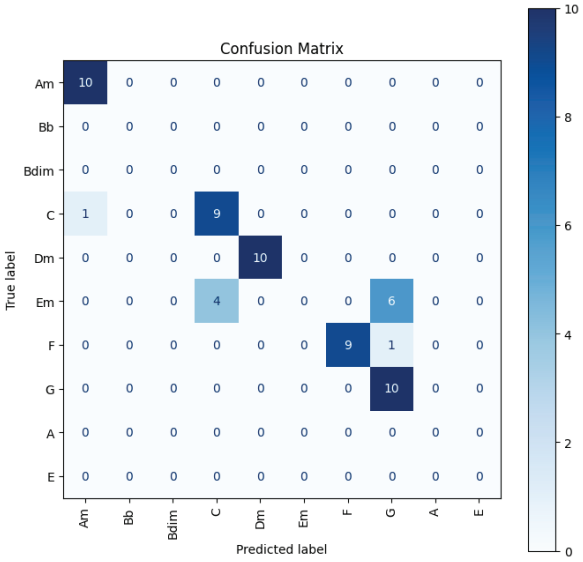

Fig. 4. Confusion Matrix of CNN Model

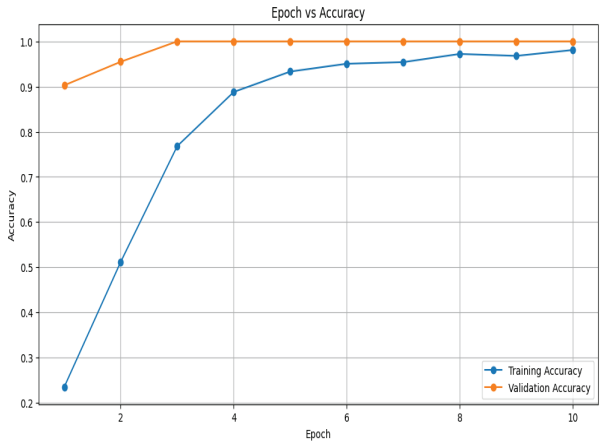

Fig. 5. Epoch vs Accuracy Curve of CNN Model

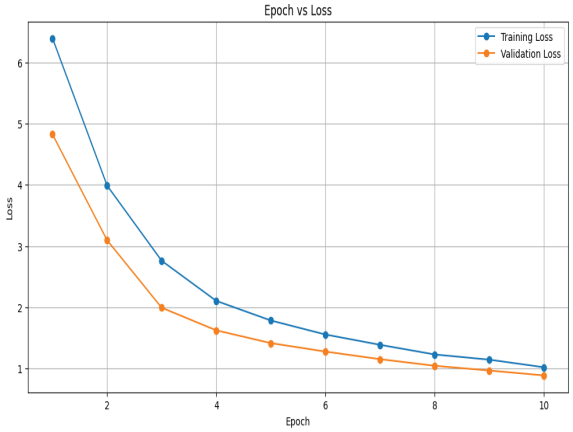

Fig. 6. Epoch vs Loss Curve of CNN Model

B. *Support Vector Machine (SVM) Model*

The SVM model, utilizing a suitable kernel function, demonstrated strong performance in separating chord classes by projecting PCP features into a higher-dimensional space [3]. The kernel trick enables efficient computation without explicitly transforming features, making SVMs computationally efficient for chord recognition [10]. Accuracy vs. Training Size: Accuracy showed a general upward trend with increasing training data, eventually plateauing. This aligns with typical SVM behavior, where performance gains diminish as data size grows [11]. Precision-Recall Curves: Most chord classes exhibited high precision and recall, indicating effective identification of positive instances with minimal false positives. Chords with fewer examples or similar harmonic structures showed slightly lower metrics, reflecting the challenges of class imbalance and feature similarity [12]. Multi-Class ROC Curves: ROC curves were positioned near the top-left corner, with high area under the curve (AUC) values for most chords. This confirms strong discriminative power across chord classes, though minor deviations occurred among harmonically similar chords [13]. Confusion Matrix: The matrix was dominated by diagonal entries, indicating correct classifications. Off-diagonal misclassifications occurred among chords with similar PCP profiles [14], underscoring the challenge of distinguishing harmonically related chords. The SVM model provides a computationally efficient alternative to deep learning approaches, achieving competitive accuracy with lower resource demands [10].

