Cover Song Similarity: Detailed Description

1. Introduction

We want to design a similarity metric that quantifies the relationship between an original song and its cover versions. Intuitively, songs that are covers of each other share certain musical characteristics, such as harmonic progressions, melodic contours, and, to some extent, rhythmic patterns, despite potential differences in instrumentation, tempo, recording quality etc. On the contrary, pairs of unrelated songs typically differ more significantly in these musical aspects.

As such, this task poses technical and conceptual challenges. On the technical side, we must identify suitable features and models that can capture the intrinsic “identity" of a piece. On the conceptual side, we must define what musical similarity means in a context where songs can differ along multiple dimensions (for instance, timbre, key, style, etc) yet still be recognized as similar (e.g., in the case of covers).

Based on the above, we propose a metric learning approach which utilizes the features provided by the Da-TACOS dataset and refines these with a model that learns to output a distance that is small for cover song pairs and large for non-cover pairs.

2. Understanding the Dataset

The Da-TACOS dataset provides detailed metadata and pre-extracted features for a large number of songs. It contains two subsets, namely the benchmark subset for benchmarking cover song identification systems and the cover analysis subset for analyzing the links among cover songs, with pre-extracted features and metadata for 15,000 and 10,000 songs, respectively; see: GitHub - MTG/da-tacos: A Dataset for Cover Song Identification and Understanding. Songs are grouped into "cliques" of covers. Each clique represents a set of songs known to be covers of one another. Key aspects of the dataset:

- **Metadata:** Titles, artist names, release years, and other relevant to the track information. While metadata can provide hints (e.g., identical titles and artist overlaps), it is not always reliable because covers can have slight title variations, and artist metadata alone is insufficient for robust similarity estimation.
- **Musical Features:** The dataset includes pre-computed harmonic and pitch-based features, such as chroma and Harmonic Pitch Class Profile (HPCP). These features are well-suited for measuring harmonic similarity as they capture pitch content in a way that

is somewhat invariant to timbre and instrumentation variations. Concretely, features in the Da-Tacos dataset include:

- **chroma_cens features:** Represent the energy distribution across the 12 semitones of the Western scale. Chroma can help identify songs that share similar chord progressions even if they differ in arrangement. For more details: Feature Design (Chroma, CENS)
- **hpcp features:** More advanced pitch-class representations that help recognize harmonic similarity under different tuning conditions. HPCP (Harmonic Pitch Class Profile) can be more robust to key changes and detuning than standard chroma vectors.
- **crema features**: Features extracted using the CREMA (Convolutional and Recurrent Estimators for Music Analysis) toolkit, which is designed for chord recognition. The CREMA toolkit provides models that analyze audio files to predict chord structures, offering outputs such as class likelihoods and chord predictions. For more details: https://crema.readthedocs.io/en/latest/index.html
- **key_extractor features,** which likely contain information about the track's key :
  - **key**: The detected musical key (e.g., C major, G minor).
  - **scale**: The scale type, such as major or minor.
  - **strength**: The confidence or strength of the key detection.
- **madmom_features (dictionary)**: Madmom is an audio signal processing library written in Python with a strong focus on music information retrieval (MIR) tasks (see: GitHub - CPJKU/madmom: Python audio and music signal processing library) Features extracted using the Madmom library, likely targeting rhythm and onset detection:
  - **novfn**: Novelty function, which highlights changes or "novel" events in the audio.
  - **onsets**: Onset times indicating when notes or percussive events begin.
  - **snovfn**: Smoothed novelty function for a less noisy version of novfn (Superflux audio novelty function at each audio frame).
  - **tempos**: Estimated tempos (beats per minute).
- **mfcc_htk**: Represents Mel-Frequency Cepstral Coefficients (MFCCs), calculated using the HTK (Hidden Markov Toolkit) method. MFCCs are widely used for speech and audio analysis to capture the spectral envelope.
- **tags:** Tags represent metadata and descriptors associated with the track, such as genre, style, or other relevant annotations.
- **label**: Denotes the work ID (WID) of the track. The WID serves as a label identifying the specific group to which the track belongs.
- **track_id**: The unique performance ID (PID) of the track. This identifier is used to uniquely distinguish each performance within the dataset.

Potential Data Redundancies and Issues:

- Some cliques may contain near-identical recordings (e.g., same studio take or remaster), which might artificially make the task trivial. It's important that our solution generalizes beyond these trivial cases.
- The dataset may have varying lengths of audio or differing sampling rates. Some feature normalization and standardization will be needed in such a case.
- Differences in audio quality and background noise might also impact the reliability of extracted features.

In our approach we would use the chroma_cens and hpcp features in our initial implementation, as these features can capture the harmonic and tonal characteristics of music, which are often preserved across versions of a song (We discuss more extensively pros and cons in the next section). As the system evolves, we can incorporate additional features and attempt to enhance its performance, like the MFCC features, to potentially distinguish between different instruments or rhythmic features (e.g.,madmom tempos), which can aid in identifying rhythmic structures.

