Next-Song Recommendations for Spotify Playlists Using GPT-2 and T5

Stanford CS224N Custom Project

Carrie Chen
Department of Computer Science
Stanford University
carriec6@stanford.edu

Janice Teoh
Department of Computer Science
Stanford University
jteoh2@stanford.edu

Abstract

Recommendation systems are critical to countless real-world tasks, such as shopping, discovering people you might know in social media applications, and music recommendation systems. In this project, we create a recommendation system for the continuation of Spotify playlists using both T5 and GPT-2 for comparison. The task is as follows: given x existing songs in the playlist (the playlist's prefix), use text data for the songs, including title, lyrical keywords, and textual representations of audio features, to predict one song which would best fit the playlist's mood. We found that our pretrained T5 models produced relatively intelligible outputs, but mostly copied its inputs rather than producing novel songs based on the overarching idea of a given playlist's prefix. On the other hand, our finetuned T5 models produced well-formulated outputs, but largely degenerated into repetitive text, where one keyword would simply repeat continuously until the audio feature section. Finally, our finetuned GPT-2 model showed the most promise, as its inputs were well-formulated, unique, and did not degenerate into repetition.

1 Key Information to include

- Mentor: Yuan Gao
- External Collaborators (if you have any): N/A
- Sharing project: N/A

2 Introduction

Recommender Systems (RS) have become increasingly critical for their use in various fields, including entertainment (eg: Netflix films) and social media (eg: Facebook's recommendation for other individuals you might know).

With Spotify's increasing popularity as both a music streaming platform for listeners and a music distribution platform for artists, the number of songs available on Spotify has spiked dramatically. This influx of data can leave users feeling confused or overwhelmed by the sheer number of songs to select from, which can ultimately drive away users. Thus, Spotify has algorithms for recommending Spotify-created playlists to users based on their listening history, but when it comes to recommending new songs to add to an existing custom playlist created by a user, Spotify's recommendation system tends to recommend songs that users have already listened to or simply other songs from existing artists in the playlist (Germain and Chakareski, 2013). This creates a lack of diversity in playlists, which can become frustrating or boring for users. To correct for these drawbacks and maintain user satisfaction, this paper seeks to create a new Spotify recommendation system based on the songs' titles, keywords from the lyrics, and the overarching audio features (ie: how energetic, happy, or sad a song is) of the existing songs in the playlist.

In our proposed system, we utilize GPT-2 and T5 in parallel for text generation and compare the performance of the two models. The target output for our models is a hypothetical song predicted based on a given list of input songs. Since T5 is a seq2seq-based model, we give the model both an input sequence (the set of existing songs in a playlist) and a set of reference sequences (a set of sequestered songs from the playlist). For GPT-2, we feed the model entire playlists without withholding any songs.

3 Related Work

Music recommendation systems are critical to music streaming software today, such as applications like Spotify, Pandora, and Apple Music. However, since music is such a subjective experience, multiple studies have been conducted surrounding what makes a playlist compelling and—by extension—how to create a music recommendation system which draws upon these ideas.

Studies have noted that a vastly wide range of variables contribute to users' satisfaction with a given playlist. Specifically, Hansen et. al. found that both recent music consumption (in regards to typical genres, for instance) and variables related to user's current listening session (such as the time of day or the device being used) can be used to predict a user's future listening habits (Hansen et al., 2020). Using this idea, they construct a neural network architecture (CoSeRNN) which predicts a preference vector embedding both a user's history and current context at the beginning of a user's listening session. Their model outperforms the current state-of-the-art by around 10 points.

Other state-of-the-art machine-learning based models for playlist generation include the use of LSTM Networks (Long Short Term Memory Networks). In their paper, "Next-Song Recommendations for Spotify Playlists Using Recurrent Neural Networks," Ye et al. create an LSTM-based recommendation architecture for online music recommendations (where the training goal is minimizing the distance between the predicted next-song embedding and user-taste embedding). As their model also outperforms most popular baselines for current music recommendation systems, they find that their model is largely successful (Ye et al., 2019).

