CS230

Assessing Music Similarity Using Deep Learning

Nicholas Allen
Department of Chemical Engineering
Stanford University
naallen@stanford.edu

Abstract

Music recommender systems are systems that aim to recommend music to a user based on existing preference, using attributes such as listener data, audio analysis, and tag/genre analysis. This project aims to use deep learning to develop the basis of a music recommender system, by developing a neural network that aims to quantify similarities between two artists using clips from tracks produced by each artist. The resulting model attained good performance over the baseline, and provided subjectively suitable results for a set of sampled artists.

1 Introduction

This project aims to assess similarity between two music artists using audio data, where music similarity is an arbitrary metric built from user listening preferences and reviews. Assessing music similarity from audio data is an integral part of music recommender systems, the likes of which are used in many popular music services like Spotify and iTunes, and such a project is well suited for deep learning, as user preference data as well as input data in the form of audio is readily available.[1] This project uses listener data from Last.FM, a popular social media site for music enthusiasts with accompanying software which collects data on music listening habits of users. Samples of audio data are also readily available, particularly as song previews on streaming services such as Spotify. To represent this data in a way that can be easily parsed as input to a neural network, this audio data was converted to 256x64 mel-spectrogram images. These images were used as pairwise input to a siamese neural network, where the siamese portion consists of five convolution layers followed by a densely-connected layer, after which L1 distance was taken between the two network outputs and connected to a final sigmoidal output layer, outputting a similarity score between the two input tracks.

2 Related work

Previous work by Choi et. al. has been done on using deep learning to tag music using audio signal data, as well as to classify music into genres.[2, 3] Their work utilized convolutional neural networks with mel-spectrograms as the input, and used tag and genre data from the Million Song Dataset.[4] This work provided inspiration for solving the problem using CNNs, and using mel-spectrograms as the input as opposed to raw audio data.

Other work done by den Oord et. al. had a similar goal to this project, using usage data to learn a set of latent factors which can be used to group songs.[5] This was similar to my goal, however, the authors of this paper assessed per-track similarity, while I wanted to predict similarity between artists using multiple audio samples from each artist.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

3 Dataset and Features

This project uses a publicly available dataset scraped from Last.FM, which contains over 17 million entries containing data on Last.FM users, artists that these users have listened to, and total number of times they've listened to tracks by a given artist.[6] To keep the dataset smaller, and avoid overfitting to extremely obscure artists, only the top 1000 most popular artists in this dataset were considered. A basic metric for similarity was constructed by counting co-occurrences between artists, i.e. how many users listened to each pair of artists, and then normalizing the resulting matrix column-wise by the number of listeners for each artist. This gave a score, for every pairing between artist A and artist B, of the proportion of people who listen to artist A who also listen to artist B.

This metric was further processed by taking the minimum between the two values for every pair of artists, assigning that value to both artists, then performing scaling along the entire dataset. This resulted in subjectively better results than using the original metric, meaning that obscure artists had other obscure, and stylistically similar, artists more highly scored as similar, rather than popular artists dominating the scores (e.g, most people who listen to obscure rock also listen to Radiohead, but it isn't interesting to suggest Radiohead to an obscure rock fan looking for new artists). Doing this also made the metric symmetrical across pairs of artists, which was a necessity for training the siamese network.

Next, audio data was collected for each artist. This was done by scraping Spotify, using iTunes as a fallback, to collect 30 second preview tracks for every artist in the dataset. Preview tracks were downloaded, and converted to mel-spectrogram images of size 256x64 (256 time steps, 64 frequency bins). Of the 1000 artists considered, audio data could be scraped for 975. Of these, 49 artists were used strictly for the dev set, and 49 artists were used for the test set.

4 Methods

The neural network was structured as a siamese neural network, by feeding two inputs through two identical networks trained with the same weights, then taking the output vector of each, computing the L1 distance, and feeding the distances through a final densely-connected neuron.[7] Mean squared error loss was used, using the Adam optimizer. See Figure 1 for diagram of the network setup.

The siamese component of the network consisted of multiple 1D convolutions along the time domain, interspersed with MaxPool, BatchNorm, and 1D Padding layers. Then, max pooling was done along the temporal axis to flatten the 2D layer into a 1D vector. Global max pooling was chosen to remove any time-dependency from the resulting vector, so that the vector can solely consist of extracted features along the frequency axis.

