Rethinking Music Recommendations: An AI-Driven Approach

Facundo Herrera, Ehiane Oigiagbe, Justin Simm, Adrian Casabal, Ethan Villalovoz, Chibuike Ugwu
CPTS 440: Artificial Intelligence
School of Electrical Engineering & Computer Science, Washington State University

Project Proposal

***Abstract*—The rise of music streaming services has made personalized recommendations essential, yet existing platforms often fail to meet user expectations. We propose an AI-driven song recommender utilizing collaborative filtering, NLP, deep learning, reinforcement learning, and graph-based models to improve accuracy and adaptability. By analyzing user preferences, listening history, and song attributes, our system aims to outperform traditional models, enhancing user satisfaction through data-driven personalization and real-time adaptation.**

I. Introduction

Music streaming services have revolutionized how people discover and enjoy music. Platforms like Spotify, Apple Music, and YouTube Music use recommendation systems to enhance user experience by suggesting personalized playlists and songs. However, these systems often produce inconsistent results, with some users praising their accuracy while others express frustration over irrelevant recommendations.

To address these limitations, we propose a third-party AI-driven music recommender system that leverages state-of-the-art machine learning and deep learning techniques. Our system takes a user's input—such as a song, playlist, or profile preferences—and generates personalized recommendations based on an ensemble of advanced techniques, including collaborative filtering, natural language processing (NLP), audio analysis, reinforcement learning (RL), and graph-based models. Using these approaches, our system aims to enhance recommendation accuracy, adapt to user preferences in real-time, and provide a superior music discovery experience compared to traditional methods.

II. Algorithms

A. Collaborative Filtering

Collaborative filtering is a fundamental technique for personalized music recommendations. It identifies user-song interaction patterns and makes recommendations based on similar listening behaviors. Our approach includes using matrix factorization methods and deep learning models.

B. Natural Language Processing (NLP) for Lyrics & Metadata

We incorporate NLP techniques to analyze song lyrics, user reviews, and metadata to enhance personalization. Our approach includes using word embeddings such as Word2Vec to capture semantic relationships between lyrics and user preferences as well as transformers and recurrent neural networks(RNNs) for extracting context from text.

C. Audio Analysis with Deep Learning

Traditional recommendation systems often overlook the audio characteristics of songs. We utilize Convolutional Neural Networks (CNNs) to extract high-level musical features such as rhythm, tempo, melody, and harmony. We can also use spectrogram analysis to learn representations from raw audio.

D. Reinforcement Learning (RL) for Adaptive Recommendations

Static recommendation models fail to adapt to shifting user preferences over time. We incorporate Reinforcement Learning (RL) to adjust recommendations based on real-time user feedback dynamically. We will use reward functions to distinguish between positive and negative feedback and policy optimization techniques to continuously refine the recommendation strategy.

III. Data Collection & Preparation

A. Music Data Collection

We will source music-related data from publicly available datasets and online music databases, ensuring that the dataset includes a variety of genres, artists, and song features. The primary datasets we will use include:

- FMA (Free Music Archive) Dataset (UCI Repository):
  - Contains metadata for thousands of songs, including genre, tempo, artist, language, and lyrics.
  - Provides audio analysis features such as spectrograms and frequency distributions, useful for deep learning models.
- Supplementary Data from APIs (if needed):
  - Spotify API: Can be used to retrieve popularity scores, playlists, and user interactions.
  - Lyrics APIs: Provides lyrical content for NLP-based analysis.

B. User Profile Data

For personalized recommendations, we need to model user preferences based on historical interactions. This includes:

- Liked Songs: Tracks explicitly favorited by the user.
- Followed Artists: Helps infer genre and stylistic preferences.
- User Playlists: Captures user-curated collections and music patterns.
- Listening Behavior: Implicit feedback, such as skipped songs, replay frequency, and listening duration.
- Language Preferences: Identifies linguistic preferences for song recommendations.