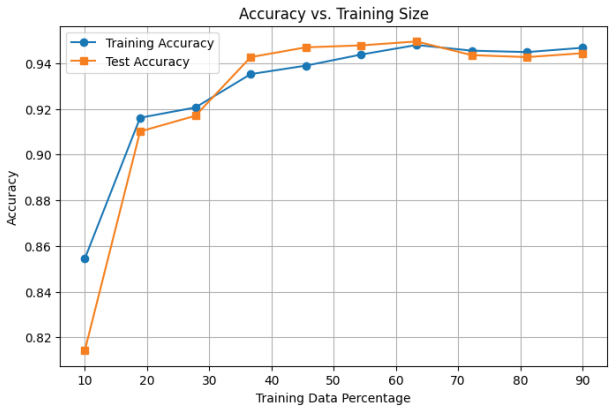

Fig. 7. Training Accuracy Curve of SVM Model

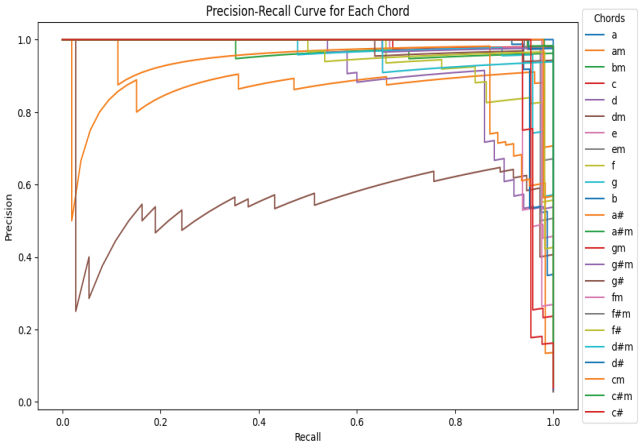

Fig. 8. Precision-Recall Curve of SVM Model

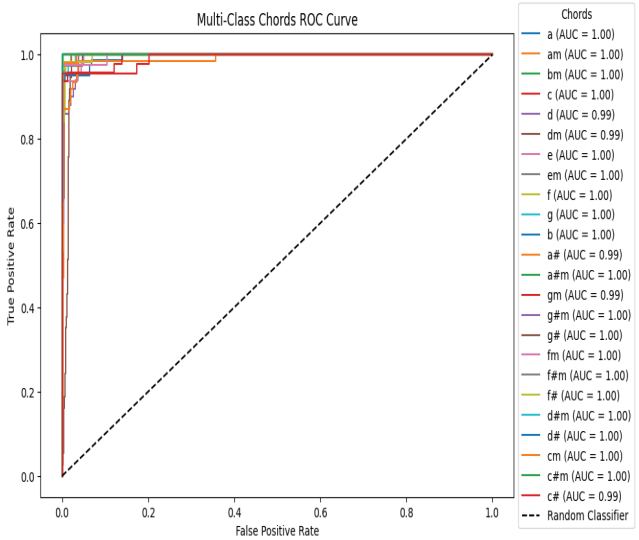

Fig. 9. ROC of SVM

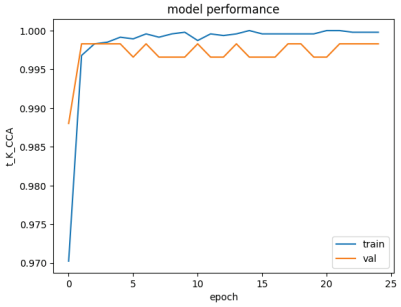

Fig. 10. Confusion Matrix of SVM Model

C. Multilayer Perceptron (MLP) Model

The MLP model, incorporating fully connected layers, batch normalization, and dropout, demonstrated robust learning and generalization. The use of dropout and batch normalization helped prevent overfitting, ensuring consistent performance across training and validation sets [9], [15]. Model Loss: Both training and validation loss decreased steadily over epochs, converging to low values. The minimal gap between these curves indicates effective generalization without significant overfitting. Model Accuracy: Training and validation accuracy increased rapidly to high values, with a small difference between them. This reflects the model's ability to learn generalizable features, supported by regularization techniques. Confusion Matrix: The matrix showed predominantly diagonal entries, with minimal off-diagonal misclassifications. Most errors occurred between chord pairs with similar pitch profiles [14], highlighting the model's overall effectiveness in chord differentiation. The MLP's performance underscores the value of feature engineering and architectural optimization in achieving high accuracy in chord classification tasks.

