Please Hire Us Spotify: Simple Song Classification Using Lyrics

M. Paredes Sepulveda, A. Chew, M. MacFarquhar

Abstract

Music recommendations systems have been a problem for decades. Most systems use a mix of user data and song tagging to create recommendation models for users. With the advent of improved machine learning algorithms, work has been done toward recommending music based on actual music content, using waveforms or extracted features. In this paper, we propose a similar content based approach to a simplified problem, genre classification, using song lyrics as a data source. We show that acceptable accuracy can be obtained using this system, which could be used as a jumping off point for more complex music data classification.

I. Problem

With the growth of music streaming services over the past decade has come a new-found demand for methods of grouping, evaluating, and recommending music. Streaming services are heavily invested in allowing users to discover music that aligns with their unique preferences. One of the most defining features of music similarity, and consequently a successful indicator of shared user preference, is genre. Music genre, however, presents a complex classification problem. Thankfully, music enthusiasts and streaming services alike have generated a wealth of manually labeled music by genre for data analysis. In addition, most songs have similarly associated lyric transcriptions.

While tempo, instrument, and release year can be intriguing indicators of genre, more defining are lyrical content, context, and structure. This presents an intriguing problem as lyrics hold spatial and contextual value in addition to meaning derived by each individual word. Incorporating a word embedding data processing method, we can consolidate words into individual nodes for such analysis. Using an RNN we hope to draw genre indicating information from the contextual relationships of lyrics. Conversely, using a CNN we aim to learn extrapolated spatial information from lyrics in relation to song progression. Furthermore, we predict that one model may perform better on some genres than the other. To exploit the potential varying performances across genre class, we will evaluate an ensembling of both CNN and RNN models.

For the purposes of this study we have chosen the five most commonly listened to music genres which include sufficient lyrics for analysis (thereby excluding classical and jazz). These genres are rock, pop, country, hip-hop, and metal. The similarity of these song genres from a lyrical perspective, particularly metal with rock, and pop with hip-hop, make this task extremely challenging. Current (2018) state of the art learning-based models covering a suite of seven music classes (pop, rock, hip-hop, techno, blues, vocal, reggae) have only been successful in achieving up to 65% test accuracy, with traditional machine learning approaches failing to exceed 59% accuracy.

II. Dataset

The MetroLyrics Kaggle dataset [2] provided us with over 380,000 songs of varying lyric length and genre. The data set needed extensive cleaning and pre-processing for us to use it in training our models. First, we needed to remove all songs with the genre "Other" or "Not Available" so we would only have data that had labels of specific genres. We then cut our dataset to contain only lyrics with lengths 200-3000 characters which maintained about 70% of the data. The dataset had some non-English songs, to remove these, we used python's langdetect module –which is a port of Google's language detection code– to remove any song that was not English. We then cut out the songs with genres indie, R&B, electronic, jazz and folk since there were too few examples of those genres to properly train our network. At this point, our data only consisted of Rock, Pop, Country, Metal and Hip-Hop. Our data was still unbalanced, for instance there were 94,000 Rock

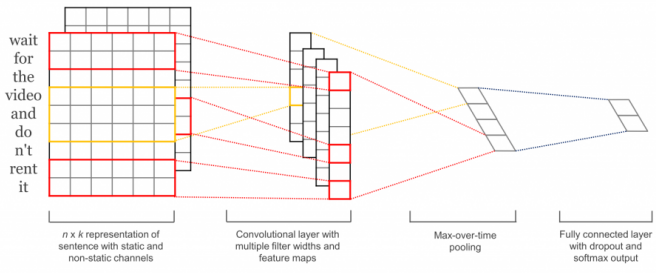
songs and only 13,000 Country songs. The final step was to balance the genres, we took a random sub-sample of 10,000 songs from each of the 5 genres.

After we obtained the pre-processed dataset, we loaded loaded the Facebook fastTest [3] pre-trained word embedding model. This is a skip-gram trained word representation model that learns vectors of length 300 for each word in a corpus. Each feature in an embedding vector represents some learned attribute that can be compared directly to that same feature in other embeddings. Using these embeddings, we created 2D sentence embeddings by stacking the embeddings for the words it contained, giving us k x 300 sentence embeddings, where k is the length of a song. Each song had different lengths. We then split the dataset into train and test, 80% of the data randomly placed into train, and 20% to test. In order to provide our model with standard 100 x 300 sentence embeddings and perform data augmentation, we created a PyTorch Dataset class that, when indexed using __getitem__, randomly crops the indexed sentence to 100 x 300. This gives us many more unique examples in our dataset and allows us to use a standard CNN that takes fixed size images for one of our models.

