Geospatial Music Regression - Finding an Artist's Location Based on their Music

Christian Lee, Andrew Hu

Introduction

Music, as an aspect of culture, is undoubtedly greatly influenced by (and some might even say a product of) it's location of origin. This characteristic is easily distinguishable to most people, whether by the unique rhythms of Latin American songs or the instrumentation and harmonies of eastern oriental music. When looking at modern western popular music, often even past the blending of cross-cultural references, it's possible to identify either a regional genre or origin of a song. The goal of this project is to create a machine learning application that can take a song and estimate the artist's location. Code used for the experiments is available at https://github.com/andrewhu/wav2loc.

Initially, the original idea of the project was to generate music based on two input coordinates using Google's Magenta library built on Tensorflow; however, as we progressed further into this project, we slowly began to realize that modifying the architecture of Magenta's complicated variational autoencoder was unfortunately out of scope for the class. As a preliminary project to get us familiar with the library, we did, however, train a model using Magenta's Performance RNN over ~600 jazz piano performances scored as MIDI files that were scraped from the internet and preprocessed to isolate piano tracks within the arrangements. The model showed fairly good results which can be seen here: https://www.youtube.com/watch?v=lFM1nwgqqIw

Dataset

Most of the work done in this project was centered around a subset of the Million Song Dataset (http://millionsongdataset.com/) which provides metadata and analytical information for nearly a million songs. For our project, we needed audio files and their corresponding artist's latitude and longitude coordinates. Because the matched audio files had to be pulled from the 7digital API and we were unable to get an access key in time, we got the data by contacting Colin Raffel, who created the Lakh MIDI Dataset which has a subset of the audio files from the Million Song Dataset (https://colinraffel.com/projects/lmd/). From there, we matched audio files to their corresponding metadata files via md5 hash, and cross-referenced the Million Song Dataset's list of artists for which they had their location metadata.

Methodology

Feature Extraction

Due to the nature of the Nyquist-Shannon sampling theorem and the digital signal processing of audio files, it would have been nearly impossible to purely use the raw audio data as a means of input at its sample rate and bitrate, else there would have been too many raw inputs to realistically train. In order to counteract this, we selected two features that represent the audio visually over the frequency-time domain, which allowed us to use these audio files post-processing as image inputs. The two features we selected were:

- **Spectrogram**: A visualization of audio waveforms across the frequency-time domain
- **Mel-Frequency Cepstral Coefficients (MFCC)**: A set of approximately 12-20 values that define the spectral envelope of an audio segment. [4]

Research has shown that the MFCC can benefit from a dimensionality reduction from a Principal Component Analysis, which was provided by Echo Nest as a column in the Million Song Dataset instead of having to extract it through Librosa. [1][2]

Model Architectures

For our experiments, we decided to try two models: a basic linear regression model and a convolutional neural network. We chose a linear regression model to (a) serve as a baseline to compare our methods to, e.g. “how well would a simple model perform?”, and (b) to use a model that we learned about extensively in class, which we had strong theoretical intuitions for. Then we wanted to choose a more powerful model to see if the increased model capacity would lead to better results. We decided to use the reference convnet implementation from the PyTorch CIFAR-10 tutorial (https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html) in order to reduce the complexity of choosing model architectures, picking hyperparameters, etc. To repurpose the classifier as a regression model, we added a hyperbolic tangent activation at the end of the model (to restrict the range of outputs to valid latitude/longitude values), and used L2 loss (MSE in PyTorch) instead of cross entropy. Since latitude and longitude values range between [-90,90] and [-180,180], respectively, we rescale both values to be between -1 and 1. Both models were trained with vanilla Stochastic Gradient Descent with a learning rate of 1e-4.

