# **YouTube Video Title Generator using LSTM**

**Group Members:**

|  |  |  |
| --- | --- | --- |
| **First name** | **Last Name** | **Student number** |
| Abhishek | Naithani | c0871411 |
| Karishma | Shirsath | c0871245 |
| Tejaswi | Kalla | c0852124 |
| Venkata Sai Charan | Yerapasetty | c0863224 |

**Instructor:** **Vahid Hadavi**

**Class: AML –2304 2 (Natural Language Processing)**

**Date:** 12-08-2023

Contents

[**YouTube Video Title Generator using LSTM** 1](#_Toc142994173)

[**Abstract** 3](#_Toc142994174)

[**Introduction** 3](#_Toc142994175)

[**Data Pre-processing** 4](#_Toc142994176)

[**Data Cleaning and Validating** 4](#_Toc142994177)

[**Data Visualization** 6](#_Toc142994178)

[**Methods and Model Used** 10](#_Toc142994179)

[**Result** 13](#_Toc142994180)

[**Conclusion and future work** 13](#_Toc142994181)

[**References** 14](#_Toc142994182)

# **Abstract**

  In this project, we propose the development of a YouTube video title generator utilizing Long Short-Term Memory (LSTM) networks. The primary objective is to create engaging and captivating video titles for YouTube content creators. The data is taken from Kaggle for three countries. Data contains three csv files for the YouTube videos the data is for following three countries- US, Canada, UK. It contains information about various Youtube videos, such as their Video Id, Trending Date, Title, Channel Title, and other relevant details. This report presents a neural network model which is LSTM, for the title generation. It covers some of the crucial data wrangling steps to format the data for analysis and visualization.

# **Introduction**

Title generator is a natural language processing task and is a central issue for several machine learning, including text synthesis, speech to text, and conversational systems. To build a model for the task of title generator or a text generator, the model should be trained to learn the probability of a word occurring, using words that have already appeared in the sequence as context. In this case, we will use TensorFlow2, Keras to obtain text processing which includes:

* Tokenization
* Sequence
* Padding

Through this report, we will use the YouTube trending videos dataset and the Python programming language to train a model of text generation language using machine learning, which will be used for the task of title generator for YouTube videos or even for your blogs. Data cleaning, validating, transforming, and various visualization plots will be understood along with data analysis. recurrent neural network architecture consisting of LSTM model is used to generate a recommendation system that will give a YouTube video Title based on the given word and number of words expected in title as input parameters.

# **Data Pre-processing**

We will start this task to build a title generator with Python and machine learning by importing the libraries and by reading the datasets. Dataset is taken from Kaggle which contains CSV file for YouTube videos from three countries – UK, US, Canada. We are combining all three CSV file into one dataset.

The concatenated ‘CSV’ file and converted to a data frame, a highly efficient Pandas tool of the Python library. This data frame is crucial for manipulating and integrating with other libraries like TensorFlow, Scikit-learn, Matplotlib, Seaborn, etc. The dimension of the raw data is 120746 rows and 16 columns. The data type of each attribute, the dispersion (descriptive statistics), and the presence of missing and duplicate values are checked before preparing the data for further analysis. Data pre-processing is done using Spacy. We are doing text processing on column ‘Title’ of the videos. We are doing following steps on title column:

• Stop-Words removal. • Punctuation Removal. • Removing special characters.

After the pre-processing we are storing ‘Titles’ in new list of words i.e. Normal and Popular. We have 27416 Normal words and 2902 Popular words.

# **Data Cleaning and Validating**

**Removing Duplicate Values:**

Considering the data there are chances of data having duplicate row values. We are checking duplicate values based on video\_id column. After checking we found that there are duplicate values in our concatenated dataset. After removing the duplicate new data shape is 30318 rows and 16 columns.

**Transforming Columns:**

The nominal categorical column (‘Category\_id') is used for getting data into a new column called as ‘category\_names’. In our data frame we are adding new column in which we are getting 5 different categories of video.

**Handling Outliers:**

An outlier is a data point that is abnormally high or low in comparison to the closest data point and the rest of the values in the dataset. A deeper statistical analysis of the data is necessary to categorize a data point as an outlier since some of the genuine extreme values might impact the result of the analysis. Understanding the spread of the data, sorting, and splitting the data points helps in efficiently determining each (commonly 4) quartile of the data. Calculating the Inter Quartile Range is a crucial step in finding out the outliers. Monitoring the

outliers should be based on research and domain knowledge to avoid bias and distortion in the data.

We have observed that based on views we can find that we have outliers. The three techniques to address outliers are explored.

* **Quantile based Flooring and Capping:**

It involves setting a lower bound (flooring) and an upper bound (capping) on the values of a variable based on certain quantiles of the data distribution. We have chosen the **5th and 95th percentiles** of the data points as the floor and cap values that replace the points that are below and above this threshold. This way the data is constrained to have designated minimum and maximum values that improve the stability of the analysis results.

* **Trimming:**

It is a process of removing (truncating) that are present below and above specified threshold quantiles. The defined lower and upper thresholds are the **30th and 95th percentiles** of the data respectively. This method is advisable when the outliers are rare and not relatively significant.