III. Methods

i. CNN

For our CNN, we first created 100 filters for 3 different kernel sizes of 3,4,5 for a total of 300 filters. We then took the max of each of these 300 filters and concatenated them into a flattened layer with 300 nodes. We added a dropout layer with p=0.5, then we added 3 fully connected layers First:(300 $\rightarrow$ 128), Second:(128$\rightarrow$ 64), Third:(64 $\rightarrow$ 32) each with a relu activation. Finally we have an output fully connected layer with a softmax activation (32 $\rightarrow$ 5), 5 because there are 5 classes in our data. We structured our CNN based on the one in the paper[1].



In the paper, they recommended that we use kernels of sizes (3,4,5) with 100 filters each, a p=0.5 dropout and a minibatch size of 50. We experimented with different kernel sizes (5,6,7) and (7,8,9) and we found that these models did not perform as well as the (3,4,5) model. We also tried to use only 10 filters per kernel which performed poorly compared to the 100 filter model. In deciding the number of fully connected layers, we originally only had 2 (300 $\rightarrow$ 32) and (32 $\rightarrow$ 5), however this caused the training curve to have trouble getting to an optimal accuracy and it would max out at around 56% adding more linear layers allowed our model to fit the data more accurately and increased our training accuracy as well as our validation accuracy.

ii. RNN

Our RNN consists of one LSTM layer which receives the word embeddings of the song lyrics and has an output of 100 nodes and then we use a fully connected layer (100$\rightarrow$5) with a softmax activation to get the probabilities of our classes. We did not base this model on any pre-existing model. We used a standard barebones RNN as this type of network provides good performance on basic NLP tasks because it has unlimited context length. [5]

iii. Ensembled

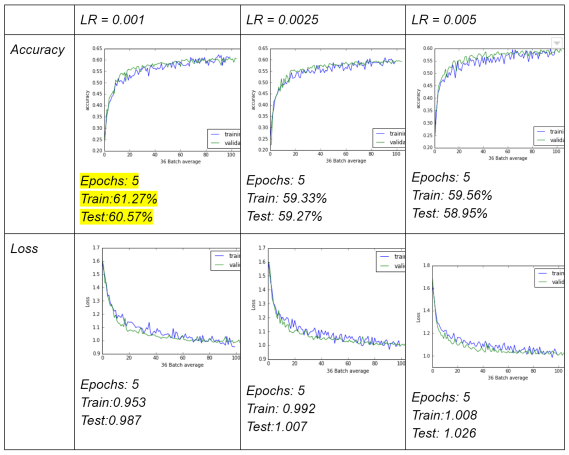
We created an ensembled model of our top performing network from RNN and CNN respectively. This ensemble would take a weighted average of our results from the two individual networks and combine them to get a better result. We trained this ensembled model on the train data for one epoch so that it could learn the weights for the average.

IV. Results

We found that training for more than 5 epochs gave us negligible improvement and eventually a fall in the test accuracy. For that reason we decided to train each model for 5 epochs.

i. CNN

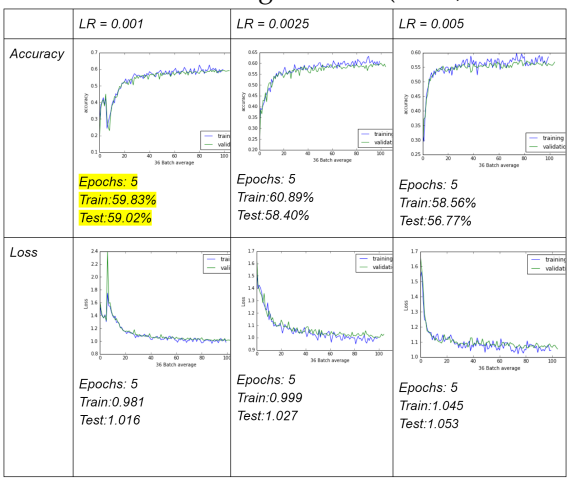
Our optimizer was Adam and we had a decay of 1e-6. In the CNN model we explored 3 different learning rates (0.001, 0.0025, 0.005)



