Gucci Gang and the Rise of Mumble Rap

CS 221 Final Project - Cristobal Sciutto - Computer Science Dept.

ANN and RNN approach to predicting the commerical success of hip-hop songs based on lyrics and Billboard charts

Introduction

With the recent surge in popularity of artists such as Lil Pump, Desiigner, and Lil Yachty, many "hip hop heads" have found themselves confused. Hip-hop has historically focused on spitting bars, with artist such as Eminem, Kendrick Lamar, and A Tribe Called Quest reigning as the greatest lyricists of all time. Nevertheless, as of late, the songs most played according to Billboards charts and Spotify rankings show very little lyrical prowess. Just look at the lyrics of "Gucci Gang", the current #3 song on the Hot 100 Billboard. In an attempt to understand this transition, I used an ensemble of an ANN and RNN to predict commercial success of tracks based on lyrics.

Approach

The task being tackled is predicting the popularity of hip-hop songs based on their lyrics, using Billboard charts as a heuristic for commerical success. Lyrical features of songs in the three major categories of semantics, structure, and vocabulary were extracted. The ideal end model that would be used was an ensemble of a recurrent neural-network (RNN) with a traditional neural network (ANN). The ANN would take into consideration features related to the totality of a song, such as lexical richness and song structure, while the RNN would be able to extrapolate on the sequentiality of lyrics, e.g. repeated catch lines.

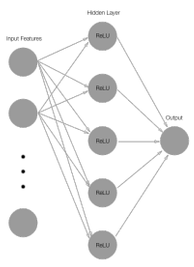


Figure 1: ANN

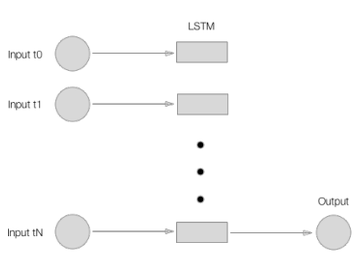


Figure 2: RNN

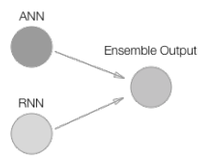


Figure 3: Ensembled Model

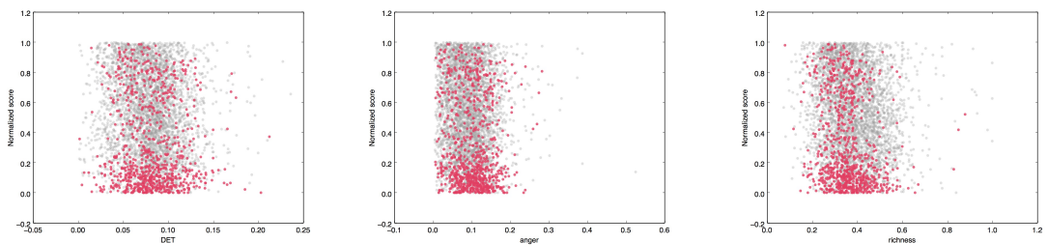
The initial baseline is the average score of songs by the same artist. A simple Linear Regression was used as a second baseline for the more complex approaches. Then, sucessive layers were added to the model to test for correlations between features. For the ANN and DNN, tests were run on different combinations of hyperparameters and regularizations to minimize both the mean squared error and the mean absolute error. Then, an RNN was trained on the sequential data. Finally, an ensemble composing the best ANN and RNN was to be used.

Feature Extraction

The key to the success of the predictor is making the features as expressive as possible, given the relatively small data set of Billboard charts since 1995, resulting in 4479 songs. The features used were:

	Gucci Gang	m.A.A.d City
Semantics		
Distr. Emolex cat, e.g. anger:	0.27	0.12
Distr. NLTK POS universal, e.g. DET:	0.06	0.12
Structure		
Distr. verse types, e.g. verse:	0.2	0.44
# of stanzas:	6	10
# of lines:	54	133
Avg. lines / stanza:	71	113
Avg. words / line:	7	8
Vocabulary		
Total word count:	431	1133
Non-word / total:	0.25	0.31
Vocab. salience (if-idf):	0	0
!-count:	1	17
?-count:	9	15
#-count:	0	2
% common words, e.g. love:	0.92	0.0

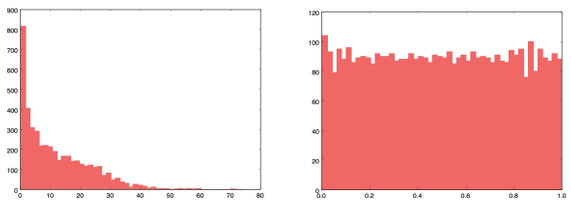
For the sequential dataset for the RNN, the semantics and vocabulary features were evaluated on a per-line basis, composing a sequence.