Further, as Leong et. al. notes, randomness or diversity in playlists can also be utilized to support rich and novel user experiences, creating a sense of refreshedness and driving user satisfaction (Leong et al., 2006)(Anderson et al., 2020). Thus, other popular techniques for playlist continuation or music recommendation include collaborative filtering (Pérez-Marcos and Batista, 2018), a filtering technique which recommends music from an ordered list of a user's most played songs over a period of time. This list of most-played songs can be extended to include the most-played songs of users with similar listening habits, providing a greater range of potential songs that the algorithm can select from. Collaborative filtering thereby helps users explore a larger range of music (ie: engendering randomness) while still remaining in the same overarching field or genres.

4 Approach

Our approach is to have pretrained T5 models (T5-small and T5-base) and GPT-2 predict the next track in a playlist, given all of the tracks so far. From here on, we will refer to songs as 'tracks', as this is how they are referred to in the Spotify API. We first use the `spacy` library to generate 15 keywords from the lyrics as a way of summarizing the most important features of the lyrics. We then collect the following metadata for each track in the playlist: danceability (how 'danceable' the track is), energy (how energetic the track is), and valence (how happy a track is). Each of these values is a floating point value between 0 and 1, so in order to convert these values to tokens, we categorized values from 0 to 0.33 as "low", 0.33 to 0.66 as "medium", and 0.66 to 1.00 as "high." To generate our input, for each track in the playlist, we concatenated the title of the track, the space-separated list of keywords, and the tokenized metadata (e.g. `[low danceability medium energy low valence]`). We then concatenated the tokens for all of the tracks into one tensor with # separators between each track's data to feed into our model.

T5 is a seq2seq model, so the output from T5 is a string representation of the predicted next track in the playlist, trained to be in the same format as described above. Similarly, as we trained GPT-2 on text generation, the output from the model was also a string representation of the predicted next track from the playlist.

For our baseline, we evaluated the performance of both T5-small and T5-base on our test set, both of which were pretrained on other seq2seq datasets. We conducted our baseline evaluations prior to finetuning the models.

5 Experiments

Input Tracks	Withheld Track
Playlist 1 tracks[0 : n-5]	Playlist 1 tracks[n-5]
Playlist 1 tracks[0 : n-5]	Playlist 1 tracks[n-4]
...	...
Playlist 1 tracks[0 : n-5]	Playlist 1 tracks[n-1]
Playlist 2 tracks[0 : n-5]	Playlist 2 tracks[n-5]

Figure 1. Example of formatted input data for T5 models.

5.1 Data

Spotify's API does not provide access to lyric data, so we curated a dataset of playlist and track information from scratch using the API for the website `Musixmatch`, the same source from which Spotify acquires its lyric data. In total, we scraped data for 200 playlists with lengths ranging from 10 to 100 tracks. For each track in a playlist, we collected the following information: track title, the first 30% of lyrics (provided by the free version of the Musixmatch API), danceability, energy, and valence (provided by Spotify's API). We then processed lyric data for each track and extracted 15 keywords using the `spaCy` library as a way of summarizing the lyrics. Finally, for each track in a playlist, we concatenated these pieces of data together as specified in section 4. For each playlist, we withheld the last 5 tracks as reference sequences for T5 and created one input sample for each withheld track. The format of our input data is shown in Figure 1.

Since GPT-2 is not a seq-to-seq model, we did not withhold any tracks in our training data for GPT-2.

The output of our models (all T5 models and GPT-2) is a generated string of text that corresponds to the title, keywords, and metadata for a hypothetical next track in the playlist.

5.2 Evaluation method

We use the 1-gram BLEU score to evaluate our model's predictions. Specifically, we calculate the 1-gram BLEU score between each playlist's predicted output and the five reference sequences from the withheld tracks, taking the maximum score to be that prediction's score. To evaluate the predicted categorizations for each metadata field (danceability, energy, and valence), we calculate how many out of the three fields had the correct classification compared to each withheld track. For example, if one of the withheld tracks had [low danceability medium energy low valence] and the predicted track had [low danceability high energy low valence], we would give it a score of 2/3. We also take the maximum score between the prediction and a reference track to be that prediction's score.

The 1-gram BLEU score gives us a quantitative measurement of how similar the generated "hypothetical" track information is to the actual reference track information based on single-word similarities in the title and keywords. Our metadata evaluation method also gives us a quantitative measurement of how well our model is able to predict the metadata features of a track that would be added to a given playlist.

