Analyzing YouTube Trending Videos: Factors that Contribute to Video Popularity and Cross-Cultural Understanding

CSE 6242 Group 102

Introduction

Understanding what makes a video trend on YouTube is crucial for content creators, marketers, and business owners who wish to increase their channel awareness and monetization. However, the current practice of analyzing trending videos has limitations in accurately predicting video relevance beyond a couple of days or to a single country. In this project, we propose a new approach to analyze YouTube trending videos across major English speaking countries and explore the relationships between video content, tags, and topics that drive video trends.

The primary objective is to determine what factors lead some videos to trend and to evaluate the relationship between the trend level, which we are determining based on number of views, and different factors such as likes, dislikes, category, tags, and title. The project will also analyze how cultural and geopolitical differences may affect trend factors in different countries.

Data

We obtained our data from a Kaggle dataset (YouTube Trending Video Dataset), which is a total of 3.3 GB, contains data from 11 countries, and is non-temporal due to the daily update of data.

We also obtained our data from web scraping straight from YouTube, which gave us a broader dataset that extracted additional information such as comments, video duration, channel count of subscribers, and transcript.

We also successfully utilized Google's API to extract additional metadata not available through web scraping.

Approaches

Linear Regression

Built and tested various regression models to determine what contributes to a video's trendiness on YouTube. We tested various models such as linear regression, generalized additive models, polynomial regression, k-nearest neighbors, and support vector machines, we found regression to be the best fit.

The most significant factors we found to contribute the most to a video's trendiness are likes, dislikes, published date/time, and category ID.

We then tested our model, which was trained on US data, on other English and non-English speaking countries including Great Britain, Canada, France, Japan, Korea, Mexico, and India. Our model showed a significant impact on 6 out of the 7 countries tested, with India being an outlier. We conducted further analysis and testing such as T-tests to determine whether this was due to cultural differences or data issues. The tests revealed that the outlier was more related to data and language processing issues than cultural interests.

Suggestive Title Maker

Our other ambitious goal of this project is using NLP to create a random text generator of the best related words that can be used in titles, based on keywords inputted by the content creator to optimize the chances of the video trending. We decided on the use of Long Short-Term Memory (LSTM) as LSTM are predominantly used to learn, process, and classify sequential data because these networks can learn long-term dependencies between time steps of data.

Based on our LSTM results, the overall accuracies would vary based on category due to specificity found within each category. The usage of domain specific keywords in titles would be important in order to capture the attention of target viewers.

Tagging Network Recommendation

Creators and brands alike want to know how their videos fit into existing communities and video genres and how those communities/genres vary from region to region. Such knowledge can inform how brands and creators market their videos, how to vary their marketing from region to region, and how to create content in the future. We've created a tool that allows a user to input a series of keywords and returns a list of similar videos; additionally, the user can take that video and use it to generate recommendations; this process can be repeated iteratively to specifically tailor recommendations to a user's taste. We also give the user the option to select eight different regions, facilitating comparisons between recommendations in different regions.

Experiments and Results

Linear Regression

We selected the linear regression model as it demonstrated the best prediction performance. Our model achieved an impressive R-squared value of 0.74 on the testing set and between 0.72 and 0.75 R-squared on the other datasets from the countries we tested. In addition, the linear model proved to be more efficient and effective compared to other models such as support vector machines, which took over 5 hours to run due to the large data size. This is important to note as it shows that the linear model is capable of providing accurate predictions while having a low run time, making it a practical solution for our purposes.

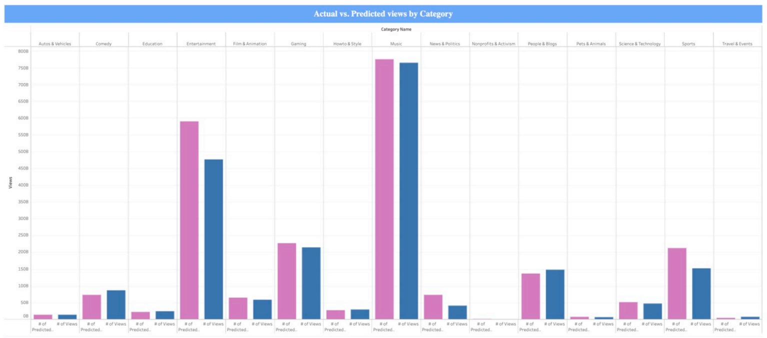


Figure 1. Actual View Count vs Predicted View Count Categorized by Category

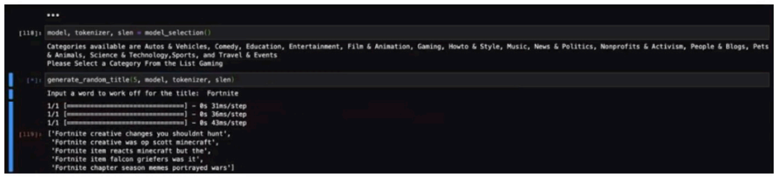
Suggestive Title Maker

To evaluate the data, we had to pre-process the data by removing redundant spaces, clean up words and numbers, correct misspelled words, clean repeat words, and clean emojis and punctuation. After the cleanup, we tokenized the titles.

The tokenizer model created a list of 37,471 unique words and the model was saved as a pickle file for faster access in subsequent code. For every title, a sequence was created which consisted of a list of pairs of words, where the first word in the pair was the start and the second word the next word. For example, the title "Dog Flipping Over Ball" would consist of [Dog, Flipping], [Flipping, Over], [Over Ball].

The next stage of the model was to prepare the features and labels, where the first word would be the feature and the last word the label. Both features and labels were converted to binary representation through the use of one-hot encoding. We utilized Callbacks in order to control the training phase and noticed with a lower batch size we could still include a decent amount of randomness within the model. With higher batch sizes the model tended to overfit which showed with an accuracy of 100%. We utilized typical parameters such as monitor = 'loss' & 'accuracy', factor=0.2, patience=3, min_lr=0.1 to not stray far from the default parameters. We started with an overfitted average score of ~95% and through multiple iterations of model fitting, achieved an average accuracy score of ~65% and average loss of ~1.5 using 50 as our epoch value.

We used the LSTM model we created and applied a category filter, and then ran the category model selected on the data for each video category which would surface a list of the top 5 suggestions for the next possible sequence of words based on the current word. The final result was a user generated prompt that a user can input a word and view the suggestions immediately.



```
...
[118]: model, tokenizer, slen = model_selection()

       Categories available are Autos & Vehicles, Comedy, Education, Entertainment, Film & Animation, Gaming, Howto & Style, Music, News & Politics, Nonprofits & Activism, People & Blogs, Pets
       & Animals, Science & Technology,Sports, and Travel & Events
       Please Select a Category From the List Gaming

[*]:   generate_random_title(5, model, tokenizer, slen)

       Input a word to work off for the title:  Fortnite

       1/1 [==============================] - 0s 31ms/step
       1/1 [==============================] - 0s 36ms/step
       1/1 [==============================] - 0s 43ms/step
[119]: ['Fortnite creative changes you shouldnt hunt',
        'Fortnite creative was op scott minecraft',
        'Fortnite item reacts minecraft but the',
        'Fortnite item falcon griefers was it',
        'Fortnite chapter season memes portrayed wars']
```

Figure 2. Demonstration of Suggestive Title Maker

Tagging Network Recommendation

Using graphs built off of extracted video id, country, and tags, we created two functions: like_videos and recommendations. The first function takes a set of keywords, a user's location, and a number of recommended videos as arguments and then executes a PageRank algorithm using tags vertices to generate matching key words. After forming a order of videos based on the sum of the PageRanks of the tags, we use the second function and input of video_id and user_id to store a user's preferences.

A specific user will continue using the recommendation function to build up a set of preferences and eventually lead to a finely tailored list of video recommendations, as shown in the demo. This method is unique because users can interact these functions through a R Shiny application and have their results returned in the form of a table and interactive graph network so they can visualize how their videos are related. To improve the performance of the functions, we included exponential smoothing algorithm to calculate personalization weights overtime.

The advantage of this model lies in its simplicity of using publicly available data and basic graph anaylsis. However, it does perform less efficiently when the number of tags per video are lacking or commonly used. To improve upon the model would require us to look beyond tagging.

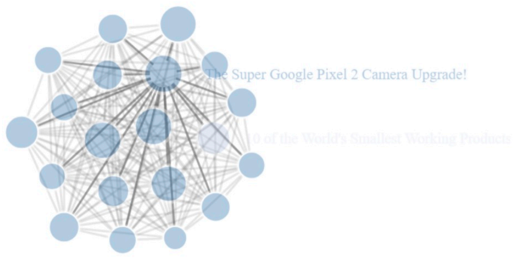

The Super Google Pixel 2 Camera Upgrade!
of the World's Smallest Working Products

Figure 3. Demonstration of Tagging Network Recommendation Graph