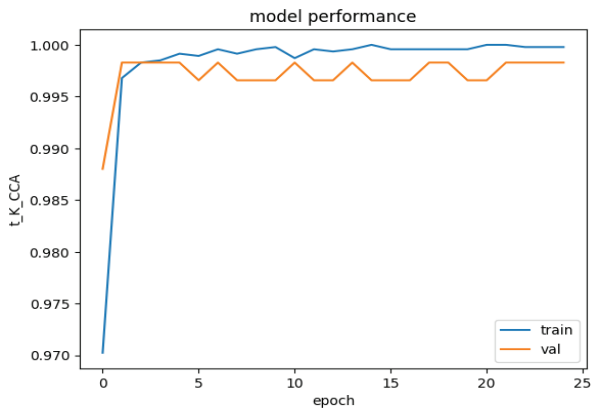

Fig. 11. Top-K Accuracy of MLP Model

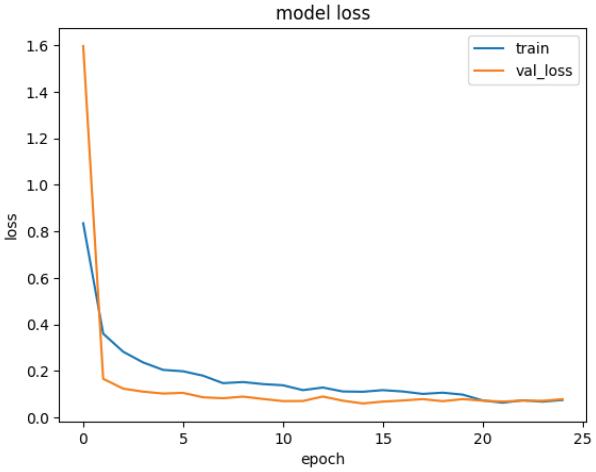

Fig. 12. Training Loss of MLP Model

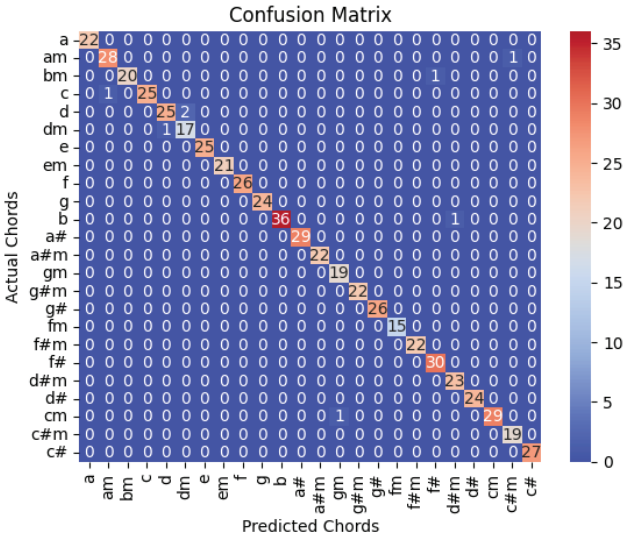

Fig. 13. Confusion Matrix of MLP Model

D. *Comparative Analysis and Performance Metrics*

A comparative analysis of the models reveals distinct trade-offs between accuracy, computational efficiency, and generalization:

Table 3.

Comparative Performance of Chord Classification Models

Model	Accuracy	Precision	Recall	F1-Score	Computational Efficiency
SVM	0.96	0.97	0.96	0.96	High
MLP	0.99	0.99	0.99	0.99	Medium
CNN	0.97 (train)	-	-	-	Low (due to complex structure)

The classification reports for SVM and MLP (Tables 3 and 4 in the original text) provide detailed precision, recall, and F1-scores for each chord class. The F1-score, as the harmonic mean of precision and recall, offers a balanced measure of model performance, especially valuable for evaluating classification tasks with potential class imbalance [16].

The confusion matrices for all models were used to identify specific misclassification patterns, particularly between harmonically similar chords [14]. This analysis provides insights into areas for improvement, such as feature enhancement and dataset expansion.

II. Discussion and Future Work

To summarize, our two-phase approach to chord classification integrates signal processing and machine learning to advance automated music analysis, with the objective of high accuracy standards through optimal feature extraction and strategic model selection. In Phase I, we utilized Pitch Class Profile extraction to evaluate harmonic content, measuring frequency components, pitch class energy distributions, octave-independent representations, and normalized energy proportions [17], [18]. Our model prioritizes computational efficiency for musicians and educators. The feature extraction pipeline we have proposed consumes minimal resources (in terms of storage, CPU, RAM) as only 12-dimensional PCP vectors with normalized values are required for training the classifiers, and if required, can be comfortably executed using widely available mobile devices or cloud platforms [19]. Musicians, even with limited technical expertise, regularly record audio samples to learn new songs. This collected audio data is analyzed using various tools to transcribe chord progressions. In the absence of a reliable tool to identify chords automatically, a musician may resort to manually transcribing entire songs or, in the worst case, spending hours identifying each chord by ear in complex musical pieces. This scenario can be restated as 100% manual effort for chord identification. On the other hand, if a musician makes use of our proposed ML approach, this will result in a significant reduction in transcription time and effort, saving precious resources in terms of time and learning efficiency [20]. During our evaluation, we realized that out of diverse audio samples across multiple instruments, our models achieved 96–99% accuracy, i.e., near perfect chord identification. Further, this model can be easily extended to musicians of any skill level, from beginners to advanced players, without any proportional increase in computational demand. In the future, with increasing needs, the existing