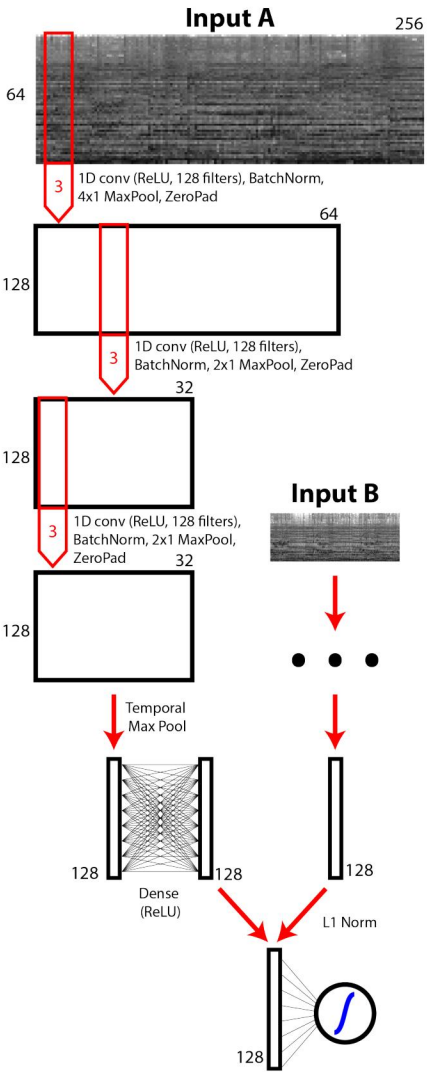

Figure 1: Diagram of the neural network used. Ellipsis represents a duplicate of the rest of the network, for brevity.

To include as much data as possible, the input consisted of all pairwise combinations of (256x64x1) spectrogram images for each audio sample collected for a given artist. For training, the target output for each pairwise combination was the similarity metric between the two artists. When making similarity predictions, a similar technique was used, with the final similarity score being an average of the pairwise track combinations.

A baseline was also constructed, which simply computed the L2 norm between the tracks. While simplistic, this presented a fair challenge to the deep learning-based approach by using the same input data, and also performed respectably on its own, achieving better results than random selection.

5 Experiments/Results/Discussion

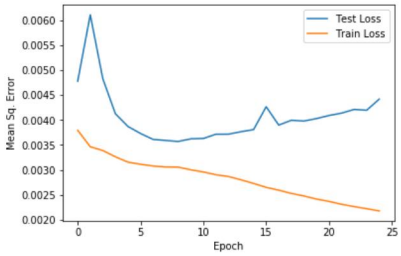

Figure 2: Test and training loss for each epoch.

The model was tuned and tested on the dev set, using grid-search to search for hyperparameters, finding an optimal batch-size of 2048, learning rate of 0.001, and no appreciable reduction in test loss with more epochs. When training was performed on the full training set, however, test loss tended to increase after approximately epoch 5, indicating some overfitting. The final model was trained to 25 epochs.

To evaluate accuracy, a custom ranking-based metric was developed. Although MSE was used as the loss function, this didn't reveal much about how the model worked as a recommender system. To evaluate the quality of recommendations, particularly on the test set artists, an accuracy metric was devised in which the model would be used to generate similarity metric scores between all pairwise artist combinations, then these similarity scores would be used to rank all artists by similarity for each artist. For a given cut-off point, the top most similar artists for every artist would be compared against the ground truth using both unordered accuracy (what proportion of artists are present in both lists) and edit distance. This approach was motivated by the idea that, for the goal of the project, accurately identifying similar artists was more important than accurately identifying dissimilar artists.

