YouTube video Recommender System

Ayush Raj (ar4283), Sumit Chavan (smc2306), Sunjana Chintala (sc4921),
Yuren Dong (yd2620),Yajie Zhang (yz3876)

Introduction

This project is focused on exploring and building recommender systems using different approaches. We use the YouTube dataset from kaggle to work on this project. Our recommender system takes in the title of the video watched/passed as the input and recommends the top 10 most similar videos from the dataset. In this project we have explored multiple approaches to build recommender models. Our first recommender system uses an unsupervised learning approach using LDA (Latent Dirichlet Allocation) to recommend videos. We then worked on building a content based recommender system using TF-IDF embeddings and cosine similarity. This model was later improved by performing more data pre-processing and utilizing more relevant features to improve our recommendations. We then experiment with BERT (Bidirectional Encoder Representations from Transformers) to create word embeddings. At last, in order to make the recommendations more relevant to the user, we include user feedback in our model.For this we take inspiration from the Rocchio algorithm to develop a feedback based recommendation system. This model builds on top of our content based recommender system and has the capability of achieving the desired precision in the recommendations.

Dataset: The YouTube dataset consists of about 40,881 videos that are available on youtube. It consists of 16 columns each of which contains different information regarding a video like video_id, title, channel_title, category, description, tags etc.

Recommendations using LDA (Latent Dirichlet Allocation): Latent Dirichlet Allocation is an unsupervised NLP model used for discovering topics from a document. It is a popular topic modeling technique that works on a principle similar to clustering on numerical data. LDA is extremely useful in delineating hidden patterns in a collection of words. In this project, LDA has been used for recommending videos based on similarity with the given input video. The following steps have been followed while developing this model:

Step 1- Creating a new dataframe
The original dataframe (without duplicates) is re-constructed by dropping all the remaining columns except for ‘title’, ‘channel_title’, ‘tags’ and ‘description’ columns. Further the channel_title, tags and description columns were grouped together into a a column named ‘overview’. The cleaned data frame now has two columns ; "title" representing the name of the video and "overview" with all the necessary information pertaining to the video.
Step 2- Text Preprocessing : Preprocessing of textual data has been executed in four stages:
(i) Removing non-english words (ii) Removing Stop words
(iii) Building Bigrams (iv) Lemmatization
Step 3- Initializing LDA Model : LDA has four assumptions:
(i) Each document is a bag of words.

(ii) Stop words do not carry any information about the topic.
(iii) The number of topics(k) are known beforehand.
(iv) Except for the word in question, all other topic assignments are accurate.

By considering these assumptions, the LDA model is initialized and k value is assigned to be 10 (for generating top 10 recommendations). LDA automatically finds the most used keywords in the ten topics constructed and arranges them according to their probability scores.
Step 4- Building the Recommender System : The recommender system is designed such that it operates on the basis of the probability scores. When a test title is given as the input, the model calculates the probability of that title belonging to one of the 10 topics. The system treats it like a classification problem. The videos with the top 10 highest scores are taken into consideration and are suggested as recommendations to the user.

Content Based Recommender system using TF-IDF embeddings: A content based recommender system recommends videos based on the similarity between the description, tags etc. with the input video. It identifies the similarity between the products based on their descriptions.

Step 1- Our first step is similar to what we performed while building the LDA model where we perform data analysis, clean the data and come up with an 'overview' column that combines text data describing the video.
Step 2- We use TF-IDF to convert the textual data in the 'overview' column into a matrix of vectors. The TfidfVectorizer function is used to create raw documents into a matrix of TF-IDF features.
Step 3- The 'title' of the youtube videos (the feature that we take in as input) is assigned indices and the vectors created in the previous step are then scored against each other using the cosine similarity
Step 4- The model takes a video title as input from the user and uses cosine similarity function to find pairwise similarity with all other videos in the dataset. We then sort the videos in descending order based on similarity with the input and recommend the top 10 most similar videos.

