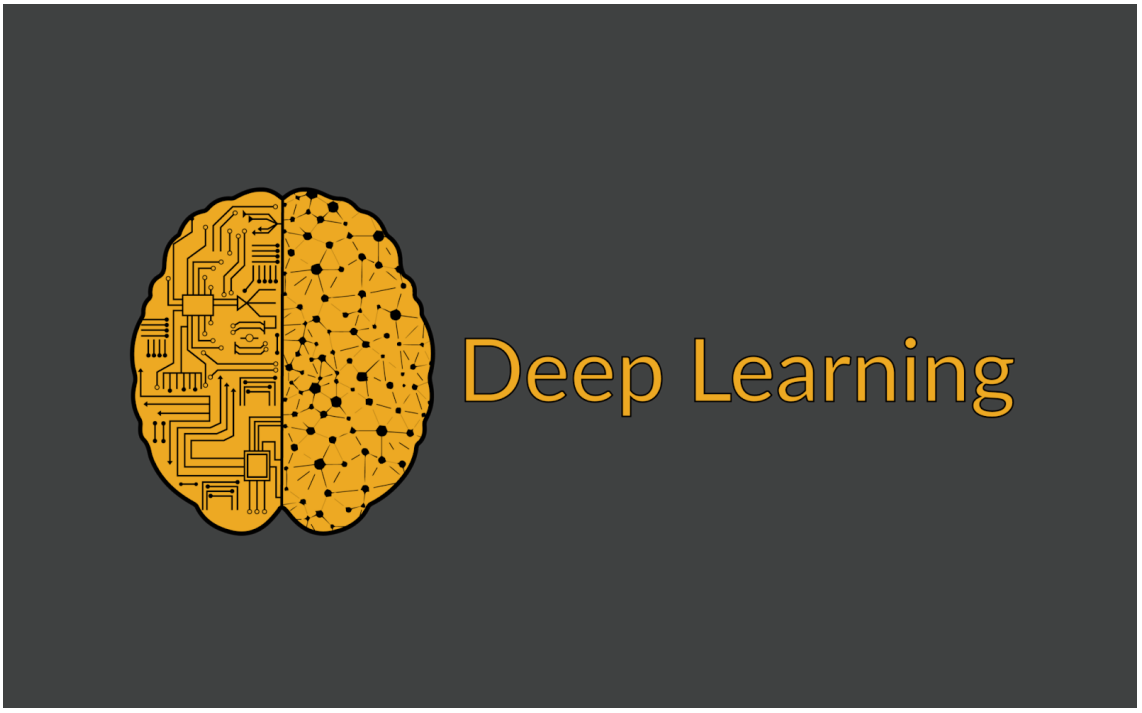
Deep learning:
Song popularity prediction

(Dal népszerűségének előrejelzése)



Authors:
Boér Lehel (Z8N953)
Gulyás Róbert (BAVBE4)

Abstract

Finding out the musical features that makes a song popular gives us huge advantage in music industry. In this research we present a deep learning approach for the hit song classification problem. We considered a song popular if its view exceeded a threshold like 10M YouTube views. We extracted more than 100 musical features from the Free Music Archive (FMA) dataset and chose the important features by recursive feature elimination. We built an MLP and tried convolutional filter too than used hyperparameter optimization on the best network to earn the highest accuracy.
The high prediction accuracy affirms that our model works well and the features that were selected are important and have great influence about a song becoming popular or not.

Introduction

In 2017, the global recorded music market grew by 8.1%. This was the third consecutive year of global growth and one of the highest rates of growth since 1997. The global recorded music industry revenues in 2017 reached 17.3 billion dollars.[11]

We decided to create a deep learning model that predicts whether a song could be popular in the future or not. With this program the industries could pre-filter a tons of unknown songs, bands, artists to decide is there any potential in them. Of course this will not predict the brand new styles like Beatles in the ‘60s. Many previous research defined a song popular if it was on a top songs billboard sometime. Unlike this, we decided to define popularity as the number of YouTube view counts.

Hit song science and previous results

Hit song science is a term coined by Mike McReady who is most known for having pioneered the science of hit song prediction. There are numerous researches available in the field and the most important is Dance Hit Song Prediction by University of Antwerp Operations Research Group from 2014. The research states that there are other researches in the field of music, but most of them deals with the classification for genre, mood, composer and instrument.[1]

Even in the few previous researches about song popularity prediction based on musical features deeply contradict.
Based on our personal research most of the articles written in this topic uses machine learning methods that aren’t belong to the deep learning field, for instance SVM, XGBoost, Random Forest, Decision Trees. According to their results these methods are more efficient than neural networks.

Data exploration and preparation

We used the Free Music Archive github repository (https://github.com/mdeff/fma) to get songs with various attributes and used YouTube to decide how popular a song is. The dataset is a dump of the Free Music Archive (FMA), an interactive library of high-quality, legal audio downloads.

All metadata and features for all tracks are distributed in fma_metadata.zip (342 MiB). The below tables can be used with pandas or any other data analysis tool. See the paper or the usage notebook for a description.

- *tracks.csv:* per track metadata such as ID, title, artist, genres, tags and play counts, for all 106,574 tracks.
- *genres.csv:* all 163 genre IDs with their name and parent (used to infer the genre hierarchy and top-level genres).
- *features.csv:* common features extracted with librosa.
- *echonest.csv:* audio features provided by Echonest (now Spotify) for a subset of 13,129 tracks.