Additionally, we would likely need to normalize and align these chosen features and potentially use temporal aggregation strategies (e.g., averaging feature frames or learning temporal alignment strategies) to ensure robustness.

3. Designing the Similarity Metric

Conceptual Approach

We will construct a metric d(x1,x2) that measures how similar two songs are. We want d(x1,x2) to be small for covers while it should be large for non-covers. The chosen features should capture invariant aspects of the music, primarily harmonic structures and pitch patterns.

Feature Selection Rationale

As mentioned, for our initial implementation we would use the chroma_cens and HPCP features. We discuss pros and cons.

Why Chroma and HPCP?

- **Pros:**
    - Harmonic invariance: Chroma and HPCP features likely capture harmonic content in a way that is more stable against changes in instrumentation and style.

    - Key invariance: HPCP and chroma features can often detect transposed versions (in terms of key) of the same song. Covers often shift keys to suit a vocalist's range, so key invariance is important.
- **Cons:**
    - Loss of timbral and rhythmic information: Purely pitch-based features can fail to detect covers which differ drastically in style or to catch similarities that rely on rhythmic or timbral patterns.
    - Potential false positives: Songs with similar chords but no cover relation might be deemed similar based on harmonic content alone.

As discussed, to mitigate these issues, we could incorporate other feature types such as tempo or timbral descriptors (e.g., MFCCs), to refine our metric.

4. Methodology

4.1 Baseline Approaches

Cosine Distance on Aggregated Chroma/HPCP Vectors: A simple starting point might be to aggregate chroma or HPCP features over a song (e.g., by taking the mean or a median vector over time) and then compute a straightforward similarity score, such as the cosine similarity, between pairs of aggregated features. We could also combine the two metrics into a single vector and perform a similar approach (aggregation, such as concatenation after normalization, etc.).

- **Pros:**
    - Easy to implement, interpretable.
    - Computationally efficient for large datasets.
- **Cons:**
    - Lacks temporal structure, which might not capture variations across sections of the song.
    - Oversimplifies the temporal evolution of tonal patterns, making it prone to errors in identifying structurally rearranged covers.

Dynamic Time Warping (DTW) on Feature Sequences:
Instead of collapsing all features into a single vector, we can treat chroma/HPCP vectors as sequences and use DTW to align sequences of harmonic features. This helps account for temporal differences.

- **Pros:**
    - Better temporal alignment, handles differences in song length and performance.

    - Captures structural variations, such as timing changes, between covers and originals.
- **Cons:**
    - Computationally more expensive, especially for large datasets.
    - Requires a robust similarity measure on a frame level basis.

Incorporating Cross Recurrence Quantification (CRQ):

Building on DTW, Cross Recurrence Plots (CRPs) can be used to analyze how similar state-space representations of two songs are across time. CRPs enable tracking of potentially curved and disrupted patterns in tonal sequences. As such, they are particularly robust for identifying covers that differ significantly in arrangement or tempo. Recurrence Quantification Analysis (RQA) measures, such as Lmax, Smax, and Qmax, quantify the similarity by identifying the longest matching tonal sequences, curved patterns (to account for tempo variations), and disruptions. See Cross recurrence quantification for cover song identification - IOPscience for more details.

- **Pros:**
    - Provides temporal flexibility, capturing nuanced similarities between songs, even with significant tempo deviations or interruptions.
    - Robust to rearrangements and small disruptions in tonal structure, common in covers.
    - Offers interpretable metrics (e.g., length of matching tonal sequences).
- **Cons:**
    - Requires additional preprocessing steps like constructing delay-coordinate state space embeddings.
    - Computationally more intensive than DTW for larger datasets.

4.2 Advanced Approaches

Siamese Neural Network for Metric Learning:
We can train a neural network model (e.g., a Siamese network, see Siamese Neural Networks for One-shot Image Recognition) which takes two sequences of features (chroma/HPCP) as input and outputs a similarity score. The model would learn an embedding space where cover versions are close together and unrelated songs are far apart.

- **Method:**
    - **Input:** Two sets of feature sequences: (X1,X2).
    - **Output**: A similarity score of the network's output embeddings (e.g., cosine similarity)
    - **Network:** BiLSTM or Transformer-based layers to process temporal sequences, followed by a shared embedding layer.

    - **Loss:** Contrastive or triplet loss. For each known cover pair, we minimize the distance between embeddings; for known non-cover pairs, we maximize the distance.
- **Pros:**
    - Learns a domain-specific metric, can capture complex patterns, handle temporal information, and become robust to small variations.
    - Can incorporate CRQ-inspired features like Qmax etc, as auxiliary inputs to guide learning.
- **Cons:**
    - Requires careful training, hyperparameter tuning, and more computational and data resources.
    - Can be sensitive to overfitting without careful data augmentation or regularization.
    - Suffers from low interpretability.

