Sentiment Analysis Tool for YouTubers

Project Report

Created by:

Dinesh Nariani

Table of Contents

INTRODUCTION 3

Problem Definition 3

Project Overview 3

LITERATURE REVIEW 3

Existing Systems 3

Proposed System 3

Feasibility Study 4

SYSTEM ANALYSIS & DESIGN 4

Algorithms 4

BERT 4

Pseudo Code 7

RESULTS 7

CONCLUSION 10

REFERENCES 11

INTRODUCTION

Problem Definition

With increased emotional perception and online engagement, it is vital to realize and monitor what type of emotions we are stirring in others. It is the key reason behind the recent uprise of sentiment analysis. The analysis of emotions can be used to create value for the end users of various products and services. We delve into a specific service, YouTube which is the world's leading online video platform, owned by Google.

Project Overview

As of 2021, there are around 2 billion YouTube users in the world. It is difficult for content creators to stand out from each other. Our project aims to help youtubers understand the impact of their videos on viewers and further enable them with a choice to take their channel in their desired direction. We intend to achieve that by creating a tool that will conduct a sentiment analysis of YouTube videos of a channel based on a particular timeline.
For the core sentiment analysis, we will be using deep learning by deploying the Bert model. We will use the YouTube API to fetch subtitles of videos uploaded between the specified time period and create a clean corpus out of it by using the fundamental concepts of natural language processing. Finally, we will output the sentiments visualized by a bar graph, classifying it based on 7 key emotions: joy, sadness, fear, anger, neutral, love and surprise.

LITERATURE REVIEW

Existing Systems

While much interesting work has been done on sentiment analysis of the comments on YouTube videos, no prior work is done on the analysis of the language used in the videos themselves. Our work is, if not the only, then one of the first attempts in that direction.

Proposed System

Instead of analyzing the reactions to the videos, we propose a system to analyze the sentiments of the content imparted in the videos in the form of speech. We specifically aim to create a valuable resource for YouTubers by molding our system in form of a user-based tool. In our tool, the YouTubers can put in their channel information and get the sentiment analysis over a time range, also specified by them. Additionally, they also have the option of using the tool to do the analysis

over a single video. This is designed to serve as a reflection/feedback tool for the YouTubers to check what type of sentiments they impart in their content.

Feasibility Study

Sentiment analysis of given text is a fairly common practice. The gap we needed to fill for our system to work was fetching the speech of the videos as text, which we accomplish by fetching the closed caption transcripts of the videos and working on them.

SYSTEM ANALYSIS & DESIGN

Algorithms

In our algorithm, we first load the data from the extracted array using text_from_array. The first argument is the array where we extracted the emotion and intent dataset. For sentiment analysis the encoding is done for the 7 emotions. We basically create a vocabulary where each emotion is given a specific key because the neural network model can only accept integer values. Similarly for Intent based classification we have encoding around 150 words so that our BERT model can understand the text and give intent based analysis for our YouTube Video.

The maxlen argument specifies the maximum number of words in the emotion or intent dataset where the longer texts are cut short(truncated) to 256. BERT can handle a maximum length of 512 but to improve our speed and reduce memory load we want to use maxlen of 256. The text preprocessing should be done using preprocess_mode to ‘bert’.
Then we are loading BERT and wrapping it in a Learner object. The first object to get_learner uses the model to load the pre-trained model. While the second and third argument are training and validation data. The last argument to get_learner is batch size which we have set it to 6 based on what was recommended by Google for 12GB GPUs. And finally, to train the model we use fit_onecycle method of ktrain. It gives a single cycle learning rate policy that linearly increases the learning rate for the first half and decreases the second half.

BERT

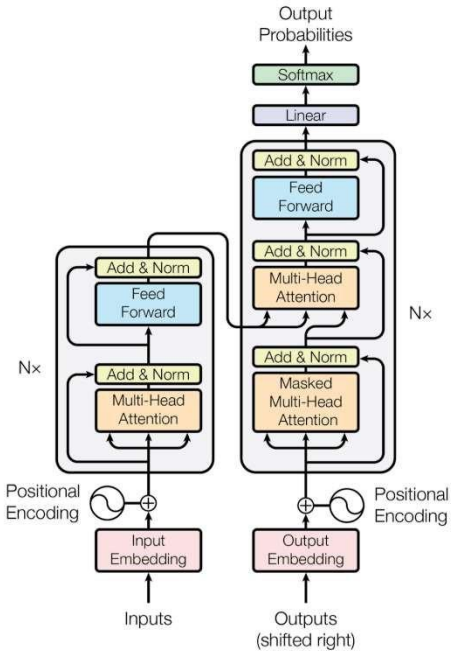
What is BERT?