The results from this accuracy metric are shown in Table 1. From these results, it appears that the model consistently outperforms baseline, especially at lower values of n. Admittedly, accuracies using this metric are quite low with the model; however, there are multiple explanations for why this could have happened. First, this accuracy metric is somewhat arbitrary and is based on co-listener data, which may not accurately reflect the true musical similarity between artists. To give an example, my metric labels the artists Radiohead and The Beatles as being extremely similar to one-another, but in reality, the artists are rather musically dissimilar, and their common co-occurence among listeners is more due to the two artists both being popular rock bands. Hypothetically, a band that was much more obscure but aims to completely replicate the sound of The Beatles would have had a lower similarity score by virtue of it being more unknown. While this was partially the goal of the project, that is, to recommend artists based on preference rather than strictly audio similarity, it did make it difficult for the model to extract relationships between audio data when the metric may not have only reflected similarities present in the audio data, but also taking into account other factors such as general popularity of an artist.

n	Edit Distance	Unordered Distance
3	0.1224 (0.0748)	0.1293 (0.0748)
5	0.1469 (0.1143)	0.1796 (0.1388)
10	0.1796 (0.1653)	0.3163 (0.3020)
25	0.1633 (0.1543)	0.5731 (0.5543)

Table 1: Accuracy scores of model trained for 25 epochs, with batch size of 2048, using selected accuracy metrics. n gives the number of top artists considered, out of a total test set size of 49 artists. Baseline values are shown in parentheses, for comparison.

Another issue that might have arisen is the variance between tracks for a given artist, and using the same similarity metric as the target. Artists may change their sound through their careers, and even between different tracks in the same album. In the absence of more data, I used the same similarity score between artists as the target for all pairwise combinations between the tracks for any two artists. For artists that vary a lot in their musical style, this would have made it difficult for the model to extract similarities between drastically different music combinations when the target output is the same value for each.

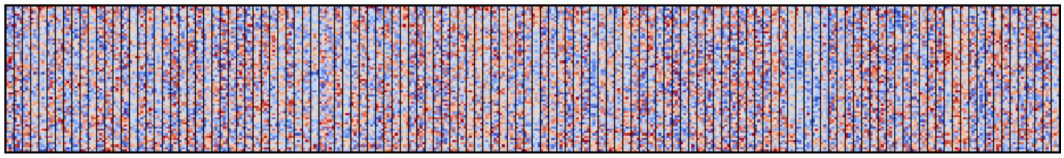

Figure 3: Activations of all filters in the first layer of the model. Color scale goes from blue (low) to red (high), with low frequencies appearing towards the top of the image.

To visualize how the model was making decisions, I plotted heatmaps of all filters in the first layer of the model, shown in Figure 3. Most appear to be noisy and difficult to evaluate, but a selection of filters do seem to correspond to specific audio patterns, shown in Figure 4. Filters 4 and 64 appear to strongly respond to a sharp bassy sound, perhaps corresponding to a bass beat in hip hop, or a kick drum. Filter 99 appears to respond to a complex pattern immediately followed by silence, which might correspond to strongly defined beats in a track.

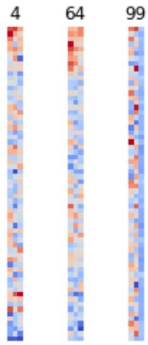

Figure 4: A selection of interesting filters from layer 1 of the model.

Lastly, the model was qualitatively evaluated by selecting three musically distinct artists that I am personally familiar with (The Beatles, In Flames, Tupac) and using the model as well as the baseline to generate a ranking of the top 5 most similar artists (from both the training set artists and the test set artists) for each of these. While both the model and the baseline had significant deviations from the ground truth, this analysis revealed interesting patterns in how the model makes decisions that aren't reflected in the previously reported accuracy scores. This data is displayed in Table 2.

The ground truth for The Beatles seems to consist of popular rock and pop rock artists, which have varying degrees of musical similarity with The Beatles. This illustrates one of the issues brought up earlier regarding my similarity metric, which is that it is also heavily influenced by confounding associations between artists, and doesn't strictly represent musical similarity. The baseline performs quite badly in this case, with the artists coming from genres like hip-hop and new-age, having little to do with the classic rock style that The Beatles played. Interestingly, while the model was quite off with predicting the exact artists in the ground truth, it came close to predicting the style of The Beatles, mostly predicting indie rock and alt-rock bands.