Video statistics based adjustments : Our TF-IDF implementation took 'title', 'channel_title', 'tags', and 'description' columns into consideration to make recommendations for videos. To assimilate real life search recommendation scenarios, video statistics should also be taken into account. After we generate the top 10 most similar videos, some of these recommendations might have millions of views while others might just have a few hundred views only. In this approach, we define a threshold for the similarity score (0.05 in our case). We only consider videos with similarity score more than this threshold and rank them in decreasing order of the number of views.

Content Based Recommender system using BERT embeddings:
Bidirectional Encoder Representation from Transformers (BERT) is applying bidirectional training of transformers to language modeling. We used BERT embeddings and found that the

results were better than TF-IDF embeddings. BERT learns the context of the word and hence the recommendations are better as explained in the analysis section.

Content based Recommendations including user feedback: All the above implementations do not take into account feedback from the user. Our next model builds on top of the content based recommender using TF-IDF and uses an algorithm inspired by the Rocchio algorithm to include user feedback. This model gives us the flexibility to set a precision value and iterates until the required precision is achieved. The system shows the initial top 10 recommendations to the user and asks the user to mark each recommendation as 'relevant' or 'not-relevant'. Then, taking this feedback into account we use the equation mentioned below to calculate the new 'tags' that are then appended into a temporary data frame which is then used to re-calculate the similarity scores and recommend more relevant videos hence improving precision of our recommendations and making them more relevant for a specific user.

$$\text{Augmented Tags} = (0.8)*R - (0.1)*NR$$

where R : sum of embeddings of all relevant documents
NR : sum of embeddings of all non-relevant documents
The embedding matrix is generated for the youtube videos where each row represents a tag as a row vector and every column vector represents a youtube video.
The values of constants were chosen after trying out a few combinations given in literature. The resultant column vector is then sorted in the decreasing order and based on the indices of top 15 tags, the tags are chosen from vocabulary and appended to the 'overview' feature of the current video. This new embedding is used to calculate the cosine similarities and recommend new relevant videos.

The transcript of the run of the system has been attached and the results can be seen to be slightly better. The dataset has limitations in the sense that it has less records that are vague to show actual improvement in the subsequent iterations. The dataset is also static and hence, this would be better on dynamic datasets like youtube where tons of videos are added every minute.
Future scope also includes implementing the algorithm on other features such as 'description', 'title' and additional words can be generated out of these features to be appended to the 'overview' feature on which cosine similarity is calculated.

Evaluation: In order to evaluate the quality and relevance of videos recommended, we used both manual inspections and category-based metrics. Since we would want the recommended videos to be of the similar category as the searched video titles, we can use the category_id from the trending videos dataset to measure the level of resemblance of our recommendations. We performed the evaluation as follows: we randomly selected the titles of 100 videos from our dataset and used each algorithm to give recommendations. For the top 10 recommendations, we used 10 boolean values to represent whether each video is of the same category with the searched title and took the mean of all recommendations to get an estimate of the 'relevance score' for each algorithm. For recommendation systems using TF-IDF, the relevance score is calculated to be around 78%. For recommendation systems using LDA, the relevance score is

above 90%, showing good topic modeling results. For the BERT recommendation systems, the relevance score is around 88%, as it benefits from pre-trained embedding weights, showing good semantic resemblance as well. From these statistics, we know that all systems achieve a satisfying level of relevance when it comes to video recommendations because the category information is not available to the model, thus standing as an unbiased evaluation metric.

Conclusions: In this project we have used the YouTube video dataset from kaggle and have explored different approaches to build recommender systems. The output received after passing the same input to the different models can be seen below. Outputs from the model that uses the BERT embeddings are different as compared to the recommendations given by TF-IDF embeddings. For the specific input mentioned below, the BERT model outputs songs by different artists of similar genres while the TF-IDF embeddings have songs from the same artist'. This makes logical sense, since BERT tries to understand the context of each word as compared to tf-idf which is based on the weights of the words present in each documen

Discussions: Some of the limitations of these systems include that the models are trained on a limited dataset (popular YouTube videos in California region). It would be beneficial to train and test these models on a larger dataset (for example, the joint dataset of videos from all English-speaking countries).
Another obvious improvement to the systems is that we can use an ensemble approach where we use multiple recommendation systems and generate final recommendation.This can help us create more relevant recommendations. For example, a model that produces final recommendations via majority vote from multiple models may have the potential of providing more relevant recommendations.