* **Log Transformation:**

Data undergoes a mathematical process known as a log transformation to lessen the proportion of extreme values and improve the symmetry of the data distribution. It computes the logarithm of the data, typically in base 10 (log10) or the natural logarithm (ln). This method shows the best results especially when the data is skewed and does not contain zero or negative values for outliers.

Through visualizing the boxplots of the above-mentioned methods, it is observed that ‘Trimming’ has given decent results and retained most of the data. While Log Transformation has shown some distortion in the distribution of the data. Therefore the data after trimming the outliers is used for clustering.

# **Data Visualization**

It is an interactive form of understanding Big Data and simplifies complex relationships, patterns, and trends to make informed business decisions.  We have plotted mostly used words, Bi-grams, Tri-grams, videos by their categories. We have plotted pandas profiling report using columns – Views, Like, Dislike and Comment count.

Below is the screenshot of Pandas Profiling Report:

**A screenshot of a computer

Description automatically generated**A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Bar plot of Top 20 Channels by views:**

A graph with blue lines

Description automatically generated

**Bar Plot of** **Most Popular words:**

A graph with blue and white bars

Description automatically generated

**Bar plot of Bi-grams:**

A graph with orange and white bars

Description automatically generated

**Bar plot of Tri-grams:**

A graph with red lines

Description automatically generated

**Pie chart of Videos by their Category:**

A colorful pie chart with numbers and a number of different colored circles

Description automatically generated with medium confidence

It is inferred through the above plot that Among the videos, 32.2 percent are Entertainment category videos followed by ‘News & Politics’ being second largest share of percentage.

# **Methods and Model Used**

**TfidfVectorizer:**

Tf-idf analyzes the impact of tokens (words) throughout the whole documents. For example, the more times a word appears in a document (each title), the more weight it will have. However, the more documents (titles) the word appears in, it is 'penalized' and the weight is diminished because it is empirically less informative than features that occur in a small fraction of the training corpus ([source](https://www.kaggle.com/adamschroeder/countvectorizer-tfidfvectorizer-predict-comments))

tf(t)= the term frequency is the number of times the term appears in the document

idf(d, t) = the document frequency is the number of documents 'd' that contain term 't'

**Generating Sequences:**

Natural language processing tasks require input data in the form of a sequence of tokens. The first step after data cleansing is to generate a sequence of n-gram tokens.

An n-gram is an adjacent sequence of n elements of a given sample of text or vocal corpus. Elements can be words, syllables, phonemes, letters, or base pairs. In this case, the n-grams are a sequence of words in a corpus of titles. Tokenization is the process of extracting tokens from the corpus

**Padding the sequences:**

Since the sequences can be of variable length, the sequence lengths must be equal. When using neural networks, we usually feed an input into the network while waiting for output. In practice, it is more efficient to process data in batches rather than one at a time.

This is done by using matrices [batch size x sequence length], where the length of the sequence corresponds to the longest sequence. In this case, we fill the sequences with a token (usually 0) to fit the size of the matrix. This process of filling sequences with tokens is called filling. To enter the data into a training model, I need to create predictors and labels.

I will create sequences of n-gram as predictors and the following word of n-gram as label

**Why are we using LSTM Model:**

In recurrent neural networks, the activation outputs are propagated in both directions, i.e. from input to output and outputs to inputs, unlike direct-acting neural networks where outputs d activation are propagated in only one direction. This creates loops in the architecture of the neural network which acts as a “memory state” of neurons.

As a result, the RNN preserves a state through the stages of time or “remembers” what has been learned over time. The state of memory has its advantages, but it also has its disadvantages. The gradient that disappears is one of them.