5.3 Experimental details

We split our dataset of 200 playlists into a training set of 180 playlists and a test set of 20 playlists; with five samples for each playlist (one for each of the five withheld tracks), we had a total of 900 input training examples and 100 test examples. For T5-base and T5-small, we evaluated the models prior to any training on our task, and then finetuned them on our training set for 3 epochs with a learning rate of 5e-05. For our GPT-2 model, we finetuned them on the training set for 5 epochs with a learning rate of 2e-5. Both models use cross-entropy loss, or log loss, as their loss function.

As our dataset was relatively small, training time ranged from around five minutes for T5-small, to up to an hour for GPT-2 (which was trained without a GPU due to technical roadblocks).

Moreover, because of GPT-2's token limit of 1024 tokens, we had to cut the training data for GPT-2 so that each playlist in the training set only had 10 tracks. This was a stark difference from the original training set, since each playlist in the training set originally contained around 40 tracks on average. The impacts of this change are discussed in section 6 below.

We used the following huggingface models for our experiments:

- T5-base (pretrained, without finetuning)
- T5-small (pretrained, without finetuning)
- T5-base (pretrained and finetuned)
- T5-small (pretrained and finetuned)
- GPT-2 (pretrained and finetuned)

5.4 Results

BLEU Scores We report the average 1-gram BLEU scores for the 20 playlists in the test set below. Here we calculate average by finding the maximum BLEU score across all the target tracks for a given playlist, divided by the total number of tracks in the playlist. The baseline model illustrated in the table below is the pretrained (without finetuning) T5-base model.

	T5-small	T5-base	GPT-2
Before Finetuning	0.33738	0.25756	-
After Finetuning	0.11548	0.35022	0.270812

From a purely numerical perspective, it appears that the non-finetuned T5-small model outperforms the non-finetuned T5-base model, and that the performance of T5-small worsens with finetuning while the performance of T5-base significantly improves. However, we will see in the following section (Section 6) that this is not necessarily the case.

	T5-small	T5-base	GPT-2
Before Finetuning	54.1%	52.8%	-
After Finetuning	-	55.5%	48.7%

Metadata Similarity For each playlist in the test set, we performed a pairwise comparison between the predicted metadata and the metadata for each of the withheld tracks. We calculated the average percentage of metadata fields each model predicted correctly across all of the test samples. Note that if the model were to randomly predict values for the metadata fields, the expected value for this percentage would be 33%, so all three of our models performed significantly better than a random baseline. Data is missing for T5-small after finetuning because the model did not produce correctly formatted outputs and we were unable to extract metadata predictions.

6 Analysis

When we trained T5-small and T5-base, the generated output format varied greatly. Even though we provided reference sequences in the specific format we wanted the model's output to follow, in many cases, the model produced nonsensical, incorrectly formatted, or repeating text. For example, the

expected output from the model is of the form:

```
title:[...].keywords:[...].metadata:[...]danceability.[...]energy.[...]valence.
```

However, many of the predictions started with "keywords: ..." or "metadata: ..." rather than "title: ...", were missing a title or metadata, or had nonsensical keywords such as abbreviations or non-words.

For reference, we show the first two input tracks of one of our test set playlists below and will examine the T5 predictions for this playlist.

Reference Playlist Prefix:

```
title: say my name (feat. zyra) - hermitude remix. keywords: better
gon know corner cause damn letter wanna look knows hands let think girl
dance. metadata: high danceability high energy high valence

title: another day in paradise. keywords: said steal time motion
strong need evening running crash lets gold night mind live moving.
metadata: medium danceability high energy medium valence
```

6.1 T5-base

Pretrained T5-base:

The predicted output for this playlist by the non-finetuned T5-base model was the following:

```
say my name (feat. zyra) - hermitude remix. keywords: saying steal
time motion strong need evening running crash lets gold night mind live
moving.
```

We can see that the prediction clearly copies the title of the first input track, and the keywords are copied from the second track. The prediction also lacks a metadata section.