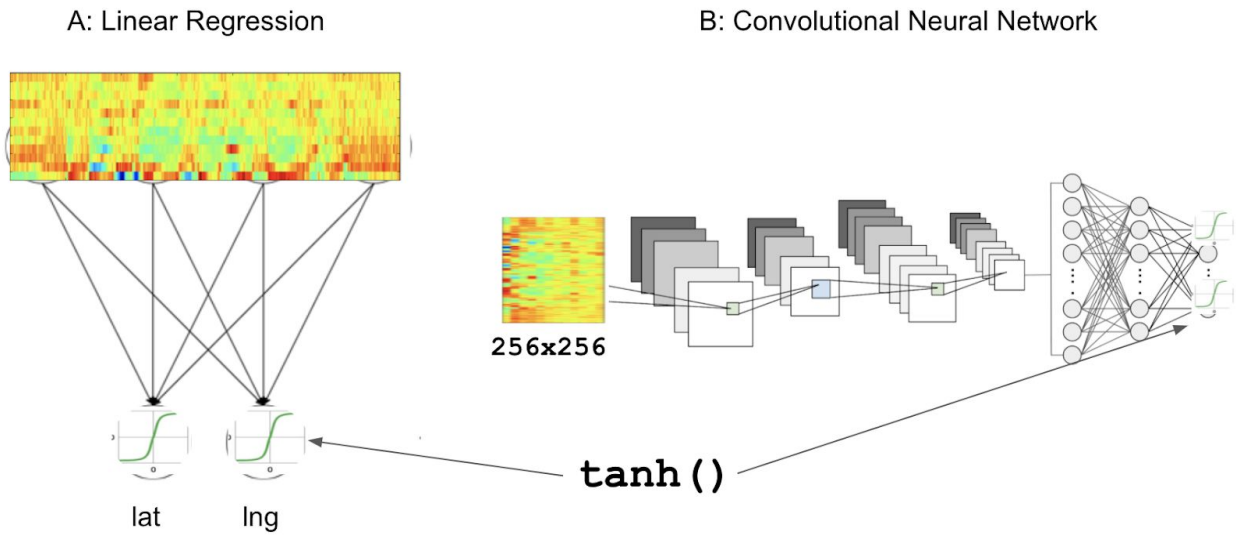

Figure 1: We tested two models, a linear regression (left) and a convolutional neural network (right).

Results

Training loss curves

Shown below are the train/test loss curves of our models during training. Training was done in Colab with their provided P100 GPUs, and both models converged after just a few minutes of training. The convolutional neural network barely outperforms the linear regression model, though the difference is insignificant (when latitude and longitude are rescaled to their original magnitudes, the difference is less than 0.2 degrees latitude/longitude).

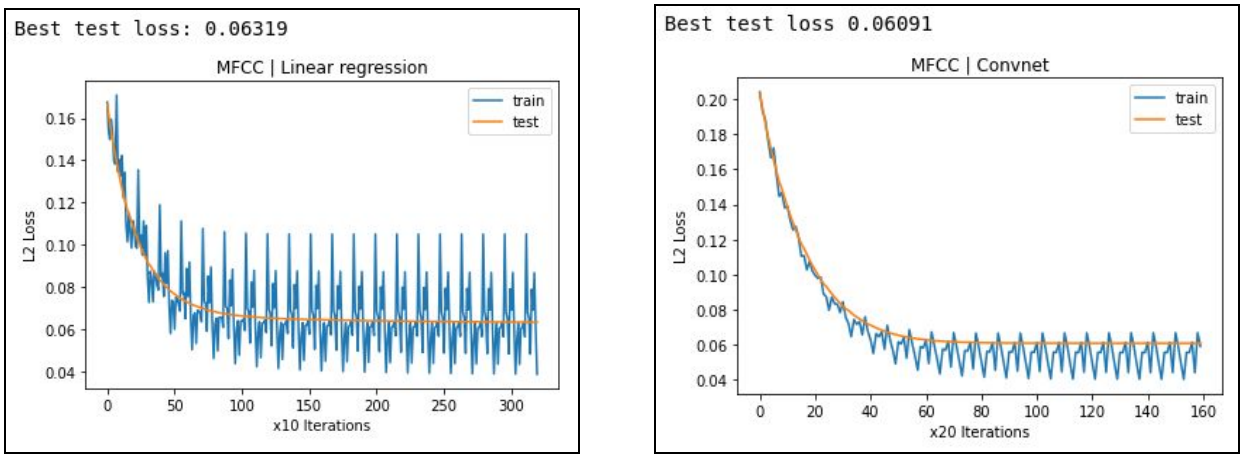

Figure 2: Loss curves for the linear regression model (left) and convolutional neural network (right).