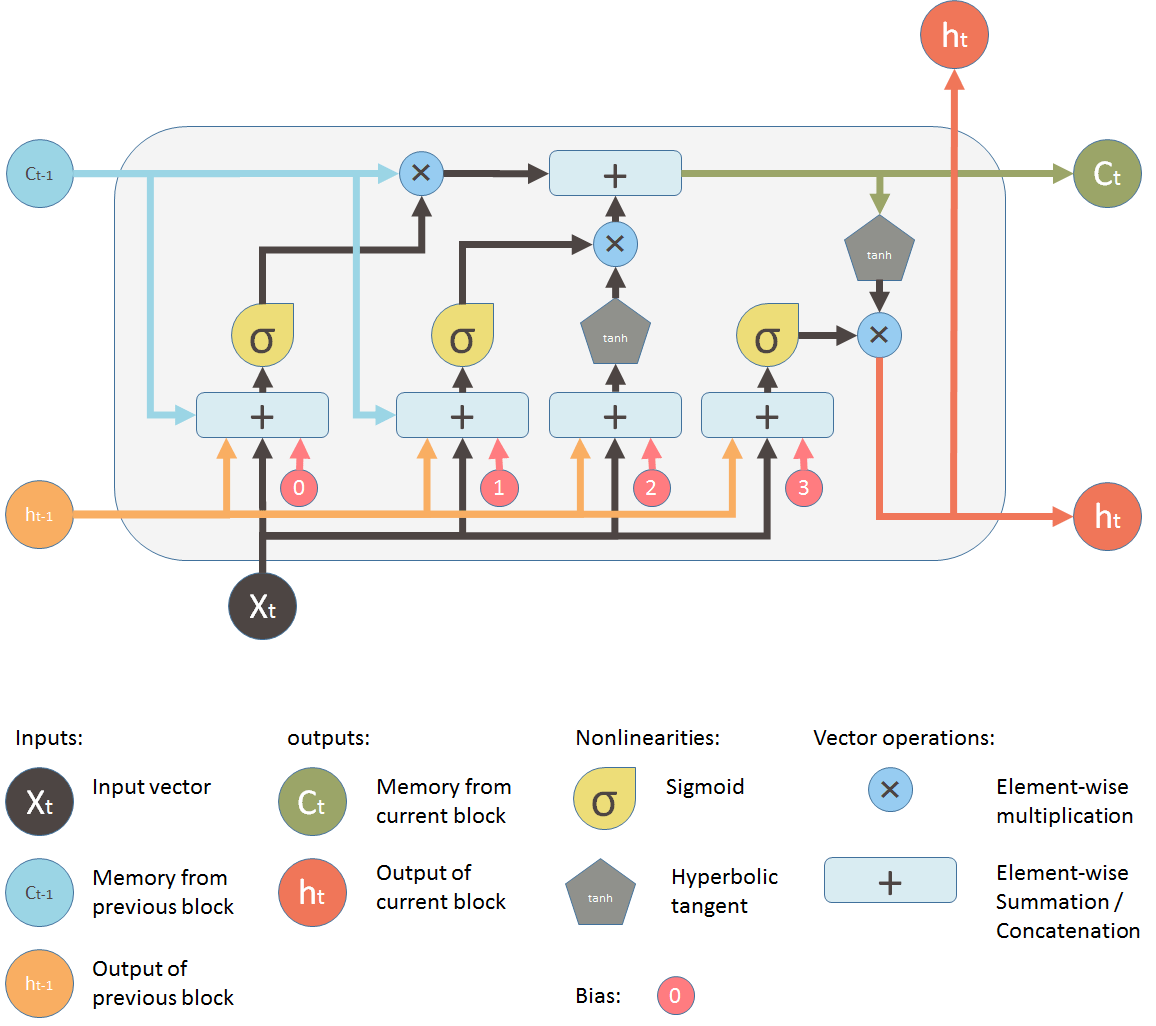
In this problem, while learning with a large number of layers, it becomes really difficult for the network to learn and adjust the parameters of the previous layers. To solve this problem, a new type of RNN has been developed; LSTM (long-term memory).

**Title Generator with LSTM Model:**

The LSTM model contains an additional state (the state of the cell) which essentially allows the network to learn what to store in the long term state, what to delete and what to read. . The LSTM of this model contains three layers:

* Input layer: takes the sequence of words as input
* LSTM Layer: Calculates the output using LSTM units.
* Dropout layer: a regularization layer to avoid overfitting
* Output layer: calculates the probability of the next possible word on output.

We will run this model for total 20 epoochs but it can be experimented further

The input word is first tokenized, the sequence is then completed before being passed into the trained model to return the predicted sequence

# **Result**

Now as we have created a function to generate titles let’s test our title generator model:

**Input:**

We have provided words with the number which is a number of words we are expecting in output title after the original word:

A screen shot of a computer program

Description automatically generated

**Output:**

Kardashian React Lele Pons

Superman Official Trailer

United States Vs Bangladesh

Fight Official Trailer Hd 20Th Century

# **Conclusion and future work**

In conclusion LSTM model is generating promising results. Again, we can have far better results than this! If we know what we would like to have as a result, we can try to clean the dataset and slightly change the texts we have, we can use far more texts and far more powerful models (for example increasing the number of RNN units and layers). In future we will run pipeline for ‘English’ language words only and try to run the model with ‘50’ epoch to get more precise output.

# **References**

[1] Konstantin Lopyrev. Generating news headlines with recurrent neural networks. *arXiv preprint arXiv: 1512.01712*, 2015.

[2] Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *EMNLP 2015*, pages 379–389, 2015.

[3] P. M. Hanunggul and S. Suyanto, “The Impact of Local Attention in LSTM for Abstractive Text Summarization,” 2019 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), Yogyakarta, Indonesia, 2019, pp. 54–57, DOI: 10.1109/ ISRITI48646.2019.9034616.

[4] Rong Jin and Alexander G. Hauptmann. (2001). Automatic Title Generation for Spoken Broadcast News. In Proceedings of HLT-01, 2001, pp. 1´lC3.

[5] T. Luo, K. Guo, and H. Guo, “Automatic Text Summarization Based on Transformer and Switchable Normalization,” 2019 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom), Xiamen, China, 2019, pp. 1606–1611, doi: 10.1109/ISPA-BDCloud-SustainCom-SocialCom48970.2019.00236.