If real user data is unavailable, we will generate simulated user profiles based on aggregated preference patterns from public datasets or pre-existing user interaction datasets.

IV. Evaluation Metrics and Experiments

To assess our intelligent song recommender, we will evaluate speed/response time, accuracy, scalability, comparative analysis, and visualization/reporting.

A. Speed/Response Time

Speed/response time measures how quickly recommendations are generated. We will compare response times against industry standards and optimize performance using caching and efficient data structures.

B. Accuracy

Accuracy will be assessed using implicit feedback (plays, skips, replays) and explicit feedback (ratings). We will compare our recommendations against traditional collaborative filtering and content-based filtering models.

C. Scalability

Scalability testing includes varying dataset sizes and assessing performance with extensive listening histories (500+ liked songs). Stress testing will evaluate concurrent user loads.

D. Comparative Analysis

Comparative analysis involves benchmarking our recommendations against platforms like Spotify and YouTube Music. User surveys will provide insights into satisfaction and preference trends.

E. Visualization and Reporting

Visualization and reporting will analyze user interactions using charts, graphs, confusion matrices, and precision-recall curves to improve engagement and model effectiveness.

V. Backup Plans

If needed, we will implement:

- **Artist Recommender**: Recommends artists instead of individual songs, reducing computational overhead.
- **Playlist Recommender**: Matches user playlists with publicly available ones based on metadata and NLP-based analysis.

VI. Team Roles and Responsibilities

Each team member is responsible for a specific component of the project, ensuring efficient development and implementation.

- **Algorithms**:
  - Implement and optimize collaborative filtering, NLP, CNNs, reinforcement learning, and graph-based recommendation methods.
  - Research and fine-tune deep learning models for improved recommendation accuracy.
- **UI/UX**:
  - Design a user-friendly interface to input song preferences and receive recommendations.
  - Develop intuitive interactions for exploring recommendations and providing feedback.
- **Data Gathering and Management**:
  - Collect and preprocess data from FMA, online music databases, and APIs.
  - Manage data pipelines for user preferences, song features, and metadata.

- **Backend Development**:
  - Implement backend infrastructure to handle recommendation logic and user queries.
  - Integrate machine learning models with a scalable server environment.
- **Metrics and Analysis**:
  - Define evaluation metrics such as precision, recall, F1-score, and engagement metrics.
  - Compare model performance with baseline methods, such as collaborative filtering and Spotify's approach.
- **Testing and Debugging**:
  - Ensure model stability through unit testing, stress testing, and debugging.
  - Validate recommendation accuracy through real-world testing scenarios.

VII. Project Timeline and Milestones

The project is divided into key phases, each with defined checkpoints to ensure steady progress and quality control.

A. Phase 1: Data Gathering and Processing (Jan - Feb 24)

- Collect and preprocess song metadata, audio features, and user interaction data.
- Implement feature extraction methods for lyrics, audio, and user preferences.
- **Checkpoint 1 (Feb 24)**: Ensure datasets are cleaned, structured, and ready for model training.

B. Phase 2: Model Building and Tuning (Feb 24 - March 17)

- Implement collaborative filtering, NLP-based models, CNNs, and reinforcement learning approaches.
- Train and fine-tune models for performance and scalability.
- **Checkpoint 2 (March 17)**: Initial model results and baseline performance evaluation.

C. Phase 3: Testing and Debugging (March 17 - April 7)

- Evaluate models using real user profiles, accuracy metrics, and feedback loops.
- Debug and optimize computational efficiency (e.g., response time, scalability).
- **Checkpoint 3 (April 7)**: Fully tested and optimized recommendation system.

D. Phase 4: Application Development and Integration (April 7 - April 21)

- Build UI/UX components and integrate them with the backend system.
- Implement real-time recommendation processing and feedback collection.
- General testing and debugging of software to ensure a seamless user experience.
- **Presentation Start (April 21)**: Final project demonstration and report submission.

This structured timeline ensures continuous progress and allows for iterative improvements based on test results and user feedback.