In Flames is a reasonably well known metal band, which provides an interesting example as a band whose musical style has shifted considerably through their career (from a harsher death metal sound to a more accessible alternative metal). The ground truth here is split between the two styles, with Dark Tranquility and Children of Bodom being similar to the older In Flames style, and Disturbed/Slipknot being similar to their new style. Interestingly, the artists that the model predicts as being similar to In Flames tend to lean towards more alternative metal styles, particularly Mudvayne and Marilyn Manson, which are closer to In Flames' newer style. Looking at the input data, I noticed that the sample audio tracks I scraped for In Flames are all from their newer albums, which explains the model's predictions. The baseline predictions here seem to be much closer to In Flames' old musical style, which could be explained by the low diversity of metal artists in the dataset.

Finally, I evaluated the model and baseline using Tupac, an influential west coast hip hop artist. The similar artists show a set of other old-school hip hop artists, including rivals and artists affiliated with Tupac. The baseline predictions here are completely off with regards to musical style, consisting of electronic and indie rock artists. The model predictions, however, are quite good, predicting primarily hip hop artists. Interestingly enough, Akon and Jay-Z are also artists that were directly affiliated with Tupac but don't appear in the ground truth.

Although this analysis shows that the model is often quite bad at predicting the ground truth, it also shows that the model makes interesting and relevant predictions quite often, especially with regards to musical style. In some regard, this is a success. This indicates that the model and neural network architecture I designed is capable of extracting musical features and at least grouping artists loosely

by stylistic similarity, solely from spectrogram data, despite the potential flaws in the similarity metric.

6 Conclusion/Future Work

This report outlined the use of a CNN-based siamese neural network to assess similarity of music artists based on a similarity score derived from listener data. Although the model attained a good mean squared error loss, on both training and test data, the model tended to inaccurately rank artists compared to the ground truth. Observing predictions for a subset of artists, however, showed that the model was fairly good at predicting subjectively musically similar artists despite this divergence from the ground truth. Future work could focus on further refinement of the similarity metric and input data, as that is likely what limited the model. The similarity scores I obtained were between artists, but perhaps similarity between tracks would be better suited for this kind of application, as well as incorporating review data in addition to listener data. Additionally, having access to a larger set of audio samples, to have a more representative set of an artist's sound, would also help in improving the performance of the model.

The Beatles		
Ground Truth	Model	Baseline
radiohead	the 69 eyes	massive attack
coldplay	leonard cohen	dj shadow
bob dylan	cat stevens	klaus badelt
the rolling stones	death cab for cutie	era
led zeppelin	built to spill	enya
In Flames		
Ground Truth	Model	Baseline
children of bodom	rhapsody of fire	immortal
slipknot	creed	soilwork
disturbed	mudvayne	converge
dark tranquillity	marilyn manson	dark tranquillity
iron maiden	cradle of filth	thrice
Tupac		
Ground Truth	Model	Baseline
nas	akon	orbital
snoop dogg	jay-z	flying lotus
50 cent	50 cent	mr. scruff
notorious b.i.g.	lil wayne	shpongle
dr. dre	the pussycat dolls	king crimson

Table 2: Similar artist rankings for a selection of artists, comparing ground truth, model output, and baseline output.

7 Contributions

This project was entirely done by Nicholas Allen.

References

References

(1) Dieleman, S. Recommending music on Spotify with deep learning., `http://benanne.github.io/2014/08/05/spotify-cnns.html`.

(2) Choi, K.; Fazekas, G.; Sandler, M. Automatic tagging using deep convolutional neural networks. *arXiv preprint arXiv:1606.00298* **2016**.

(3) Choi, K.; Fazekas, G.; Sandler, M.; Cho, K. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp 2392–2396.

(4) Bertin-Mahieux, T.; Ellis, D. P.; Whitman, B.; Lamere, P. In *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

(5) Van den Oord, A.; Dieleman, S.; Schrauwen, B. In *Advances in neural information processing systems*, 2013, pp 2643–2651.

(6) Celma, O., *Music Recommendation and Discovery in the Long Tail*; Springer: 2010.

(7) Koch, G.; Zemel, R.; Salakhutdinov, R. In *ICML Deep Learning Workshop*, 2015; Vol. 2.

(8) Chollet, F. et al. Keras., `https://keras.io`, 2015.

(9) Martın Abadi et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems., Software available from tensorflow.org, 2015.