Because of the size of the csv files in order to run the notebooks you have to download FMA_Metadata.zip from their GitHub page.

Youtube data mining

We assumed that if a song was popular at any time then it has significantly higher view count than other less popular songs. Our first try was to use the YouTube Data API that provides access to the properties of YouTube channels, videos etc. for developers. It's basically a hidden REST API communicating with YouTube in json format. The *yt_view_count.py* script is our implementation using this API to search for a video and retrieve it's view count.

Later it turned out that Google limits the usage of the API based on quota per day and we didn't have enough time for that.

We tried another approach; avoid using the YouTube API, just scrape the information from html code. The *view_count_scraper.py* is our implementation for this issue. The first problem was that the program was running in one thread and it was too slow. It would have taken two weeks to scrape all the necessary information.

in the next step we modified the code to use multithreading, but unfortunately YouTube detected that we are scraping their site and we got a temporary IP ban. Next thing we tried was to use a so called proxy rotation to avoid the IP ban. We successfully implemented this part, but the scraping took so much time that it'd last for a few month.

Then we returned to the first approach and successfully bypassed the quota limitation by creating new projects with new API key, and creating new Google accounts. We successfully got 100.000 song's view count and ran the script in 5000 sized song batches so that we do not overrun the quota. It took about one hour for each to get the view counts.

Preparing training data

To input of the model we make a subset of the full FMA by deleting the irrelevant information. In this case irrelevant information is everything about the artist because it not seems a hard task to predict a top artist's new song will have high view counts. Our goal is to create a model which capable predict the popularity from only the song's audio features like tonality, tempo, spectral decomposition, genre etc. The features.csv contains mathematical statistics of these audio features for all songs in the dataset. We treated the genres as categorical variables. The output data is the youtube view counts.

Implementation

Training

We took two approaches: We formulated the exercise as a regression and a classification problem. As a regression problem we tried to estimate the expected number of views of a song. Although we implemented the regression problem, later we changed this approach to a binary classification problem that decides whether a song is popular or not based on its number of views because it suited better to the problem. This popularity threshold is adjustable, we set it to 10 million so we assume a song popular above this line.

Firstly we build a feed forward network to see what it can do with the problem. In the classification case it trained very well, after a few epochs it reached the final accuracy.
The data consist of different groups of audio features. For example one group is consist of the energy of the twelve pitch class (these are: C, C#, D, D#, E ... B) and another is consist of the mel-frequency spectrum. So we can not use one convolution filter to sweep the whole data. We tried to use conv. filters inside the

groups, one-one independent filters to one-one groups to find out is there any shapes of them but it has not any positive effects.

We tried to reduce the size of features and filter them by importance to prevent overfitting. To solve this problem we have come up with the following solution: the Recursive Feature Elimination method that narrows the feature size by classification or regression methods (Lasso, SGD, etc.).

"Not all features however, contribute to the prediction variable. Removing features of low importance can improve accuracy, and reduce both model complexity and overfitting." [12]

The original dataset was split to 2 : 1 : 1 ratio for training, validation and test datasets, respectively. We then normalized the data with StandardScaler, created a parameterized model and used Talos for hyperparameter optimization. For speeding up the optimization process we used early stopping.

Evaluation

We constructed the model with the optimized parameter and reached about 87% accuracy on the test dataset. We trained the model with and without feature elimination but we got almost the same accuracy.

Conclusion

We earned the best results in the binary classification problem with a simple feed forward neural network. In this case the accuracy was ~87%, in our opinion it is not bad at all in this field.

References

[1] Dorien Herremans, David Martens, and Kenneth Sörensen. 2014. "Dance Hit Song Prediction." *Journal of New Music Research* 43 (3): 291–302.

[2] Lang-Chi Yu, Yi-Hsuan Yang, Yun-Ning Hungand Yi-An Chen. 2017. "Hit Song Prediction for Pop Music by Siamese CNN with Ranking Loss". *eprint arXiv:1710.10814*

[3] Dorien Herremans and Tom Bergsman. 2017. "Hit Song Prediction Based On Early Adopter Data And Audio Features."

[4] Li-Chia Yang, Szu-Yu Chou, Jen-Yu Liu, Yi-Hsuan Yang and Yi-An Chen. 2017. "Revisiting the problem of audio-based hit song prediction using convolutional neural

networks". *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*

[5] Fan, J., & Casey, M.A. 2013. Study of Chinese and UK Hit Songs Prediction.

[6] Marvin Smit."Can You Predict a hit? Finding Potential Hit Songs Through Last.FM Predictors"

[7] Benjamin Shulman, Amit Sharma and Dan Cosley. 2016. "Predictability of Popularity: Gaps between Prediction and Understanding". *ICWSM 2016*

[8] Klaus Frieler, Kelly Jakubowsky and Daniel Müllensiefen. 2015. "Is it the Song and Not the Singer? Hit Song Prediction Using Structural Features of Melodies." *Jahrbuch Musikpsychologie. 25.*

[9] Yekyung Kim, Bongwon Suh and Kyogu Lee. 2014. "Nowplaying the future billboard: Mining music listening behaviors of twitter users for hit song prediction"

[10] Ruth Danaraj and Beth Logan. 2005. "Automatic Prediction of Hit Songs." *ISMIR 2005, 6th International Conference on Music Information Retrieval, London, UK, 11-15 September 2005, Proceedings*

[11] Global Music Report 2018 - IFPI

[12] Aneesha Bakaria. Recursive Feature Elimination with Scikit Learn. 2016. *medium.com*