model can further be scaled to recognize more complex chord types beyond major and minor triads [2]. Phase II introduced a comparative ML framework to evaluate the performance of multiple classification architectures while managing chord recognition across diverse conditions. The comparative ML framework allows us to examine the impact of variables like instrument timbre, recording quality, acoustic environment, feature representation, and model architecture on classification accuracy, generalization capability, and computational efficiency [21]. To implement this framework, a major effort is required to develop relevant and functional preprocessing pipelines for audio feature extraction, which is mostly done through automated signal processing algorithms, following which, the classification analysis is easily reproduced using standard machine learning libraries [11]. With the global scale of music diversity, scalability plays a crucial role in evaluating our model's relevance. Simple chord progressions are mostly subsumed into larger musical compositions. Considering this, a modular approach technique employing hierarchical classification or multi-stage recognition can be used to partition complex songs into smaller chord segments or hierarchical musical structures [5]. Analysis can be done individually on the obtained smaller segments based on the complexity and duration of the recordings. This modular approach also helps in efficiency by focusing only on target sections of interest. In our evaluation, we validated the model's performance across guitar, piano, violin, and accordion recordings. The findings indicate that the model maintains consistent behavior across varying instruments and recording conditions, exhibiting similar classification patterns, validating the model's generalizability [6]. Moreover, the proposed PCP-based feature extraction further enhances cost-effectiveness by reducing computational overhead, memory requirements, and processing time, which minimizes resource strain on mobile devices. This efficient strategy provides musicians with time to gradually learn and understand chord progressions, reducing the cognitive load associated with manual transcription. Though automated music analysis technology is advancing rapidly in the global scenario, real-world deployment of comprehensive chord recognition systems is still in an early stage [1]. For music education in developing regions, the actual implementation of AI-powered learning tools is still not widespread. However, we have utilized advanced feature extraction techniques (PCP and spectrograms) to represent musical content realistically, considering real-world audio recordings from diverse instruments, varied acoustic environments, end-to-end audio processing, diverse playing styles, and multi-instrument compatibility [14]. This feature-based framework serves as a foundation for our future research focusing on the recognition of extended chord types (seventh chords, suspended chords, augmented/diminished chords), considering different musical genres, and incorporating granular musical aspects, for example, chord inversions and voicing variations. Additionally, we have planned to include real-time chord detection and integration with mobile music learning applications in future versions of our work to further improve music education and transcription accessibility [4].

III. Conclusions

Adoption of automated music analysis technology, despite introducing powerful transcription capabilities to musicians and offering significant educational value to learners, the implementation is always associated with several challenges. These challenges include variability in audio quality, diverse instrument timbres, recording conditions, computational complexity, limited training datasets, overfitting risks, lack of proper feature representation, and non-optimal model selection, which leads to reduced accuracy and generalization [9]. This study proposes a novel approach for chord classification, using advanced signal processing and machine learning. Our work provides a novel framework combining Pitch Class Profile extraction, multiple ML architectures, and comparative analysis to devise a competent system capable of capturing the musical chords. Using normalized PCP features, our models successfully classified all 24 major and minor chords across diverse instruments and acoustic conditions. The classification system considers variables like pitch class energy distribution, harmonic content, frequency components, instrument timbre variations, recording quality, and octave-independent representations, which influence the chords [18]. Furthermore, our research demonstrated the effectiveness of PCP-based features across various instruments, including guitar, piano, violin, and accordion. The results indicated that machine learning models significantly outperformed naive pattern-matching approaches, with error rates reduced from 32% to 4% for certain instruments. Our work uniquely combines signal processing and comparative machine learning analysis to develop a framework that captures the complex dynamics of musical harmony [20]. Earlier researchers' models focused only on a single classification approach or limited chord sets. Our framework incorporates a PCP-centric feature extraction pipeline trained on real-world audio recordings

from multiple instruments, diverse acoustic environments, and varied playing styles, enabling our models to evaluate critical harmonic characteristics for chord classification [17]. This allows for accurate, data-driven identification of all 24 chord types, optimizing the feature representation phase of the recognition process. Further, our comparative ML analysis evaluates SVM, MLP, and CNN architectures, incorporating performance metrics for accuracy, precision, recall, and generalization capability [16]. This comprehensive and rigorous evaluation improves on earlier researchers' work by addressing multi-instrument compatibility, model robustness, and real-world applicability, thus bridging gaps that overlooked practical deployment conditions and cross-instrument generalization [10].

