Overview

Playlist generation

A playlist is a sequence of songs.

‣ How should we evaluate playlist algorithms?

We propose an evaluation scheme which is:

‣ Simple, automatic, scalable, objective, and user-centric

Key observation:

‣ Playlist algorithms are generative models of a language.

Learning algorithm:

‣ Optimally integrates multiple simple playlist algorithms

Language modeling

‣ Playlist algorithm \( A \) → distribution \( P_A \) on song sequences.

‣ Can be thought of as a natural language model:

‣ Words → Songs
‣ Sentences → Playlists

‣ Evaluate algorithms by likelihood of real playlists

Algorithm evaluation

‣ Given a library of songs \( \mathcal{X} \)
‣ Collect a sample of playlists \( \mathcal{S} \subset \mathcal{X}^* \)
‣ Score algorithm by average log-likelihood:

\[
\mathcal{L}(\mathcal{S} \mid A) = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \log P_A [s]
\]

Previous methods

Human evaluation

"How good is this playlist?"

PRO

‣ Directly involves users
‣ Measures what we want

CON

‣ Expensive
‣ Does not scale
‣ Noisy/subjective

Semantic cohesion

"This playlist is 80% Blues"

PRO

‣ Simple, automatic

CON

‣ Semantics are ambiguous
‣ Cohesion ≠ quality

Sequence prediction

"Which song comes next?"

PRO

‣ Automatic
‣ Uses standard IR techniques

CON

‣ Needs negative examples
‣ Observation sparsity

Markov models

Uniform shuffle

‣ Pick each song independently, uniformly at random

‣ Obvious baseline

\[
P[X_{t+1} \mid X_t]
\]

Weighted shuffle

‣ Pick each song independently from a weighted distribution

‣ Can encode user preference or popularity

\[
P[X_{t+1} \mid X_t]
\]

Random walks

‣ Construct a neighborhood graph over songs

‣ Next song selected from neighbors of the current song

Markov mixtures

Mixture model

‣ Given a set of Markov chains \( P_1, P_2, \ldots, P_m \)

‣ Form the mixture distribution:

\[
P[X_{t+1} \mid X_t] = \sum_{i=1}^{m} \mu_i P_i [X_{t+1} \mid X_t]
\]

‣ Learn the weights \( \mu_i \) to maximize likelihood of training sample

Experiment algorithm

‣ At time \( t \), select a Markov chain \( A_{t} \sim \mu \)

‣ Pick \( X_{t+1} \) according to \( A_{t}, X_t \)

‣ Integrates heterogeneous data

‣ Optimizes neighborhood graph connectivity

Results

Data

Million Song Dataset\(^1\), Art of the Mix playlists\(^2\)

‣ 26752 songs by 5629 artists

‣ 66250 bigrams

Audio kNN

‣ Optimized VO histograms of ENTs \( \in \mathbb{R}^{222} \)

‣ Random walk on \( k \)-nearest neighbor graph

Tag kNN

‣ Echo Nest artist terms \( \in \{0,1\}^{7643} \)

‣ Cosine-similarity, \( k \)-nearest neighbor graph

‣ Implicitly maximizes semantic cohesion

Familiarity

‣ Shuffle weighted by artist familiarity

‣ Simulates (average) user preferences

References