Clustering-based Approaches:
We could also use unsupervised methods to learn a representation of the songs and then measure how well cover cliques form tight clusters. However, since we have explicit cover pair labels, supervised metric learning is likely more effective.

Early Fusion of HPCP and MFCC Features (If we use MFCC Features):
Given that we incorporate additional features, we can build on the complementarity of HPCP (harmonic features) and MFCC (timbral features) based on https://arxiv.org/pdf/1707.04680. This method fuses cross-similarity matrices (CSMs) derived from the two feature sets. The fusion is performed before alignment, to leverage both harmonic and timbral information during similarity computation.

- **Pros:**
    - Combines harmonic and timbral information, capturing complementary aspects of the music.
    - Robust against key transpositions, tempo changes, and timbral variations.
    - Incorporates structural information to fill gaps where individual features fail.
- **Cons:**
    - Requires preprocessing steps like beat tracking, which can introduce errors.
    - Could be computationally intensive due to alignment and fusion operations.

5. Evaluation and Discussion

To measure the performance of our approach, we can evaluate with metrics on ranking and retrieval tasks:

- **Precision at K / Recall at K:** For each query song, rank all other songs by similarity and check if known covers appear among the top results.

- **Mean Average Precision (MAP) or Mean Reciprocal Rank (MRR):** Comprehensive retrieval measures that summarize how well the model ranks true covers.

Considerations:

- **Overfitting:** Especially in the case where DL is used (e.g., Siamese nets), if the model is too finely tuned to the dataset's known cliques, it may not generalize. Using regularization, data augmentation (e.g., adding small pitch shifts or noise), and cross-validation can mitigate this.
- **Assumptions and Limitations:**
    - The assumption that harmonic features alone can sufficiently distinguish covers might be limiting, especially in genres where different songs share similar chord progressions.

Interpreting Results:

- If the model can consistently identify known covers as top matches and correctly differentiate them from unrelated songs, it suggests the metric is capturing core musical invariants.
- If it confuses unrelated non-covers as covers, this indicates a need to better tune the model, incorporate more diverse features or a more sophisticated model architecture\solution.

6. Implementation Details

Data Preprocessing:

- Normalize HPCP/chroma features.
- Segment the songs into fixed-length frames and optionally use DTW or learnable alignment.
- Split data into training (cover pairs known), validation, and test sets. Ensure that entire cliques are separated between sets to avoid trivial memorization. (The split could be balanced between the same clique and unrelated pairs).

Model Training (Siamese Network Example):

- Architecture:
    - Input: Two sequences of feature vectors.
    - Layers:
        - BiLSTM/Transformer encoder to process sequences,
        - Dense embedding layer to map sequences into a common vector space,
        - Distance computation layer (e.g., L2 or cosine) to produce the similarity score.
- Loss: Contrastive loss:

$$L = y \cdot \frac{1}{2}(d^2) + (1 - y) \cdot \frac{1}{2}\{\max(0, m - d)^2\}$$

where d is the distance between embeddings, y indicates whether the pair is a cover (y=1) or not (y=0), and m is a margin.

Post-Training Evaluation:

- Compute similarity scores for pairs in the test set.
- Rank candidates for each original song and measure how well covers are retrieved.

7. Example Outputs

- **Distance Matrices:**
  The system can display a matrix of distance scores between a set of songs in a clique and a random sample of non-covers. Ideally, cover pairs form a block of low distances.
- **Ranking Examples:**
  For a given original track, list the top-5 closest songs according to the metric. Show that the cover versions rank highly, while unrelated songs rank lower.

8. Scalability and Improvement Suggestions

Improvements:

- **Feature Augmentation:** Incorporate timbral features (e.g., MFCCs) and rhythm features (e.g., onset patterns) to handle tricky cases.
- **Model Complexity:** Experiment with Transformer-based models that might better handle varying lengths and complex temporal dependencies.
- **Cross-Dataset Validation:** Test on another cover song dataset (if available) to ensure generalization.

Scaling Up:

- Deploy the trained model in a system that indexes large music catalogs. The learned embedding space can be leveraged to quickly retrieve covers by using approximate nearest neighbor search techniques.
- Enhance robustness by incorporating domain adaptation methods if the model is applied to different genres or distributions than the training set.
- Incorporate more data to train on. Use additional songs and cliques from external resources.

9. Conclusion

The proposed approach using pitch-based harmonic features and a metric learning model at the initial phases of the system design aims to capture the essence of what makes a cover a cover.. While simple approaches (like cosine distances on aggregated chroma vectors) can provide a baseline, a deep learning metric learning model (such as a Siamese network) holds the promise of a more nuanced and reliable similarity measure especially as data scale up.

With careful handling of the dataset's redundancies, appropriate evaluation strategies, and openness to feature augmentation, this metric can be tuned to robustly identify cover songs and provide insights into the subtle dimensions of musical similarity.