BERT, short for Bidirectional Encoder Representations from Transformers, is pre-trains deep bidirectional representations from unlabeled text by joint conditioning on both the left and the right contexts. Each word here has a meaning to it. The important takeaway from this line is – BERT is based on the Transformer architecture. Secondly, any large corpus of unlabeled text is pre-trained

by the BERT model including the entire Wikipedia(that's 2,500 million words!) and Book Corpus (800 million words). Third, BERT is a "deeply bidirectional" model. Bidirectional means that BERT learns information from both the left and the right sides of a token's context during the training phase.

What is a Transformer?

The transformer neural network was introduced in 2017. It employs encoder-decoder architecture similar to RNN. The input data can be passed in parallel. It determines word embeddings simultaneously instead of sequentially. The entire concept of Transformer is based on attention, more importantly, self-attention. Self-attention, sometimes referred to as intra-attention, is an attention mechanism relating different positions of a single sequence in order to compute a representation of the sequence. There are two main blocks in a transformer: encoder and decoder.



As can be seen from the figure above, the encoder consists of 1 layer of multi-head attention followed by another layer of Feed Forward Neural Network. The decoder, on the other hand, consists of an additional Masked Multi-head Attention. These encoder and decoder blocks are actually multiple identical encoders and decoders stacked on top of each other.

The transformer widely changed the way natural language processing is carried out, while language translation machines process words sequentially, a transformer can do it parallelly making it a lot faster and efficient.

However, there are still some limitations to a transformer:

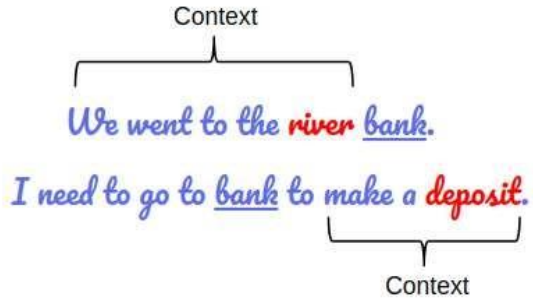
Attention can only deal with fixed-length text strings. The text has to be split into a certain number of segments before being fed to the system as input
This chunking of text causes context fragmentation. For say, if a sentence is split from the middle, then a significant amount of meaning is lost. Moreover, without respecting the sentence or any other semantic boundary the text is split.

How does BERT work?

BERT makes use of a Transformer, a mechanism that learns textual relations between words (or sub-words) in a text. In its basic form, Transformer includes two separate mechanisms — an encoder and a decoder. The encoder that reads the text input and the decoder that produces a prediction for the task.

Let us take an example to understand this:

We will make a mistake in at least one of two examples if we try to predict the type of word "bank" by simply taking the left or the right meaning. One way to address this is before a forecast, to remember both the left and the right sense.



BERT's text pre-processing

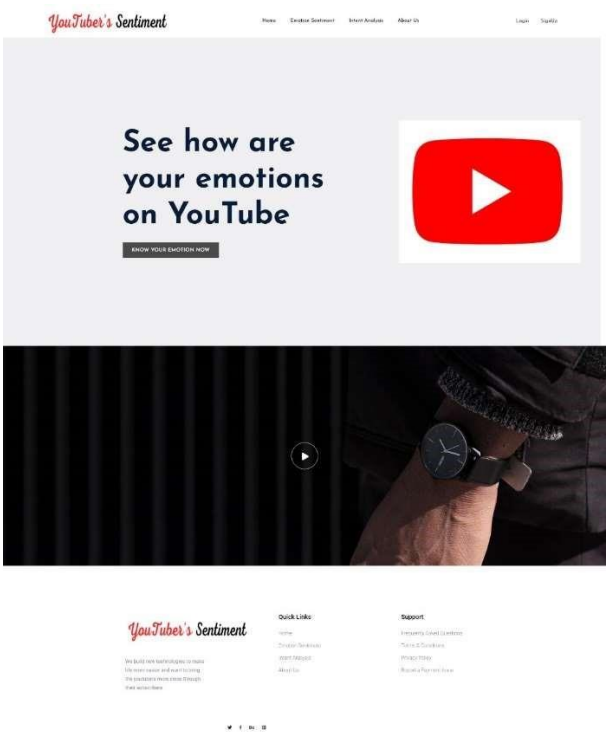
BERT learns and uses positional embeddings to know the position of words in a sentence. These are added to overcome the limitation of Transformer which, unlike an RNN, it can also take sentence pairs as inputs for tasks (like QnA). To help the model distinguish between them it learns a unique embedding for the first and the second sentence. In the above example(see photo), all the tokens marked as EA belong to sentence A and same for sentence B (EB). Token Embeddings are the embeddings learned for the specific token from the Word Piece token vocabulary.