Finetuned T5-base:

After training the T5-base model on our data, it produced completely degenerate predictions. For a playlist that had no tracks that mentioned sailors, this was the output that our finetuned T5-base produced:

```
title: i'm a sailor. keywords: sailor gon sailor sailor sailor
sailor sailor sailor sailor sailor sailor. metadata: medium
danceability high energy high valence.
```

All of the predictions for the finetuned T5-base were correctly formatted, but they were also all degenerate and repetitive, similar to the example above. Thus, we can see that during training, the T5 models learned to mimic the labels of "title", "keywords", and "metadata" well but did not learn how to properly generate the words for each of these fields. It would require further experimentation to determine if this is due to an insufficient training set or due to our approach and how we structured our dataset.

6.2 T5-small

Pretrained T5-small:

On other other hand, our pretrained (without finetuning) T5-small model was able to produce relatively well-formulated outputs without degenerated text. For instance, for the same playlist mentioned above (containing the tracks "Say My Name" and "Another Day in Paradise," it generated the following:

```
metadata:  high danceability high energy high valence  title:  bo$$.
keywords:  force wear deal love stopping feel heart big
```

Here we can see that although the model did not degenerate into repetitive text, it produced the output in the wrong order. We predict that this occurred because the model was not able to correctly read in the separator token '#'. Furthermore, this model also had issues with simply copying previous tracks, as the following track:

```
title:  bo$$ keywords:  talking pay holla working know gettin momma
money ai maybach brought respect tryna card dollars metadata:  high
danceability high energy high valence
```

occurred in the original playlist that the model was generating new tracks from. Our pretrained T5-base had similar issues in copying track data from the training set as output, while it did not exhibit the degenerate responses of finetuned T5-small and T5-base.

Finetuned T5-small:

Finally, our finetuned T5-small exhibited similar issues as finetuned T5-base, although its degeneration was not quite as severe. Furthermore, all of its outputs were missing the metadata section. For instance, while one of the outputs was:

```
title:  bo$$ title:  a sexy - edx remix.  keywords:  sexy gon know gon
know
```

And another degenerate output was:

```
title:  bo$$ title:  i love you.  keywords:  screams love love love
love love love love love love love love love love love love love
```

Here we see that the degeneration does not occur specifically in all outputs, but still remains in most outputs (while being interjected by other words).

6.3 Finetuned GPT-2

Our finetuned GPT-2 model was also able to learn the general format of the track information, without degenerating into repetition. For a different playlist in the test set containing the following tracks,

```
title:  stay gold keywords:  wide dawn gold stay high eyed stray goes
today moon work sun haze wish loved metadata:  medium danceability high
energy medium valence
```

```
title:  master pretender keywords:  smile knew thought master honey
wait sense', "somethin'", 'feet hurt believed round bring free sticks
metadata:  medium danceability high energy medium valence
```

GPT-2 produced the following output:

```
title:  girl after my grandfather.  keywords:  steal all my silver
money lil brad mi hard to find money be quiet eat order new glass blind
truth may one day follow.  metadata:  medium danceability.  medium
energy.  medium valence
```

Here we see that the model was able to produce a sensible output without degenerating into repetition, like the T5 models illustrated above. Furthermore, its output did not simply copy one of the tracks in the original playlist; the track it generated (including keywords and metadata) was original. Nonetheless, its outputs did not align well with the same 'true' tracks sequestered from the original playlist, leading to a lower BLEU score than average. We believe that a potential reason for this issue was that GPT-2 was not able to train on as many tracks per-playlist as T5, due to the token limit described in the approach section above.

7 Conclusion

7.1 Achievements and Findings

Quantitatively, our finetuned T5-base model achieved the highest BLEU score of 0.35022, despite degeneration. Moreover, it also achieved the highest metadata prediction score, with a score of 55.5%: 20 points higher than the random baseline.

However, out of all of the models, GPT-2 performed the best qualitatively–its outputs were relatively well-formulated, and its keywords did not degenerate into repetitive text. Despite its relatively small training set, it was able to achieve a 1-gram BLEU score of 0.2708, the third highest in all of the model configurations we tested. Although it had the lowest metadata prediction accuracy out of all of the model configurations, we believe this may be attributed to the large size of the model combined with an insufficient amount of training data. Furthermore, it is unclear how effective our evaluation metric for metadata predictions is at gauging a model's performance, given that a collection of tracks may have a wide variety of danceability, energy, and valence levels even if they are part of one playlist. Further experimentation could reveal whether this metric has consistent values when the same model is run on different subsets of the test data.