Evaluation

Figure n: Results from evaluating test audio tracks from the convolutional neural network model. On the left is a track from Peru, and on the right is a track from France. Note how the model tends to predict locations in the middle of the Atlantic Ocean. More on this in the discussion section below.

Since the loss curves are hard to interpret by themselves, we decided to plot the longitude and latitude values (rescaled) with Google Maps with test songs that weren't in the original dataset. Disappointingly, all of the songs we evaluated the model on end up being predicted in the middle of the Atlantic Ocean. In addition, the song from Peru is closer to France than the song from France is (and vice versa). So not only did we see that the model predicts locations that aren't on land, but also that it doesn't seem to be placing tracks close to their original origin.

Discussion

It's clear from the train/test loss curves that our model is learning *something* instead of just overfitting to the training set, and we're even seeing a nice smooth convergence. But why are results almost always in the Atlantic Ocean?

Figure n: Visualization of artist locations from the Million Song Dataset. Notice how most of the artists are either located in North America or Europe.

To answer this question, we'll take a look at the visualization of artist locations, in which we see that most of the artists are located either in North America and Europe. From this, it starts to become evident what our models are actually learning: because we're regressing latitude and longitude and therefore assuming a linear relationship based on raw audio, we believe the best our models could do was to predict points that are essentially equidistant from the points in the training set. And since the distribution train/test splits are likely very similar, this explains why our models didn't overfit to the training set, simply because they didn't have enough capacity to.

However, this doesn't entirely explain why the convolutional neural network model had the same tendency, since the architecture we used has the ReLU activation, which should allow the model to learn complex non-linear relationships. We think that our model is too simple to learn the task at hand well.

Conclusion

One modification to the project that would have been worth exploring is more in-depth feature engineering. A few features used in other audio machine learning applications such as music genre classification and speech recognition tend to use specific spectral qualities of audio files such as zero crossing rate, spectral centroid, and chroma features. Extraction of rhythmic qualities have also been shown to be able to identify dialects in speech, which may also prove useful in identifying the geolocational origin of audio files [3]. A kernel transformation to account for the spherical nature of geospatial coordinates may have also provided a more accurate model.

Another possible modification is to structure the problem differently, as a core issue with our approach is the assumed linear relationship between musical quality and locational coordinates. This problem may have been better posed as a classification than a regression by mapping the coordinates in the dataset to a country code. A differential model better suited for sequential data, such as an RNN, may have also shown an improvement in performance.

Overall, this problem proved to be a more interesting and complex challenge than we had initially expected. Since the field of machine learning is constantly putting out results that make us think "wow, I didn't think something that simple that would work", we wanted to try something that we weren't confident would work. Although we didn't achieve amazing results, we're glad that we at least tried. Looking forward, we plan to continue work on our generative jazz piano idea as it proved to show good results.

References

[1] A. Winursito, R. Hidayat and A. Bejo, "Improvement of MFCC feature extraction accuracy using PCA in Indonesian speech recognition," 2018 International Conference on Information and Communications Technology (ICOIACT), Yogyakarta, 2018, pp. 379-383, doi: 10.1109/ICOIACT.2018.8350748.

[2] A. Winursito, R. Hidayat, A. Bejo and M. N. Y. Utomo, "Feature Data Reduction of MFCC Using PCA and SVD in Speech Recognition System," 2018 International Conference on Smart Computing and Electronic Enterprise (ICSCEE), Shah Alam, 2018, pp. 1-6, doi: 10.1109/ICSCEE.2018.8538414.

[3] McGowan, R., & Levitt, A. (2011). A Comparison of Rhythm in English Dialects and Music. Music Perception: An Interdisciplinary Journal, 28(3), 307-314. doi:10.1525/mp.2011.28.3.307

[4] Mitrović, Dalibor, Matthias Zeppelzauer, and Christian Breiteneder. "Features for content-based audio retrieval." Advances in computers. Vol. 78. Elsevier, 2010. 71-150.