Acknowledgement

The authors would like to thank the Department of Electronics and Computer Engineering at Pulchowk Campus for their support and resources. We also acknowledge Jos Malsy for making a subset of chord samples publicly available.

References

[1] T. Fujishima, "Realtime chord recognition of musical sound: A system using common lisp music," in Proceedings of the International Computer Music Conference. International Computer Music Association, 1999, pp. 464–467.

[2] J. Osmalskyj, J.-J. Embrechts, S. Pierard, and M. Van Droogenbroeck, "Neural networks for musical chords recognition," in Proceedings of the 11th International Conference on Signal Processing and Multimedia Applications, M. S. Obaidat and J. Filipe, Eds. SciTePress, 2012, pp. 75–80.

[3] C.-W. Hsu, C.-C. Chang, and C.-J. Lin, "A practical guide to support vector classification," Journal of Machine Learning Research, vol. 1, pp. 1–16, 2010.

[4] I. Fujinaga and K. McMillan, "Realtime recognition of orchestral instruments," in Proceedings of the International Computer Music Conference. International Computer Music Association, 2000, pp. 252–255.

[5] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent, "Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription," in Proceedings of the 29th International Conference on Machine Learning, J. Langford and J. Pineau, Eds. Omnipress, 2012, pp. 1159–1166.

[6] E. J. Humphrey, J. P. Bello, and Y. LeCun, "Feature learning and deep architectures for music audio," in Proceedings of the 14th International Society for Music Information Retrieval Conference, 2013, pp. 447–452.

[7] L. Deng and D. Yu, "Deep learning: Methods and applications," Foundations and Trends in Signal Processing, vol. 7, no. 3–4, pp. 197–387, 2014.

[8] N. Jaitly and G. E. Hinton, "Vocal tract length perturbation (vtlp) improves speech recognition," in Proceedings of the 30th International Conference on Machine Learning, S. Dasgupta and D. McAllester, Eds., vol. 28. PMLR, 2013, pp. 1–5.

[9] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," Journal of Machine Learning Research, vol. 15, no. 1, pp. 1929–1958, 2014.

[10] I. Sheikh, K. Lee, and X. Li, "Efficient chord recognition using support vector machines," in Proceedings of the 17th International Society for Music Information Retrieval Conference, 2016, pp. 661–667.

[11] C. M. Bishop, Pattern Recognition and Machine Learning. Springer, 2006.

[12] C. N. Silla, A. L. Koerich, and C. A. A. Kaestner, "The problem of class imbalance in music genre classification," in 2009 IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, 2009, pp. 169–172.

[13] T. Fawcett, "An introduction to roc analysis," Pattern Recognition Letters, vol. 27, no. 8, pp. 861–874, 2006.

[14] D. P. W. Ellis and G. E. Poliner, "Identifying cover songs with chroma features and dynamic programming beat tracking," in 2007 IEEE International Conference on Acoustics, Speech and Signal Processing, vol. 4. IEEE, 2007, pp. IV–1429–IV–1432.

[15] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in Proceedings of the 32nd International Conference on Machine Learning, F. Bach and D. Blei, Eds., vol. 37. PMLR, 2015, pp. 448–456.

[16] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," Information Processing and Management, vol. 45, no. 4, pp. 427–437, 2009.

[17] J. C. Brown, "Calculation of a constant q spectral transform," The Journal of the Acoustical Society of America, vol. 89, no. 1, pp. 425–434, 1991.

[18] M. Müller, Fundamentals of Music Processing: Audio, Analysis, Algorithms, Applications. Springer, 2015.

[19] B. McFee, C. Raffel, D. Liang, D. P. W. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in Proceedings of the 14th Python in Science Conference, S. van der Walt, K. Huff, and J. Bergstra, Eds., 2015, pp. 18–25.

[20] E. Gómez, "Tonal description of music audio signals," Ph.D. dissertation, Universitat Pompeu Fabra, 2006.

[21] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyperparameter optimization," in Advances in Neural Information Processing Systems 24, J. Shawe-Taylor, R. S. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2011, pp. 2546–2554.