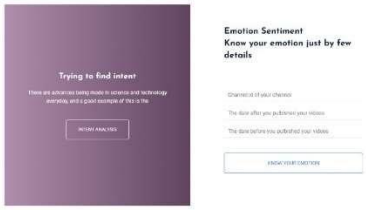
Pseudo Code

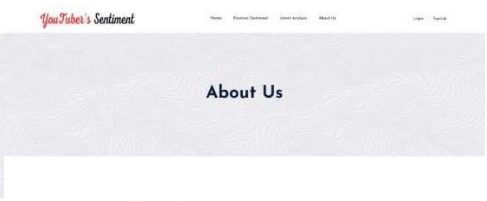
1. Install all dependencies (modules)
2. Set up training and validation data
   2.1. Read data from .csv files
   2.2. Convert it to lists
   2.3. Append training and validation data to check number of entries in each class
   2.4. Define classes
   2.5. Check the dataset
   2.6. Define training and validation data couples and preprocessor
   2.7. Label encoding is applied for the classes
3. Set up the model
   3.1. Define a pretrained classifier model of type ‘bert’
   3.2. Reload previously saved weights to resume training from last time (optional)
   3.3. Define a ‘learner’ object
   3.4. Find an appropriate learning rate
4. Training and validation
   4.1. Train the model with training data (transfer learning: fine tuning)
   4.2. Validate the trained model with validation data
   4.3. Prepare the model for testing by defining a ‘predictor’ object
   4.4. Save the predictor object
5. Fetch testing data
   5.1. Get channel ID from user
   5.2. Get start and end date from user
   5.3. Fetch video IDs from channel IDs
   5.4. Get subtitles/transcripts for each video using video IDs
6. Prepare testing data
   6.1. Remove special characters, repetitions.
   6.2. Expand negative abbreviations.
   6.3. Remove all stop words except not, nor and no.
7. Directly use the saved predictor on the curated testing data and plot results in a bar chart.

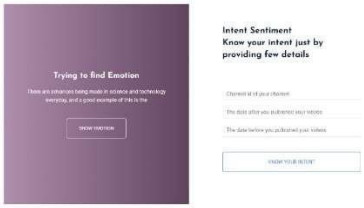
RESULTS

We have created a website for the same, where the entire process happens end-to-end for users, using our pre-trained model. The website looks like:





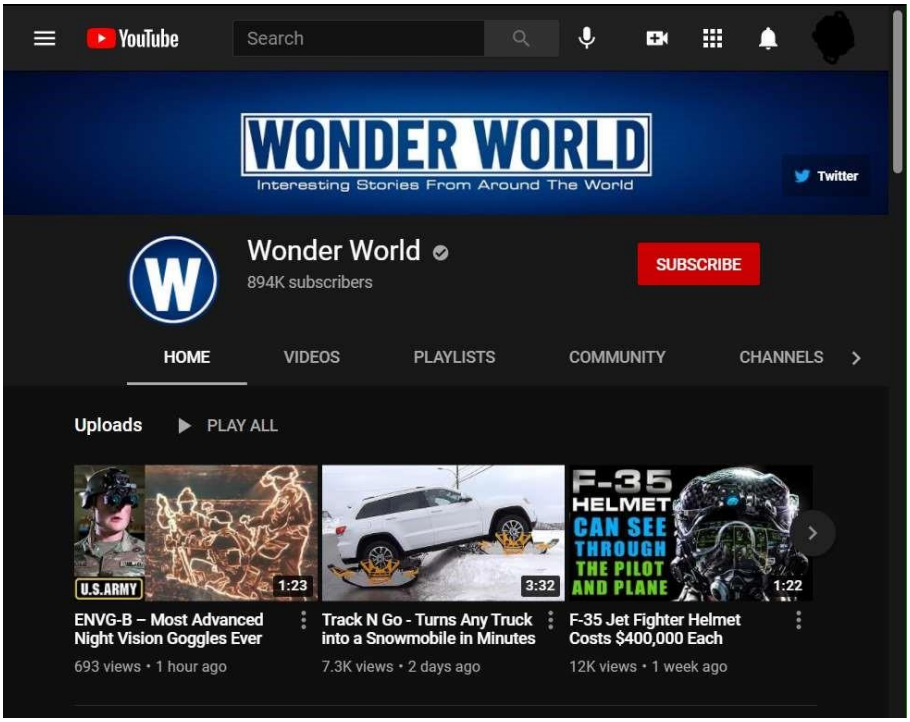




It has a login and email verification system:

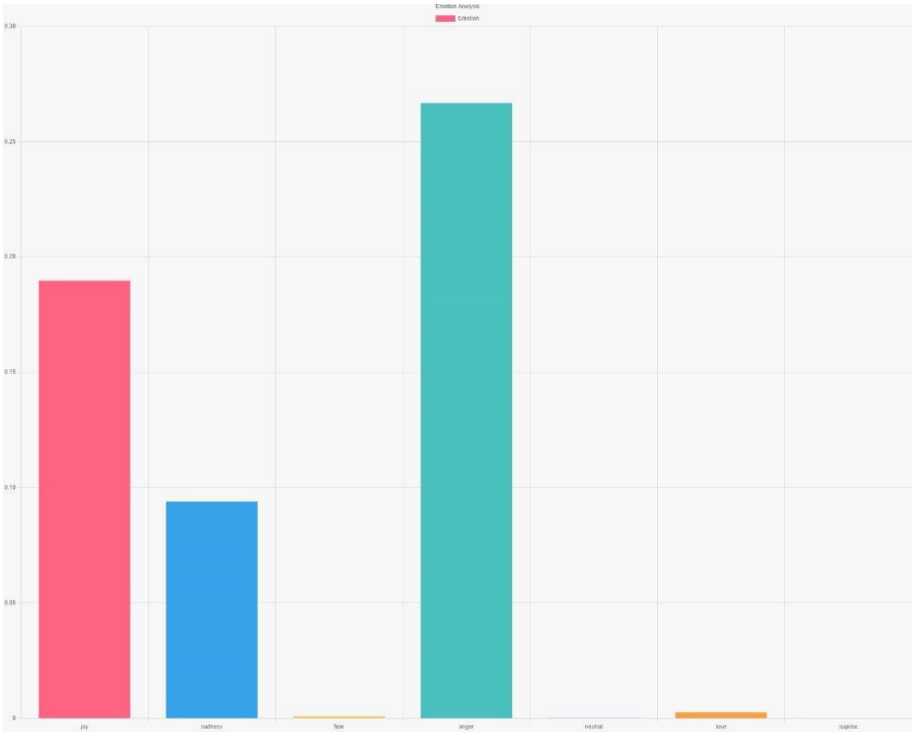
Here is an example of the sentiment analysis performed on our website:

The YouTube channel (ID: UC0k173Oca1nPZurW2ITHlYw):



Dates: 01/04/2021 to 05/05/2021

Result:



Please also refer to the video shared with this report.

CONCLUSION

Using the BERT model for deep learning and some of the fundamental concepts of Natural Language Processing, we were able to develop a fully functioning sentiment analysis tool for YouTubers. We hope to help content creators gain a deeper insight into the general impact their videos have on viewers. Our project helps in visualizing the results with the help of a clean bar graph classifying the video in 7 different sentiments based on the subtitles fetched by YouTube API.

There is a scope for future works in our project. Though it is beyond the requirements or the goal of our project, out of curiosity, and to try to add additional functionality to our website, we tried to implement intent analysis but due to lack of proper data, results did not come out to our satisfaction. With further work on it, we will be able to perform proper intent analysis as well. Moreover, we can deploy our website in the future. Further, we can extend our services to creators on other video based content websites if proper API's are available for use.

REFERENCES

[1] (n.d.). BERT Explained: State of the art language model for NLP | by Rani .... Retrieved May 5, 2021, from https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-fornlp-f8b21a9b6270

[2] (n.d.). BERT Explained: State of the art language model for NLP | by Rani .... Retrieved May 5, 2021, from https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-fornlp-f8b21a9b6270

[3] (n.d.). YouTube Data API | Google Developers. Retrieved May 5, 2021, from https://developers.google.com/youtube/v3

[4] (2021, March 26). ktrain · PyPI. Retrieved May 5, 2021, from https://pypi.org/project/ktrain/

[5] (2019, September 25). Demystifying BERT: A Comprehensive Guide to ... - Analytics Vidhya. Retrieved May 5, 2021, from https://www.analyticsvidhya.com/blog/2019/09/demystifying-bertgroundbreaking-nlp-framework/

[6] (2019, June 19). Understanding Transformers in NLP: State-of-the-Art Models. Retrieved May 5, 2021, from https://www.analyticsvidhya.com/blog/2019/06/understanding-transformers-nlpstate-of-the-art-models/