Figures 4, 5, 6: DET, Anger, and Richness feature distribution. Temporal test set is red.

Scores are calculated proportional to the peak position of the song on the Billboard charts and the time spent on the charts. The number computed is then percentiled in order to evenly distribute values. Note that this imposes linearity in quality of songs.

$$y_{raw} = r_b \cdot t_b = \frac{(101 - \text{peakPos})^2}{10000} \cdot t_b$$



Figures 7, 8, 9: Score Calculation, Pre and Post Normalization

Infrastructure

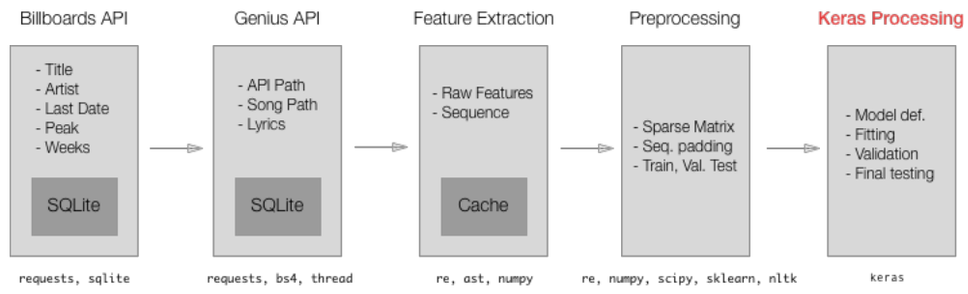


Figure 10: Data Pipeline

The data pipeline followed is roughly illustrated above. Summarized, data was scraped from both the Billboard charts and Genius lyrics website. After tokenization and feature extraction, the raw features were cached. Then, when training, a preprocessing phase occurs in which the data is manipulated to correctly satisfy the Keras model. For recurrent training, sequences of fixed length must be built. The first 200 lines of songs were considered, and those who didn't reach the limit were padded. From there Keras with a TensorFlow backend was used to implement and train the models.

Results and Analysis

The dataset was split in two distinct ways: temporal and random. For the temporal split, songs released after 2015 were used as the test set, while the older songs were used to train and validate (70%, 30%). On the other hand, I simply split the dataset randomly. For each method used, the best MSE and MAE for both data splits are registered below:

	TEMPORAL		RANDOM	
	MSE	MAE	MSE	MAE
LR.	0.13	0.32	0.10	0.30
ANN.	0.13	0.32	0.10	0.29
DNN.	0.13	0.33	0.11	0.30
RNN.	0.13	0.33	0.12	0.31

Unfortunately, all techniques attempted failed at properly predicting the success of lyrics, leading to the abandonment of the final ensemble approach. For comparison, a 0.25 MAE is equivalent to always guessing the mean, and we are failing to breach this minimal threshold. Noticeably, all models results in nearly identical errors for the test set, a clear signof underfitting. The baseline of merely averaging all other songs by the artist results in a better 0.27 MAE.

There are two main explanations for the failure: underfitting or randomness. Our dataset is a mere 4500 samples, of which only around 3000 are used to train. Furthermore, the lyrical domain is not the sole domain related to music, and therefore cannot fully express a song. This makes underfitting likely, yet potentially solvable. Nevertheless, there might simply be no correlation between song features and their commercial success, which seems to be justified by the feature distribution shown above. Both Eminem and Future have had songs top the Billboard charts, despite extremely different styles (and critical acclaim). It is possible that we can't reduce down music to an array of features.

For future developments, it will be interesting to incorporate an even larger number of features, even outside of the lyrical domain, e.g. rhymes, bpm, instrumentation. Furthermore, a detailed parsing of lyrics would reduce the sparsity of data, possibly combating the underfitting.

Acknoledgements and References

I would like to express gratitude to my CA Michelle Mei for her feedback on the previous phases of the development of this project. Furthermore, I would like to thank my roommate Vishnu Sarukkai for helping identify some core problems in the project and proposing solutions. This project would not have been possible without several open-source tools avaiilable: Keras, Scikit-Learn, Numpy, guoguo12's Billboard API, and the Genius API

[1] Rudolf Mayer, Robert Neumayer, and Andreas Rauber. *Rhyme and Style Features for Musical Genre Classification by Song Lyrics.* ISMIR 2008.
[2] Saif M. Mohammad and Peter D. Turney. *Crowdsourcing a Word–Emotion Association Lexicon.* Arxiv 1308.6297
[3] Doug. *How to choose the number of hidden layers and nodes in a feedforward neural network?* Stack Overflow