Sentiment Analysis of YouTube Comments for Estimating Like-Dislike Ratio

A Report by

Amanbeer Khanduja	013714181
Aanchal Agarwal	015923778
Palash Shankar Bhusari	015721498
Prasaanth Radhakrishnan	015279524
Shubhangi Manoj Kumar	015932826

Abstract—Youtube's removal of showing dislikes on a video resulted in a big backlash and was widely criticized for supporting big organizations which did not want their videos getting disliked as it would result in the videos not being watched as much. Though the company believes this would help content creators avoid hate on the platform through "dislike bombing", many also criticized this move saying it removed an important metric for users to judge a video's legitimacy. A majority of Youtube users also criticized this move and had a petition on change.org to bring back the button which has presently been signed by about 19,000 people. We intend to make a browser extension that goes through the comment section of the video and uses Machine Learning algorithms with Sentiment analysis to get the overall response of the commenters and estimate the like-dislike ratio the video could have gotten.

Keywords—Machine Learning, Natural Language Processing, Sentiment Analysis, YouTube Dislikes

INTRODUCTION

Sentiment Analysis or Opinion Mind is the technique of using Natural Language Processing (NLP) to determine if a sentence is a positive, negative or neutral sentence. This is used by companies to determine the overall customer satisfaction or reviews so they can get an overall view of the customer's experience. This is a vast field and is widely being used to determine various outputs or make changes to have better products. It's mostly used for customer feedback and their needs.

There are four types of Sentiment Analysis Techniques, they are Graded Sentiment Analysis, Emotion Detection, Aspect-based Sentiment Analysis, and Multilingual Sentiment Analysis.

A. Graded Sentiment Analysis

It mainly focuses on the polarity of the sentence, based on analyzing the sentence it can fall under five categories or as determined by the requirements. The 5 popular classifications are as follows:

- Very Positive
- Positive
- Neutral
- Negative
- Very Negative

B. Emotion detection

It is used to get an even better analysis of the sentence as it can be used to determine emotions such as happiness, anger, frustration, or sadness. Major systems use lexicons to determine emotions. Lexicons have a list of words and the emotions associated with them.

C. Aspect-based Sentiment Analysis

This is used to determine which aspect of the sentence they are expressing an emotion about. For example, if a user comments “the length of the video was too long” or if the “audio quality is bad”, we can determine which aspect of the video was liked or disliked(which would be length and audio in this case).

D. Multilingual Sentiment Analysis

This would include classifying based on many different languages. A good approach to this would be to first use a language classifier and use a custom model for each language to get better results.

We plan to use Sentiment analysis as the main tool to determine how many dislikes a Youtube video could have based on the comments by all users in the video. It mainly leans towards using a variation of Graded Sentiment Analysis to assign a number to the negative and very negative comments for the video.

BACKGROUND

A. Problem being investigated

In November 2021, YouTube, which is one of the biggest video viewing websites in the world removed all of the dislike counts from the videos which prevented users from seeing them. While the dislike button was still present on the website for users to press, only the video creator could see the exact number of dislikes a video received. Users, however, can only see the exact number of likes.

This change was very controversial and received a major backlash from frequent YouTube users. While YouTube stated that the reason they did it was that it actively affected their ecosystem where large groups of people would intentionally target a specific creator and make sure their videos got a lot more dislikes than they deserved, and this constituted harassment of creators, the general user base did not agree with this being a solution for that problem.

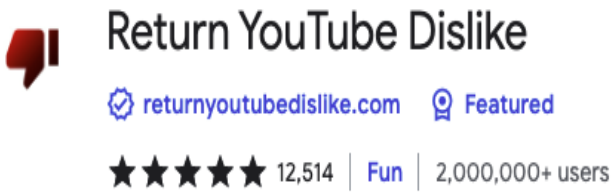
The dislike count was a major part of the viewer experience as that determines whether the user would even watch that video or watch it for a longer period of time. It also determined the legitimacy of the video. Users generally want to see what other people think of the video without having to scroll through the comments to figure out why a video is facetious. The dislike button, and subsequently the like-dislike ratio, was important in giving users a general idea of what viewer perception on a video was like without having to do a lot of work.

B. Purpose of the Project

This project's purpose is to be able to make a browser extension that would go through the comment section of the video and be able to gauge the number of users that could have disliked the video and be able to display a "sentiment" of the video through the extension.

II. CURRENT APPROACHES

A. Existing extension



There aren't a lot of solutions to this problem but there does exist one Extension which seeks to answer the same problem, which is to show the dislike count on the YouTube video. This extension is called 'Return YouTube Dislike'. This extension was created by Dmitry Selivanov. The approach this takes is that:

- This extension doesn't use any machine learning methods as such, instead relying on the power of a large number of users using this extension to make the number of dislikes.
- It has a database of videos that were posted before December 13 2021 and their dislike counts. A prime example of this is the video 'YouTube Rewind 2018' which was known to be the most disliked video ever and had a dislike count of about 20M.
- For newer videos, it stores the dislikes that are placed on the video by the users of the extension.
- This extension has a database of about 200 Million plus videos before that date hence will be able to provide the exact number of dislikes for those videos.

B. Issues not being solved

This is a pretty decent extension but it also has a few drawbacks. They are:

- This would require going through a database of about 200M videos to find the actual video and then retrieve the actual dislike count to display which makes it take a lot of time to actually load the video and then display the song. This deters the user experience of wanting to watch a video.
- This method is good for videos that were posted prior to December 13 and are not very efficient for videos posted after that date.
- It requires more users to determine the dislikes ratio of newer videos but counting their dislikes and displaying them hence it would not be an accurate value for the dislikes.

- There are just two extensions(for Google Chrome and Firefox respectively) to display YouTube dislikes and this shows that this project is not a well-explored field yet.
- The Google API that the extension uses to predict the dislikes for new videos is going to be shut down on December 13th which would result in problems with newer predictions after that.
- It depends on the users of the extension for newer videos which is a very small user base of the people who actually watch the videos.
- At present, there are 2,000,000 users of the extension while over 2.24 billion use YouTube. This indicates the efficiency of this extension in determining the likes is based on the opinion of 0.083% of the actual user base.

C. Improvements that could be done for this method

There are various improvements that could be done for that particular extension. Some of them are

- Store cookies of previously watched videos by the user so it does not have to retrieve the dislike count every time the video loads.
- Needs to find a way to get more users to use the extension as the accuracy of the extension solely depends on the number of people actively using it.
- Work on a different algorithm that does not make the extension depend on the user of the extension.

RELATED APPROACH

A. Research Paper referred

We have referred to a research paper titled 'NLP Based Sentiment Analysis on Twitter Data using Ensemble Classifiers', authored by Monisha Kanakaraj and Ram Mohana Reddy Guddeti. Their paper's main focus is to use a Natural Language (NLP) based approach to enhance the sentiment classification by adding semantics in feature vectors and thereby using ensemble methods for classification.

B. Introduction

This paper proposed adding semantics while extracting feature vectors. As various machine learning techniques like SVM, Naive Bayes, and Maximum Entropy are used to solve classification problems usually.

It involves the analysis of the mood of society on a piece of particular news or topic from Twitter posts. Their key idea is to increase the accuracy of classification by using Natural Language Processing(NLP) techniques, especially with Synsets and Word Sense Disambiguation with ensemble methods.

C. Key Contributions

The key contributions in the proposed work are:

- One of the first papers to propose the use of Sysnets and Word Senses as features in the sentiment analysis systems.

- One of the first papers to work on the combined framework of the semantics-based ensemble classification system.

D. Comparison with existing techniques

There were two main approaches that they compared in mining sentiment from the SNS:

TABLE I

COMPARISON OF EXISTING METHODOLOGIES

	Dictionary Based	***Machine Learning-Based***
Approach	Uses reference dictionary for classifying individual sentences/words	Uses Supervised or Unsupervised ML algorithms for probability-based classification
Advantage	Lesser computation overhead as no training and modeling required	Can be customized to work more suitably for specific domains
Limitation	Lacks context or domain-based classification capability	Affected by class imbalance and linguistic variations problems.

In general, all the DB techniques refer to a pre-built dictionary for classifying the sentiment. But then it has a limitation in that the strength of classification depends on the reference dictionary used. On the other hand MLB systems achieve classification based on training data and algorithm used.

TABLE II

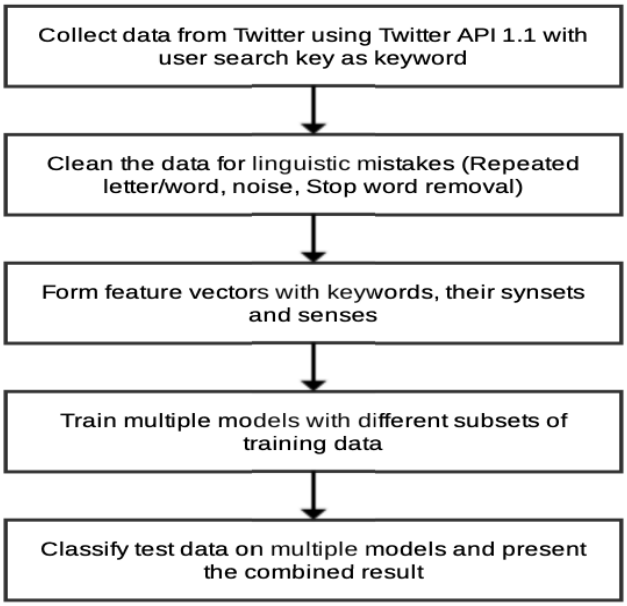
COMPARISON OF EXISTING WORKS

Work	***Advantages***	***Limitations***
[1]	Considers Lexicons & Emoticons	Lacks Domain Context
[2]	Compares Online vs Social reviews	Lacks aspect rich data
[3]	NN combined SVM	Only on stock market data

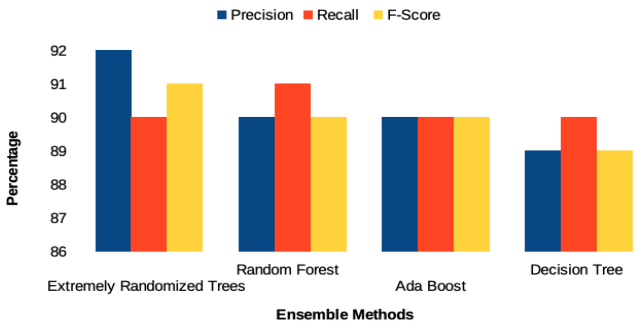
[4]	Naive Bayes classifier	Baseline methodology

From the above table we can see the advantages and disadvantages of various methods that were compared. [1] is a method that is a content-based ranking method (AFINN) for a Facebook page done on user engagement. [2] is a method for data collection, and sentiment analysis for online reviews and tweets using KNIME. [3] is a method for combining NN and SVM for analyzing tweets on a particular stock market data. [4] is a method of using Naive Bayes Classifier to identify the sentiment of a particular status on Facebook.

Proposed System by Twitter Classifier



E. Results



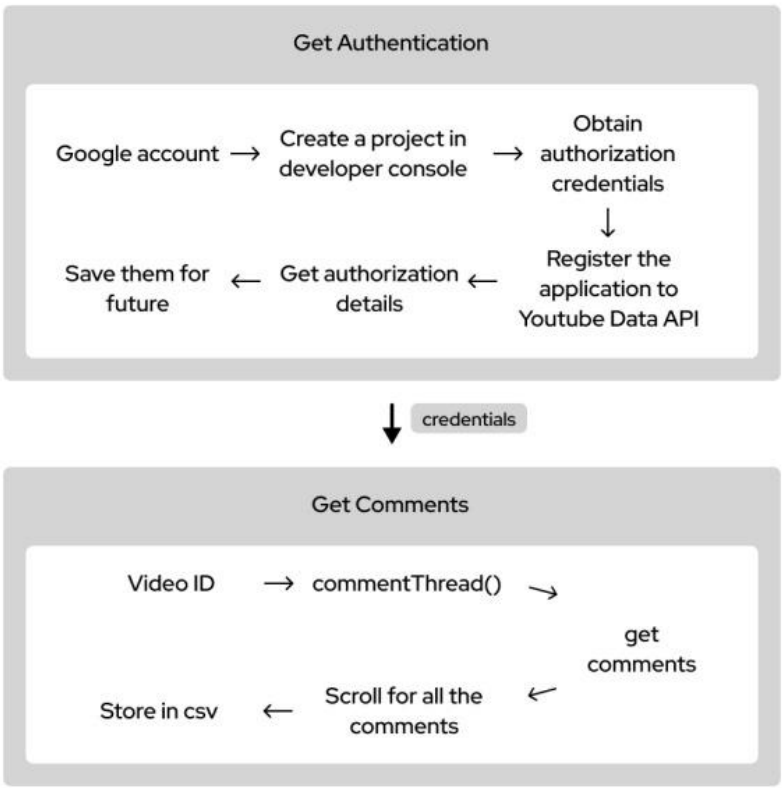
This result is pivotal because it not only proves that sentiment analysis is prudent in social network analysis, like we plan to do, but is also an efficient process that has been tried and

tested using various algorithms before. Based on this confidence, we are going to make our own implementation.

SELECTING DATASETS

Our process consisted of selected datasets with comments for the analysis of the comments. Firstly, we went through the datasets in a lot of sources such as Kaggle, Data World, Open ML Repository, and US Census. However, once we completed the data preprocessing and started working on the data, we realized that most datasets were all popular video datasets with very one-sided types of comments such as song videos, movie comments, etc. It is pertinent to note that most of the comments in the music video were the most common datasets of youtube comments.
Therefore, for the purpose of our project, we took videos off the trending section of YouTube and scraped the comments from those videos. We found an existing dataset of YouTube comments on Kaggle and added the scraped comments to the existing dataset to augment it with more recent data. We tried not to rely on music videos or movie-based videos, since a lot of comments on these videos contain neutral information such as movie facts or song lyrics, and also become relevant, skewing user analysis.

In the scraping process, we used the parameter of relevance that the YouTube API provides to help skim through spam comments. While this method isn't perfect, since some spam comments still make it through to the dataset, it has been found that their sentiments are often neutral, so they don't negatively impact the analysis very hard unless they are present in a majority. Thus, spam comments were not a major problem for our approach.



OVERALL PROCESS

Our overall process mainly consists of two parts which are i) Getting the authentication and ii) Getting the comments on the YouTube video. Its explanation is given below

Getting the Authentication

Google Account: This part of the process requires a Google Account to access the APIs required for getting the comments on the video.

- ***Create a Project in the developer console:*** This involves accessing the API developer console and creating a project with a unique ID and setting up a billings account if we need to charge project costs.
- ***Obtain authorization credentials:*** YouTube supports two types of API which are OAuth 2.0 and API Keys. We can generate OAuth 2.0 for web applications, service accounts, or installed applications. A request that does not generate an OAuth 2.0 token must generate an API key.
- ***Register the application to YouTube Data API:*** The YouTube API is an application programming interface that allows you to embed videos, curated playlists, and offer other YouTube functionalities on your website. It helps businesses offer advanced video-sharing features on their website or app without needing to write code from scratch. It has a wide selection of features and functionalities that allow us to retrieve data at a mass scale without accessing individual videos and channels.
- ***Get authorization details:*** This step is forgetting the authorization to use the YouTube API and the API key to get all the YouTube comments.
- ***Save them for the future:*** We can find the API key stored in the Google Developer Account and this can be reused in the Project Dashboard under the APIs card.

Getting the Comments

- *Getting the Video ID:* Every YouTube video has a specific ID that can be used for retrieving the comments on it. This is used for the API.
- *Comment Thread():* A commentThread resource contains information about a YouTube comment thread which comprises a top-level comment and replies if any exist to that comment. A commentThread resource can represent comments about either a video or a channel.
- This supports two methods for getting the resources which are
    i) List: Returns a list of comment threads that match the API request parameters.
    ii) Insert: It can also create a new top-level comment. Most people misuse this to spam a YouTube comment section to promote their own channels or products which is now a major problem that YouTube is facing.

```
{
  "kind": "youtube#commentThread",
  "etag": etag,
  "id": string,
  "snippet": {
    "channelId": string,
    "videoId": string,
    "topLevelComment": comments Resource,
    "canReply": boolean,
    "totalReplyCount": unsigned integer,
    "isPublic": boolean
  },
  "replies": {
    "comments": [
      comments Resource
    ]
  }
}
```

Listing Down Comments: Making use of the List of comments that we received from the commentThreads() method from YouTube API, we can now use this data to perform further operations.

Scroll for all the comments: Once we have the list of comments, we go through the list to check for possible comments that could provide us essential information about how a user finds that video i.e., good or bad. This data should help us in determining the likes and dislikes of the video. This data should also not include spam comments such as channel promotions or unrelated comments. This is done by using the order = relevance parameter, which uses YouTube's default

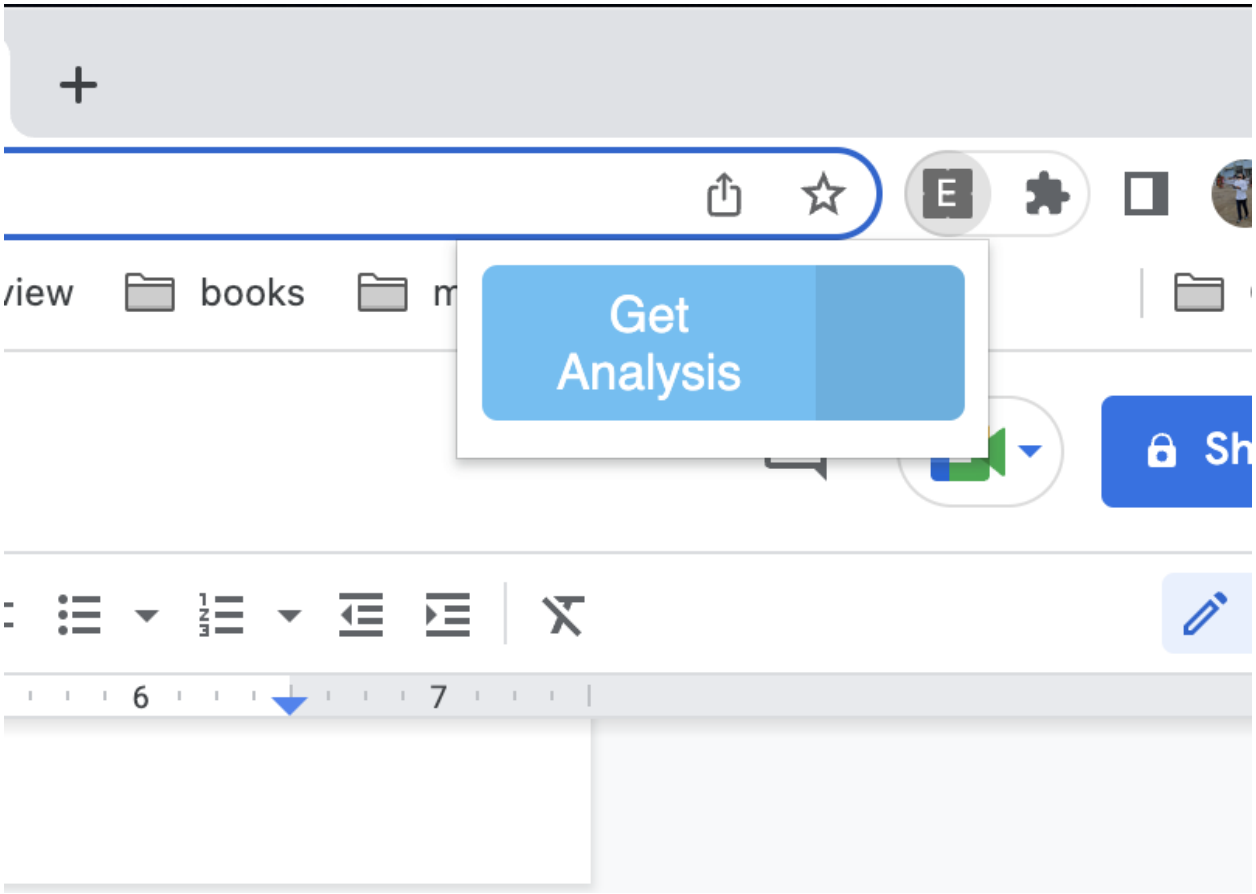
Store in CSV: This list of data cannot be directly used for performing Machine Learning operations so we need to store it in a CSV file so we can perform more operations on them and use it as a data frame.

For the ML part, we are using an implementation of BERT (Bidirectional Encoder Representations from Transformers), which entails a pretrained representation of a large word corpus for use in sentiment analysis. We trained our model with the dataset that was described above. The model was then configured with the TextBlob wrapper library, which opened functions for sentiment analysis on a large scale with acceptable time complexity. We needed this since we would be applying sentiment analysis to a large number of comments in real-time, therefore time complexity is of utmost importance.

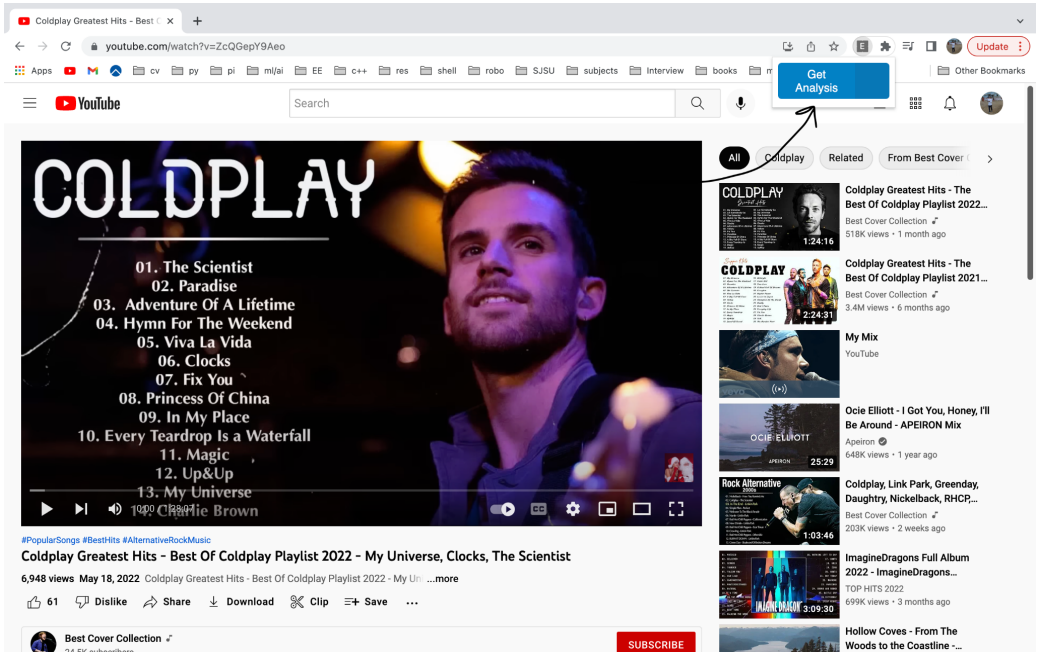
RESULTS

An important note that should be made about the "accuracy" of a custom model trained by BERT is that since the model is unsupervised and unlabelled, the best metric of performance is its performance with YouTube videos. As per our heuristic evaluation, we found that all videos with a large number of dislikes (from before those videos had their dislikes removed) were categorized as "Bad", while most positive and legitimate videos were categorized as "Good".

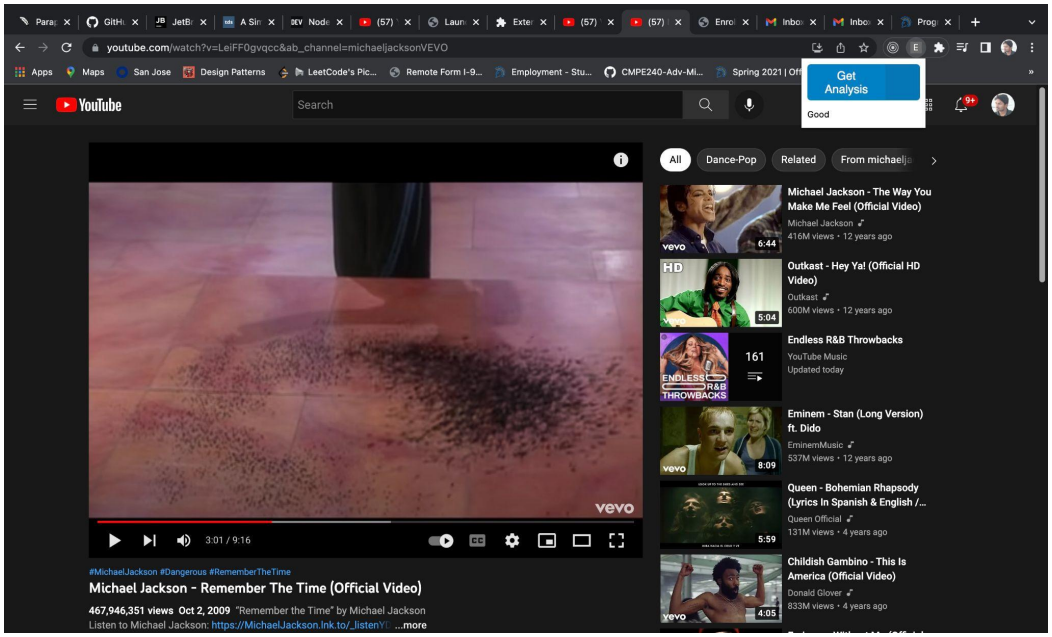
The chrome extension looks like the figure below. As is visible, the extension is faded out when the active tab is not a YouTube video. While there is no logic currently to stop a user from pushing a non-YouTube link to the extension, the Google API rejects any other links and returns a blank answer. Thus the extension will only work for YouTube videos.



When a YouTube video is in the active tab, the extension lights up to blue. The button can now be pressed, which will send the currently active URL to the python script. This python script will strip the URL to its video ID, scrub through the 100 most relevant comments, and perform sentiment analysis on all the comments. After that, it will average out the sentiment score of all non-zero values found from the comments and return a value. We found that the threshold for polarity was 0.08. Anything below 0.08 as a score was most definitely a negatively perceived video, and anything above was better/well-received. While this value is lenient or skewed, as one could say (considering the function can give values between -1 and +1), it was seen that averaging out has a correction effect on all comments, and that is taken into account when deciding the threshold value.



Upon hitting the “Get Analysis” button, the extension returns a binary value. A video can either be “Good” or “Bad”, depending on the analysis of the top 100 comments (based on YouTube’s relevance). As can be seen from the image below, the video that’s played in the browser has a “good” rating. However, another video with conflicting or bad comments might get a rating of Bad by the extension.



LIMITATIONS AND FUTURE SCOPE

Currently, there are a few things that are missing from this implementation. They are as follows.

1. The extension can only work for comments in the English language presently. That leaves behind a large number of comments from different languages like Hindi, Spanish, Portuguese, Russian, etc.
2. The extension is currently not granular enough, only producing a binary output (Good, Bad). A more nuanced sentiment score would be able to produce better outputs, such as showing controversies or conflicting/neutral opinions on a video.
3. The extension doesn't work well with music videos currently, since music videos often have lyrics of the video written in their comments, which don't get blocked out in terms of relevance by YouTube's algorithm. This makes it so that the song's lyrics are put into sentiment analysis, and that is not an accurate metric of the *opinion* on the video.
4. In the future, a multivariate approach to YouTube comments can produce more nuanced feedback about a video's opinion. Variables such as likes on comments and replies on comment threads can be used to better inform the system about the opinion on a video.
5. Comments on a YouTube video are a function of time i.e sometimes, more recent comments have a different opinion of a video than older ones, skewing the general analysis to some extent.

CONCLUSION

While these issues are open and currently active in our implementation, the current state of the implementation proves successful in giving a binary score of the legitimacy of a YouTube video, which is what the objective of our project was.

REFERENCES

[1] M. Kanakaraj and R. M. R. Guddeti, "NLP based sentiment analysis on Twitter data using ensemble classifiers," *2015 3rd International Conference on Signal Processing, Communication and Networking (ICSCN)*, 2015, pp. 1-5, doi: 10.1109/ICSCN.2015.7219856.

[2]A. Porshnev, I. Redkin, and A. Shevchenko, "Machine learning in prediction of stock market indicators based on historical data and data from twitter sentiment analysis," in Data Mining Workshops (ICDMW), *2013 IEEE 13th International Conference on. IEEE*, 2013, pp. 440–444.

[3]P. T. Ngoc and M. Yoo, "The lexicon-based sentiment analysis for fan page ranking in facebook," in Information Networking (ICOIN), *2014 International Conference on. IEEE*, 2014, pp. 444–448.

[4] Xu, H., Liu, B., Shu, L., & Yu, P. S. (2019). BERT post-training for review reading comprehension and aspect-based sentiment analysis. *arXiv preprint arXiv:1904.02232*.