For the CNN we had our best results with a learning rate of 0.001

ii. RNN

Our optimizer was Adam and we had a decay of 1e-6. In the RNN model we explored 3 different learning rates (0.001, 0.0025, 0.005)



For the RNN we had our best results with a learning rate of 0.001

iii. Ensembled

The ensembled model outperformed the other two models slightly. RNN: 59.51% (1.010 loss) CNN: 60.23% (0.987 loss) Ensembled: 61.14% (0.964 loss). This slight improvement shows us that generalization accuracy is not better across the domain of classes for one network over the other and that by combining the strengths of these two models, we were able to achieve a 1% increase in accuracy.

V. Discussion

i. How Did We Do?

Our results show that using a simple lyrics-based classification model can yield surprisingly accurate results for 5-way classification. If our models had not learned the necessary features to classify the images, its accuracy would have been around 20%. However, we were able to learn about 3x better accuracy, meaning that our model was able to differentiate between the 5 song classes. Our accuracy of 61.14% was up to par with state of the art networks on these types of NLP multi-class classification problems. A state of the art network used in this paper[4], achieved a 59% accuracy on a 7 class dataset for classifying songs by genre. They had more classes to classify between, however, their genres were more diverse than ours and therefore should have been easier to classify than our genres were.

ii. Human Performance

We expect that humans will have a difficult time classifying between these 5 genres. Hip-Hop and Pop have similar lyric structure as well as Rock and Metal. Given just the lyrics without the cadence or accompanying tune, it will be a very difficult task for humans to classify the song genres given the lyrics. We can see in this paper [6] that is indeed difficult for humans to classify these genres, with their performance ranging from 32% to 95%

iii. Why Did The Models Do Well?

The CNN approximated an n-gram language model, in that it used local context to extract features from a sentence. These models have historically worked reasonably well in NLP tasks. In addition, the nature of the sentence embeddings, where each column corresponds to one consistent feature, made it easier for the model to learn about the semantics of the sentences, instead of focusing solely on word frequency. The RNN was able to reason about the entire sentence, not just an n-gram window, which gave it an advantage over the CNN. In the case of the ensemble model, by combing the RNN and the CNN we were able to let our new model use the strengths of each individual model to increase performance. For example, if the CNN is superior at classifying Rock vs. Metal and the RNN is superior at classifying Pop vs. Hip-Hop, then the ensemble model will learn to weigh the CNN heavier for Rock vs. Metal and the RNN heavier for Pop vs. Hip-Hop.

iv. Why Did The Models Not Do Well?

The classes were very similar as most song lyrics are similar, and include the same words. This is potentially useful because songs in each genre contain similar words, like pop music being about love, or country music lyrics being about about blue jeans. It's clear, however, that a word like "love" could show up as a high frequency word across many genres. A neural network would hopefully learn to deal with these words, and to omit the common words between classes, but it's hard to say if this occurred without testing using a dataset that omits these words to check for performance improvements.

v. What Could We Improve?

We could try doing some contextual data augmentation such as release year and tempo to improve the RNN, since the CNN probably can't be improved much in this way. Also, we can try to train a model to remove common words and phrases that don't help classification, similar to performing PCA on the dataset.

References

[1] [Yoon Kim, 2009]
New York University
Convolutional Neural Networks for Sentence Classification
arXiv:, 408.5882v2.

[2] [Gyandera Mishra, 2016]
Kaggle
380,000+ lyrics from MetroLyrics

[3] [Facebook Research, 2018]
Facebook
fastText

[4] [Hareesh Bahuleyan,2018]
University of Waterloo
Music Genre Classification using Machine Learning Techniques
arxiv:,1804.01149v1

[5] [Yin, Kan, Yu, Schutze, 2017]
IBM Research
Comparative Study of CNN and RNN for Natural Language Processing
arxiv:,1702.01923v1

[6] [Seyerlehner, Widmer, Knees, 2010]
Dept. of Computational Perception, Austrian Research Institute for AI
A Comparison of Human, Automatic and Collaborative Music Genre Classification and User Centric Evaluation of Genre Classification Systems