Overall, we believe GPT-2 is the most promising model out of the three as it produces predictions that are the most intelligible and closest to our desired output.

7.2 Limitations

One of the main limitations of our work is our lack of data. Musixmatch imposes a limit of 2000 API calls per day, so we were ultimately only able to retrieve track data for 200 playlists. This greatly decreased the accuracy of our model, as it did not have enough model to train on to generate accurate predictions. Furthermore, we were unable to clean our data of playlists that have a) tracks in different languages (for instance, some of the tracks were in Korean or Italian), or b) tracks with derogatory terms.

Another limitation we faced was the token limit for the input to GPT-2. GPT-2 was trained on inputs of at most 1024 tokens, and for most of our playlists, this meant truncating the playlist length to around 10 tracks.

The final main limitation we faced was that it was difficult to force our models to follow a specific format. As seen in section 6, it was difficult to evaluate our results due to the lack of uniformity in the output format.

7.3 Future work

Our future work for this project would be comprised of experimenting other types of models and evaluating their performance, as well as experimenting with other aspects of our approach. This would include adjusting the number of keywords we use to represent the lyrics of each track, potentially using text summarization instead of keywords to represent lyrics, expanding our training dataset to include more playlists, as well as possibly including other pieces of metadata for each track and finding a more nuanced way to represent each metadata field (rather than only having three bins of low, medium, and high). We would also perform more experiments that control for input playlist length so we can evaluate whether the length of the input affects the prediction accuracy.

Furthermore, we would like to clean our future dataset of tracks in different languages (limiting the dataset to only playlists with English tracks), and remove tracks with profanity (to avoid offensive or derogatory terms in outputs for keywords or titles, for instance).

In addition, as we predict that the output for GPT-2 was well-formulated despite not corresponding well to the intended true tracks due to a severe lack of training data, future directions could involve creating a new format for our input data which would allow more tracks per playlist in the training set. This could include reducing the number of keywords in a track, or altering batch sizes and end tokens to allow for playlists to override the input size limit.

After experimenting with ways to make our model predictions closer to actual tracks that would appear in a playlist, we could turn our system into a practical tool by formatting each track in our corpus as [title keywords metadata] and use a similarity metric to rank tracks in the corpus based

on their similarity to our predicted track information. This would allow our system to generate real recommendations for tracks to add, rather than information about a hypothetical track.

References

Ashton Anderson, Lucas Maystre, Ian Anderson, Rishabh Mehrotra, and Mounia Lalmas. 2020. Algorithmic effects on the diversity of consumption on spotify. In *Proceedings of The Web Conference 2020*, WWW '20, page 2155–2165, New York, NY, USA. Association for Computing Machinery.

Arthur Germain and Jacob Chakareski. 2013. Spotify me: Facebook-assisted automatic playlist generation. In *2013 IEEE 15th International Workshop on Multimedia Signal Processing (MMSP)*, pages 025–028.

Casper Hansen, Christian Hansen, Lucas Maystre, Rishabh Mehrotra, Brian Brost, Federico Tomasi, and Mounia Lalmas. 2020. Contextual and sequential user embeddings for large-scale music recommendation. In *Proceedings of the 14th ACM Conference on Recommender Systems*, RecSys '20, page 53–62, New York, NY, USA. Association for Computing Machinery.

Tuck Wah Leong, Frank Vetere, and Steve Howard. 2006. Randomness as a resource for design. In *Proceedings of the 6th Conference on Designing Interactive Systems*, DIS '06, page 132–139, New York, NY, USA. Association for Computing Machinery.

Javier Pérez-Marcos and Vivian Batista. 2018. Recommender system based on collaborative filtering for spotify's users. pages 214–220.

Y. Ye et al. 2019. Improved session-based recommendations using recurrent neural networks for music discovery. In *International Conference on Electrical, Mechanical and Computer Engineering (ICEMCE)*.