Output of the systems for the input : **'Eminem - Walk On Water (Audio) ft. Beyoncé'**

LDA based Recommendation system :

```
recommend_by_title('Eminem - Walk On Water (Audio) ft. Beyoncé', df)
```

```
['Critical Role | Campaign 2 Episode 9',
 'Steam Code How To Get Free And Easily',
 "Marvel Studios' Black Panther - Warriors Of Wakanda",
 "Ufc 223: Khabib Nurmagomedov Reacts To Tony Ferguson'S Injury, Max Holloway Stepping In",
 'If He Were In Max Holloway'S Spot, Khabib Nurmagomedov Says He Wouldn'T Have Accepted Ufc 223 Fight
 "Snooki Explains Why She Fears Her Marriage Is Over On 'Jersey Shore: Family Vacation' (Exclusive)",
 'Pharmarusical (Season 10)',
 "Dc'S Legends Of Tomorrow 3X16 Promo I, Ava (Hd) Season 3 Episode 16 Promo",
 'John Mayer On Andy Cohen'S Annoying Habit | Wwhl',
 'Prank Hilarant – Youtube Hero #6']
```

Content based Recommendation system using tf-idf embeddings :

```
# Testing the recommendation system
give_rec('Eminem - Walk On Water (Audio) ft. Beyoncé')
```

```
1226     Walk On Water/Stan/Love The Way You Lie (Medle...
8020     Eminem - Walk On Water (Official Video) ft. Be...
5068                          Eminem - Untouchable (Audio)
6396                 Eminem - River (Audio) ft. Ed Sheeran
27728                                       Eminem - Framed
23871    Eminem - River (Behind the Scenes) ft. Ed Sheeran
7236                          Eminem - River ft. Ed Sheeran
6894                      Eminem - Believe (Official Audio)
265      Eminem Performs 'Walk On Water' | MTV EMAs 201...
7734     Eminem – River (Lyrics / Lyric Video) ft. Ed S...
Name: title, dtype: object
```

The output after consider the video statistics based adjustments (views):

```
startsearch()
```

```
Enter your search:
Eminem - Walk On Water (Audio) ft. Beyoncé
1 Eminem - Walk On Water (Official Video) ft. Beyoncé by EminemVEVO
2 Eminem - Walk On Water (Audio) ft. Beyoncé by EminemVEVO
3 Eminem Performs 'Walk On Water' | MTV EMAs 2017 | Live Performance by MTV International
4 Walk On Water/Stan/Love The Way You Lie (Medley/Live From Saturday Night Live/2017) by EminemVEVO
5 30 Seconds to Mars - Walk on Water (R3hab Remix) by Proximity
6 EMINEM Coachella 2018 (Full Live Performance) [Dr. Dre, 50 Cent & 2Pac] by Hip-Hop Universe
7 Eminem - River (Audio) ft. Ed Sheeran by EminemVEVO
8 Eminem - Untouchable (Audio) by EminemVEVO
9 Eminem - Nowhere Fast (Extended/Audio) ft. Kehlani by EminemVEVO
10 Eminem - Believe (Official Audio) by The Best for Loyalty
```

Content based Recommendation System using BERT embeddings:

```
[ ] # Testing the recommendation system
    give_rec('Eminem - Walk On Water (Audio) ft. Beyoncé')
```

```
27728                                      Eminem - Framed
23871    Eminem - River (Behind the Scenes) ft. Ed Sheeran
21667      DJ Khaled ft. JAY Z, Future & Beyoncé - Top Off
21349    DJ Khaled - Top Off (Ft. JAY Z, Future & Beyonce)
6391     G-Eazy - No Limit REMIX (Audio) ft. A$AP Rocky...
33765    Enrique Iglesias - MOVE TO MIAMI (Official Vid...
647                      Remy Ma - Wake Me Up ft. Lil' Kim
7196     G-Eazy - No Limit REMIX ft. A$AP Rocky, Cardi ...
32323              Joyner Lucas & Chris Brown - I Don't Die
32707    Enrique Iglesias, Pitbull - Move To Miami (Lyr...
Name: title, dtype: object
```

: