TITLED SECTION IN VIDEO

B.Tech Project Report

By

Nitin Kumar - 1610110236

Pranjal Sharma -1610110261

Under the supervision of

Dr. Sonia Khetarpaul

Assistant Professor

Submitted in the partial fulfillment of requirements for B. Tech. in Computer Science and Engineering

SHIV NADAR UNIVERSITY

Department of Computer Science and Engineering, School of Engineering, Shiv Nadar University, Gautam Buddha Nagar, U.P., India, 201314

Declaration

We declare that this written submission represents my ideas in my own words and wherever others' ideas or words have been included, we have adequately cited and referenced the original sources. we also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Name of the Student - Nitin Kumar Signature and Date Nitin 15/05/20

Name of the Student - Pranjal Sharma Signature and Date Pranjal 15/05/20

Acknowledgement

We would like to thank Dr. Sonia Khetarpaul, Assistant Professor, Shiv Nadar University for giving us an opportunity to do our project under her guidance and mentorship, for valuable suggestions and keen interest in the progress of our project.

We are grateful to the Department of Computer Science, Shiv Nadar University for providing all the necessary resources for the successful completion of my course work.

Table of Contents

List of Figures

1. Abstract

Video and Image processing are very interesting topics in the field of research and data science especially deep learning. Image processing is any form of signal processing for which the input is an image, such as photographs or frames of video. In our project, we have applied different techniques to gather data from video (i.e., from audio and frames). This data will then be processed using NLP and different deep learning models to generate suitable topics and summaries related to the given text. This means exploring two interesting areas of data science, ie Topic Modelling and Deep Learning.

Topic modeling is a powerful technique for unsupervised analysis of large document collections. Topic models conceive latent topics in text using hidden random variables, and discover that structure with posterior inference. Topic models have a wide range of applications like tag recommendation, text categorization, keyword extraction and similarity search in the broad fields of text mining, information retrieval, statistical language modeling. Different Models have been found effective for different languages because of their unique morphological structure.In this research LDA and its hybrid with word2vec known as lda2vec was chosen for this project. LDA is a matrix factorization technique and as our observation in this research/project, we found it's hybrid lda2vec to be more suitable for generating topics.

As for generating summaries, abstractive text summarization aims to shorten long text documents into a human readable form that contains the most important facts from the original document. However, the level of actual abstraction as measured by novel phrases that do not appear in the source document remains low in existing approaches. Therefore, we used LSTM (deep recurrent neural networks) to generate good summaries.

So, in the end, our main challenge was to develop a reliable system that anyone can use to process any video for generating topics and chapters in multiple parts of video. Next we will define requirements, algorithms, paradigms related to the project and show how it can be done.

2. Abbreviations

API - Application Program Interface

ANN - Artificial Neural Networks

BERT - Bidirectional Encoder Representations from Transformers

CBOW - Continuous Bag of Words Model

CPU - Central Processing Unit

CUDA - Compute Unified Device Architecture

cuDNN - CUDA Deep Neural Network

Doc2Vec - document to Vector

GPU - Graphics Processing Unit

IDE - Integrated Development Environment

LDA - Latent Dirichlet Allocation

LDA2Vec - Latent Dirichlet Allocation to Vector

LSA - Latent Semantic Analysis

LSTM - Long Short Term Memory

NLP - Natural Language Processing

OCR - Optical Character Recognition

OS - Operating System

PLDA - Probabilistic Latent Dirichlet Allocation

RAM - Random Access Memory

RNN - Recurrent Neural Networks

Seq2Seq - Sequence to Sequence Model

VRAM - Video Random Access Memory

Word2Vec - word to vector

3. Introduction

We propose our idea to remove the manual interaction of adding titled section by user to add titled section (and basic summary of what is happening in that section) by using natural language processing and deep learning to provide a precise and efficient system that will process the video and automatically add titled sections and summary.

Our project is mainly divided in two parts -

1. Processing video to gather information from every frame
2. Processing data from audio

3.1 Chapters in videos

Watching a long informative video can get tiring, but if video is divided in certain parts it can become a lot easier to understand.

This is where the concept of chapters come into play. When a user plays a video, they can click on a particular titled section on the video timeline to easily navigate to their point of interest, or immediately jump to new content. This means no more scrubbing through the progress bar in videos. Chapter markers let's highlight key moments in the video that viewers can jump to instantly. Whether it’s a lecture, a film, or an archived livestream, long-form video is easier to digest in sections.

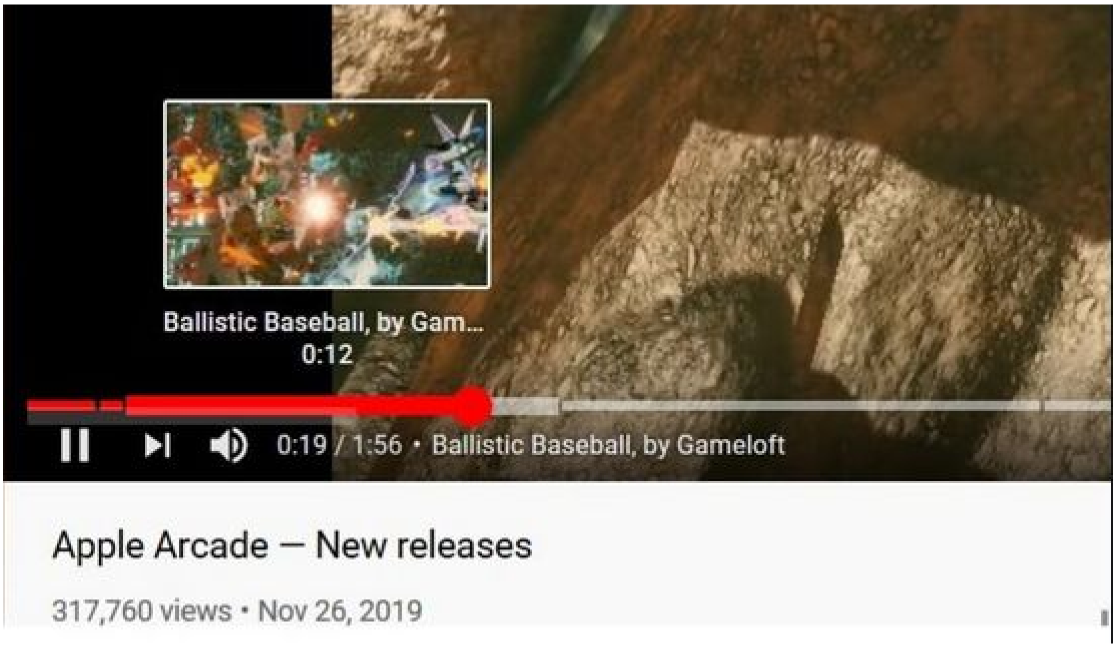

Ballistic Baseball, by Gam...
0:12
0:19 / 1:56 • Ballistic Baseball, by Gameloft
Apple Arcade — New releases
317,760 views • Nov 26, 2019

Figure - Chapter in a video [12]

This is mainly quite prominent in demonstrations, tutorials, and other educational videos. But adding such titled sections require manual work and can be quite time consuming especially when working on multiple videos.

3.2 Summary of content (sections)

At most, people tend to add titled sections inside videos that show titles of the specific content at specific timelines. But we went one step further to add a short summary of what exactly happens in that section.

Adding such a summary is no easy task. Although there are multiple methods available to facilitate the same, these methods rely heavily on the exact text present in order to make a short summary.

Add a large amount of time it takes to generate a summary, it seems infeasible to spend time to generate any summary for the viewer.

3.3 Video processing

Often, we have to capture a long live stream or video with a camera. But, adding chapters or summary inside such a video is a hectic task especially if the video itself is hours long. No one would want to do it manually. Our whole idea is to automate the processes involved with adding summary/titles and then processing the video with a decent hardware so it's ready to be shared.

This processing of video involves different tasks such as extracting audio and converting video into frames (since a video is a collection of frames/images which changes in a very fast motion) to extract data. This data is then processed to form titles/summary which is then added to the video at specific timelines.

Image Analysis is a very common field in the area of Computer Vision. It is the extraction of meaningful information from videos or images. In python, OpenCV library can be used to perform multiple operations on videos.

3.4 Motivation

In recent times, videos have become a primary source of entertainment as well as education for a lot of users. With a lot of videos with ever increasing length used to study across the field.

The main goal is to get topics and summaries from video. This will make getting to know the content of videos easier, helping us to browse the contents of video. This can help save a lot of time and energy.

3.5 Scope

We went with our project with the idea that our project will be able to process any video, be it educational, news, sports, long live streams, etc., generate relevant data related to that video, process it accordingly and put back final processed data back into the original video without any human interaction.

This would mean developing sophisticated deep learning models for both processing data for titles and processing data for generating summary.

We have used natural language processing with deep learning to develop the models and the models can always be further improved by adding more data to it.

4. Related Work

For text modeling there have been many models available well before the first decade of the century. The LDA is related to LSA and PLSA. Word2Vec is related to Doc2vec and other models.

[13] LSA is among the foundational techniques in topic modeling. The core idea is to take a matrix of what we have （ docu ments and terms ） and decompose it into a separate document (topic matrix and a topic term matrix).

- Generate document term matrix.
- We can construct an a× b matrix. By using ‘a’ documents and ‘b’ words.
- Each entry can be anything like a count of the number of times the mth word appeared in the nth document.
- LSA models use tf-idf score in the document term matrix to get good results.

PLSA uses a probabilistic method to tackle the same problem. LDA is a Bayesian version of PLSA. In particular, it uses dirichlet priors for the document-topic and word-topic distributions, lending itself to better generalization.

[14] Doc2vec method was a concept that was presented in 2014 by Mikilov and Le. Documents do not come in logical structures, so to create a numeric representation of a document Doc2Vec method was created.

[15] Google was the first company to introduce Seq2seq. Before that, the translation worked in a very naive way. Each word was converted to its target language. Seq2seq revolutionized the process of translation by making use of deep learning. It not only takes the current word/input into account while translating but also its neighborhood.

Currently, quite a lot of work is already done in the field of video and audio processing like using machine learning in videos to write basic sentences about content of video or adding subtitles based on audio, processing textual data to generate keywords using tf-idf, textrank, etc. But we are trying to process audio and video simultaneously so as to generate topics using topic modelling and deep learning fast and efficiently.

For summarization based methods, there are multiple ways to do “extractive based summarization” but summaries resulted from these only contain words present in the initial data. Therefore, they are more like reducing text than creating actual human-like summaries.

Another way is to use “abstraction based summarization”. There are ways like BERT summarization which are used in some domains. We decided to instead develop RNN Deep Learning model to generate better summaries which could be further improved by adding more data to the model. This results in a situation where we can keep improving the model, and other people can also use Transfer Learning to improve accuracy of similar models.

5. Technical Requirements

There are multiple hardware/software requirements for our project.

5.1 Hardware Requirements

- **CPU with high computational power** - A good CPU is required since we are processing a lot of data, both, for processing and modelling. Also, at least 4GB RAM will be required to ensure smooth processing of large amounts of data.

- **GPU with CUDA & cuDNN support and decent VRAM (at least 4GB)** - GPU is optional, but training model on GPU (with CUDA) is much faster (4-8x) than running on CPU. Also, with cuDNN installed, neural networks train much faster (around 2-3x times faster when training LSTM networks on GPU with cuDNN than without it). We used Google Colab's NVIDIA Tesla K80 GPU (8GB VRAM) which was sufficient to train the deep learning model for our project.

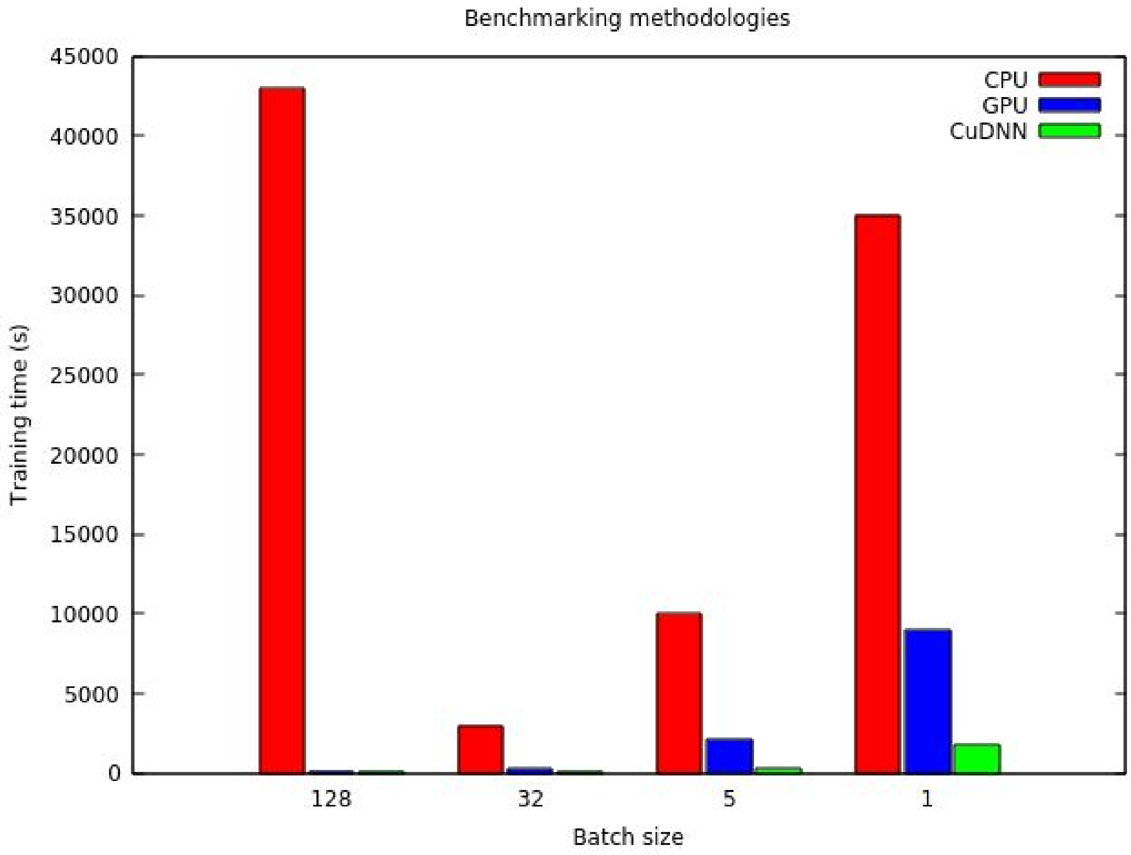


Figure - CPU vs GPU vs GPU (with cuDNN) [1]

This image shows that with increasing batch sizes, GPU progressively outperforms CPU.

For example, the parallel processing capabilities of GPUs accelerates the LSTM training and inference processes. GPUs are the de-facto standard for LSTM usage and deliver a 6x speedup during training and 140x higher throughput during inference when compared to CPU implementations. cuDNN is a GPU-accelerated deep neural network library that supports training of LSTM recurrent neural networks for sequence learning. [2]

5.2 Software Requirements

- **OS -** Windows 10 / Linux

  We used Arch Linux as a base for all IDEs and libraries.

- **IDE used -**
  1. Google Colab
  2. Jupyter Notebooks
  3. Python IDE (Spyder)

- **Important libraries -**
  1. Tensorflow (v2.2) - for Keras and Pytorch
  2. Keras
  3. Pytorch
  4. nltk - for natural language processing
  5. LSTM (using tensorflow, with cuDNN support)
  6. Preprocessing libraries like
  7. cudatoolkit - for GPU processing
  8. Gensim - for topic modelling and NLP

6. Feasibility Study and Methodologies

6.1 Technical Feasibility

- Prior programming skills are required.
- Experience in working with data and deep learning is needed.
- High hardware requirements, which was made feasible using google colab's CPU and GPU.

Technically feasible.

6.2 Operational Feasibility

- We made a model for generating topics which can be used on any video.
- We also made a model for generating summaries which can be used on any video.
- Model only needed to train once.

Operationally feasible.

6.3 Schedule Feasibility

- We have past experience of working with data.
- We have experience in Data Mining and Deep Learning Techniques.
- Although the model takes a long time to train, it only needs to be trained once.

Therefore, feasible.

6.4 Financial Feasibility

- Mostly used open source technologies are used to avoid proprietary software costs. Although IBM Watson's speech-to-text api is proprietary, google's cloud speech-to-text free service can be used as an alternative.
- We have used two publicly available datasets.

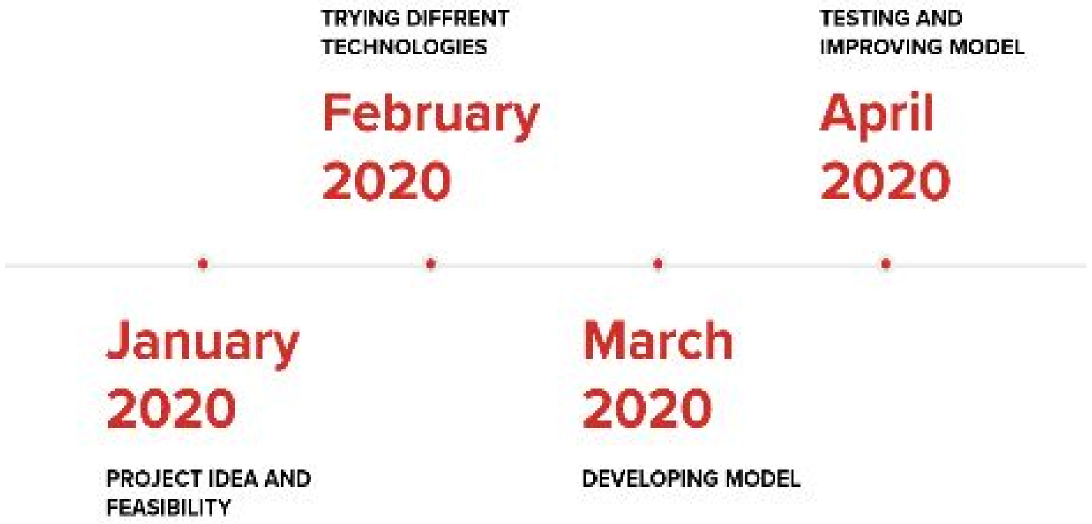
Financially feasible.

6.5 Risk Analysis

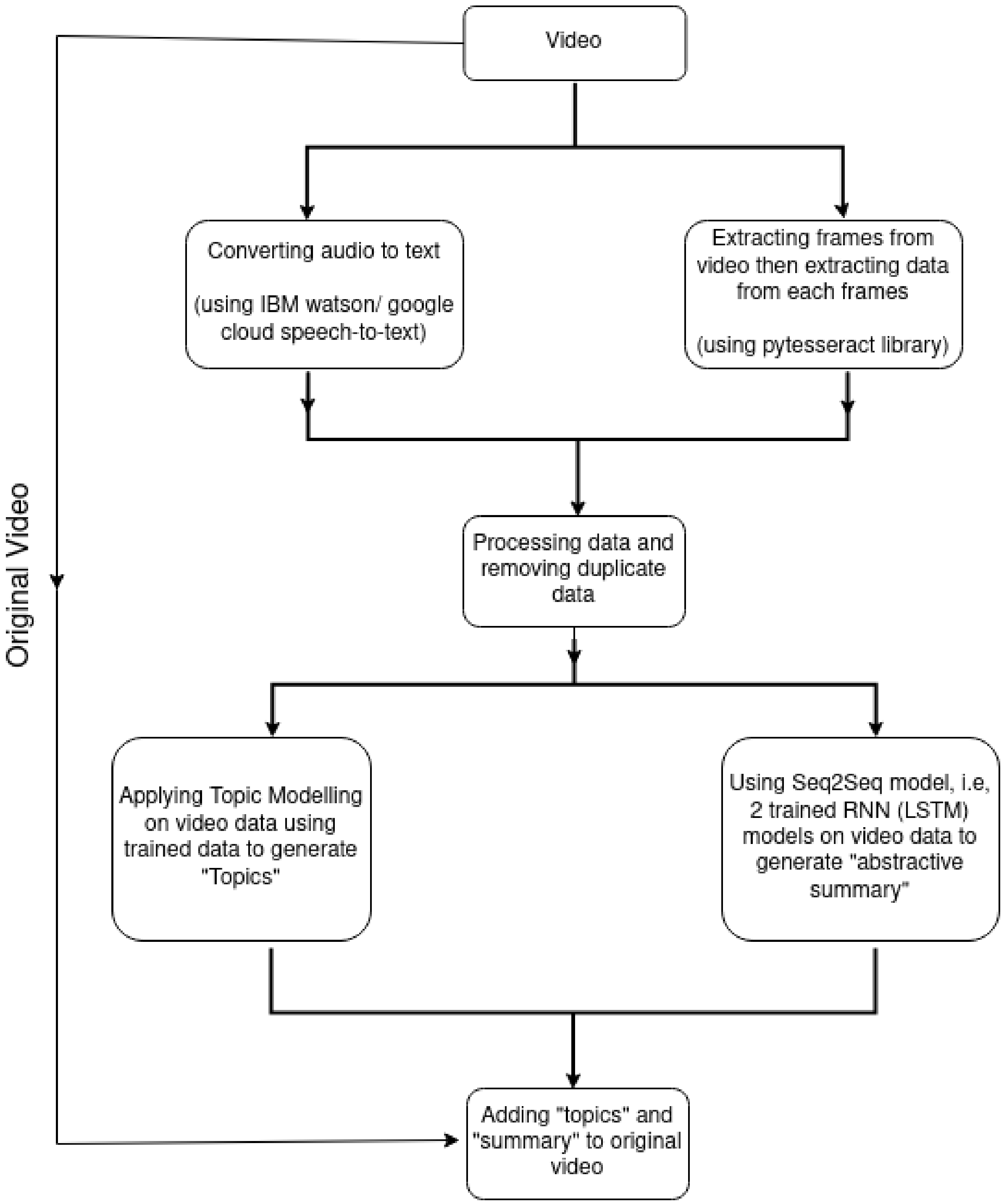
We used agile methodology to develop our project. We built modules one by one in a particular sequential order so that any module/library issue caused later could be minimized.

6.6 Conclusion

Analysing requirements, feasibility and all possible risks, we concluded our project to be feasible and to be worked upon. Dividing our project into smaller modules helped us successfully complete our project.


TRYING DIFFRENT TECHNOLOGIES
February 2020
TESTING AND IMPROVING MODEL
April 2020
January 2020
PROJECT IDEA AND FEASIBILITY
March 2020
DEVELOPING MODEL

7. DATAFLOW


Video
Converting audio to text
(using IBM watson/ google cloud speech-to-text)
Extracting frames from video then extracting data from each frames
(using pytesseract library)
Processing data and removing duplicate data
Applying Topic Modelling on video data using trained data to generate "Topics"
Using Seq2Seq model, i.e, 2 trained RNN (LSTM) models on video data to generate "abstractive summary"
Adding "topics" and "summary" to original video
Original Video

8. Titled Sections in Video

8.1 Topic Modelling

[16] Topic modeling is an unsupervised machine learning technique that's capable of scanning a set of documents, detecting word and phrase patterns within them, and automatically clustering word groups and similar expressions that best characterize a set of documents.

Here, we have used it to discover abstract "topics" that occur in a collection of documents.

Main types of topic modelling -

1. **LDA** - it is a generative statistical model that allows sets of observations to be explained by unobserved groups that explain why some parts of the data are similar. LDA is aimed mostly at describing documents by assigning topic distributions to them, which in turn have word distributions assigned.

    - Clustering document based on word usage
    - Use Bag-of-Words as a feature for clustering

    Working :

    1. Assume there are k topics across all of the documents
    2. Distribute these k topics across document m by assigning each word a topic.
    3. For each word w in document m, assume its topic is wrong but every other word is assigned the correct topic.
    4. Probabilistically assign word w a topic based on two things:
        - what topics are in document m

    - how many times word w has been assigned a particular topic across all of the documents

5. Repeat

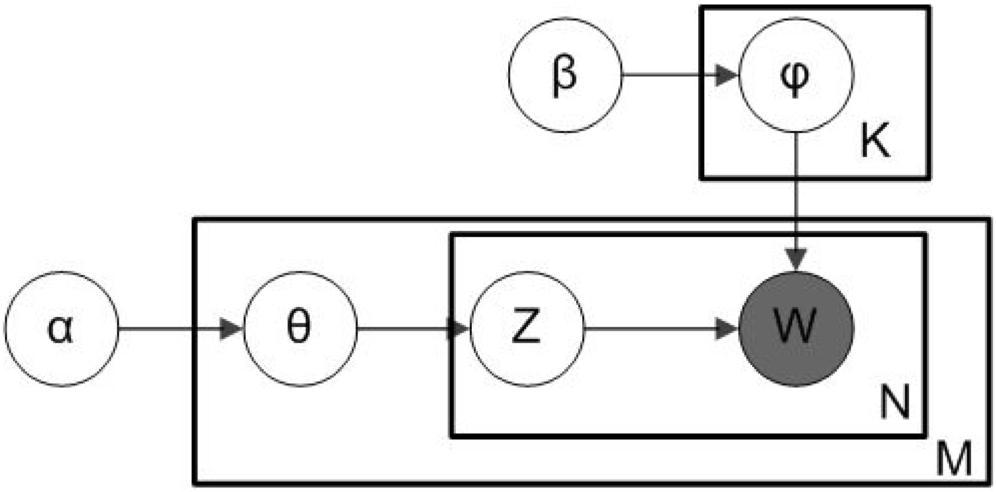


Figure -LDA Model [11]

Above is what is known as a plate diagram of an LDA model where:
α - per document topic distributions,
β - per-topic word distribution,
θ - topic distribution for document *m,*
φ - word distribution for topic *k,*
z - topic for the *n*-th word in document *m*, and
w - specific word [11]

2. **Word2Vec** - Word2vec is a two-layer neural net that processes text by "vectorizing" words, developed by Mikolov et al... Its input is a text corpus and its output is a set of vectors: feature vectors that represent words in that corpus. word2vec looks to embed words in a latent factor vector space.

Procedure to obtain the word embeddings:

1. Select one of the words as pivot in the text. The context words of the current pivot word are the words that occur around the pivot word, and the combinations of the pivot and context words is called a set of word-context pairs.

2. Two variants of the word2vec model exist
    a. In CBOW, the pivot word is predicted based on a set of surrounding context words (i.e. given a set of words the model has to predict a word most related to them). In CBOW, the order of context words does not matter.
    b. In the skip-gram architecture, the pivot word is used to predict the surrounding context words (i.e. given a word the model has to predict a set of words most related to the word).

The following image depicts the two different word2vec architectures.[10]

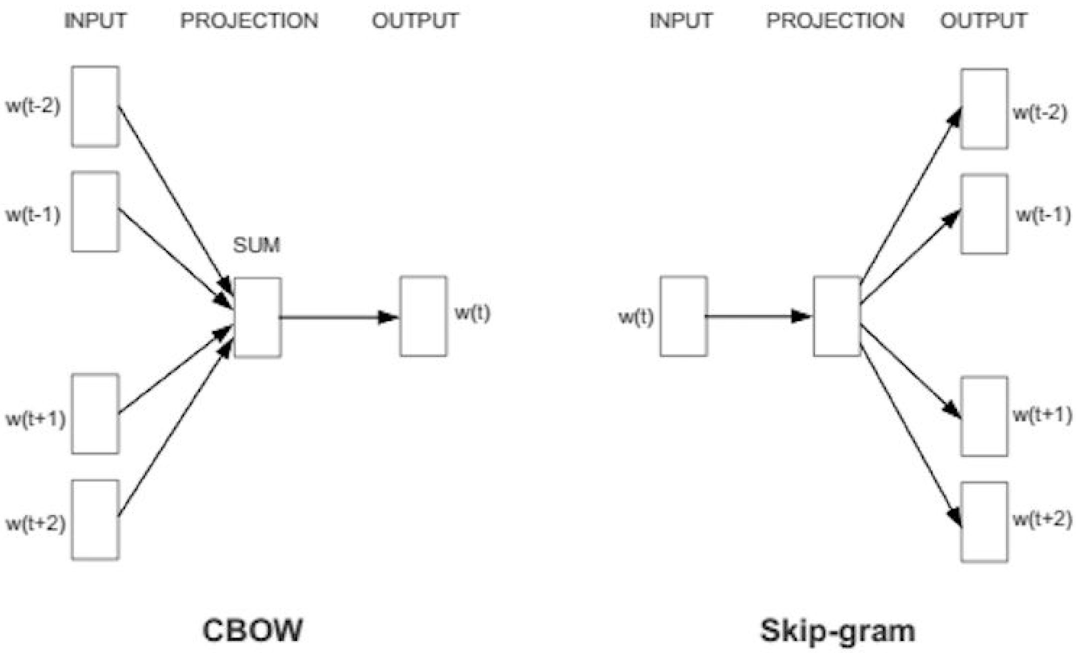


Figure - Word2vic cbow vs skip-gram [10]

- It is capable of capturing context of a word in a document
- predict context words based on a pivot

8.1.1 LDA2Vec

The goal of LDA2Vec is to make a huge amount of data useful to humans while still keeping the model simple to modify. It uses both word representations from word2vec and human interpretable document representations from LDA.

The LDA2Vec model tries to mix the best parts of Word2Vec and LDA into a single framework. Word2vec captures powerful relationships between words, but has a huge disadvantage as vectors are uninterpretable and don't represent documents. LDA on the other hand is quite interpretable by humans, but doesn't model local word relationships like word2vec. We build a model that builds both word and document topics, makes them interpretable.

Lda2vec absorbed the idea of "globality" from LDA. It means that LDA is able to create document (and topic) representations that are not so flexible but mostly interpretable to humans. Also, LDA treats a set of documents as a set of documents, whereas word2vec works with a set of documents as with a very long text string.

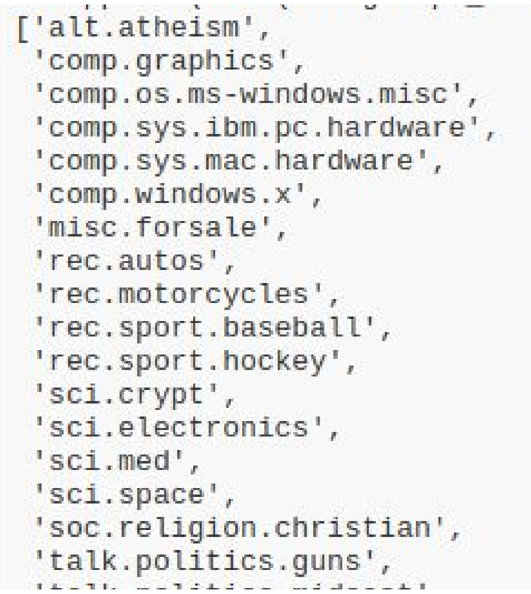
So, lda2vec took the idea of "locality" from word2vec, because it is local in the way that it is able to create vector representations of words (aka word embeddings) on small text intervals (aka windows).

Word2vec predicts words locally. It means that given one word it can predict the following word. At the same time LDA predicts globally: LDA predicts a word regarding global context (i.e. all set of documents).

8.2 Datasets

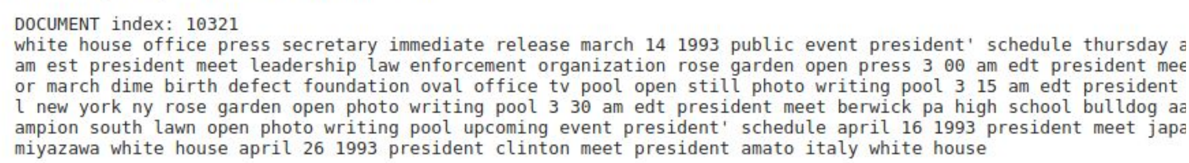
8.2.1 For Topic Modelling

The **20 newsgroups** dataset comprises around 18000 newsgroups posts on 20 topics.



```
['alt.atheism',
 'comp.graphics',
 'comp.os.ms-windows.misc',
 'comp.sys.ibm.pc.hardware',
 'comp.sys.mac.hardware',
 'comp.windows.x',
 'misc.forsale',
 'rec.autos',
 'rec.motorcycles',
 'rec.sport.baseball',
 'rec.sport.hockey',
 'sci.crypt',
 'sci.electronics',
 'sci.med',
 'sci.space',
 'soc.religion.christian',
 'talk.politics.guns',
 'talk.politics.mideast',
```

Figure - Topics of articles



```
DOCUMENT index: 10321
white house office press secretary immediate release march 14 1993 public event president' schedule thursday a
am est president meet leadership law enforcement organization rose garden open press 3 00 am edt president mee
or march dime birth defect foundation oval office tv pool open still photo writing pool 3 15 am edt president
l new york ny rose garden open photo writing pool 3 30 am edt president meet berwick pa high school bulldog aa
ampion south lawn open photo writing pool upcoming event president' schedule april 16 1993 president meet japa
miyazawa white house april 26 1993 president clinton meet president amato italy white house
```

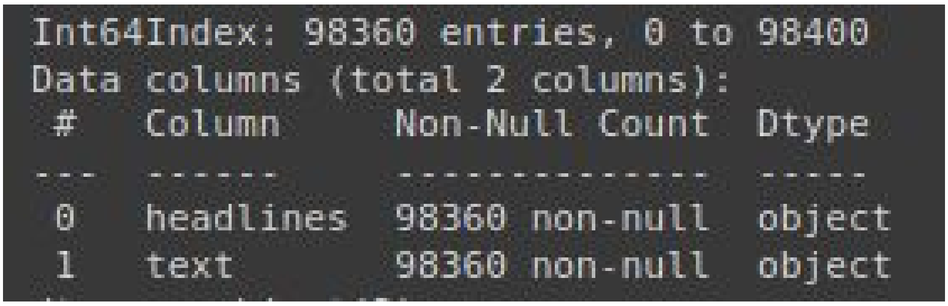
Figure - Content of field after removing stopwords

8.2.2 For Seq2Seq Model

"NEWS SUMMARY" dataset

The dataset consists of two columns ‘headlines’ and ‘text’. The data contains summarized news from Inshorts and only scraped the news articles from Hindu, Indian times and Guardian.

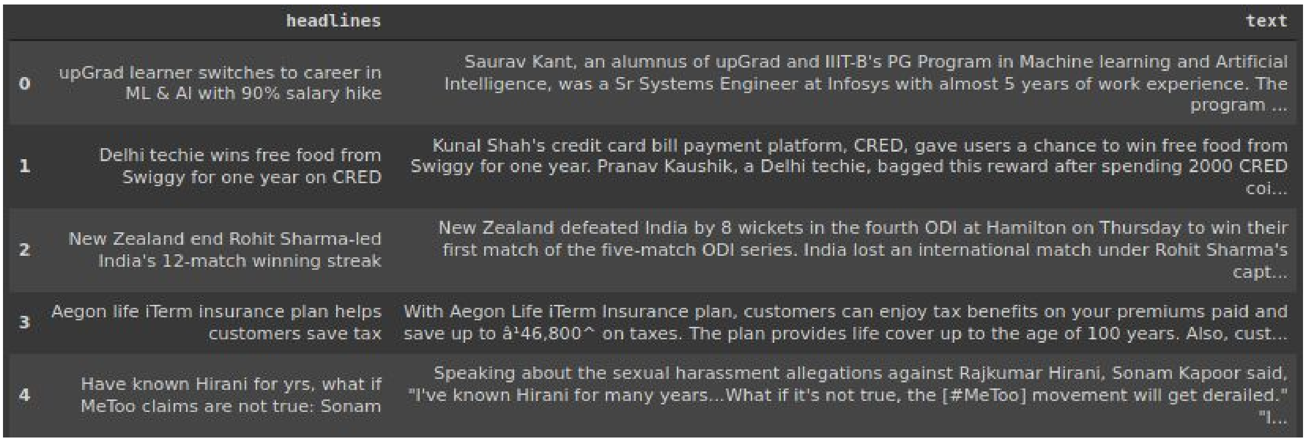
After removing rows with null values, the total number of rows are -



```
Int64Index: 98360 entries, 0 to 98400
Data columns (total 2 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   headlines  98360 non-null  object
 1   text       98360 non-null  object
```

Figure - No. of processable rows in “NEWS SUMMARY” dataset

First five rows -



	headlines	text
0	upGrad learner switches to career in ML & Al with 90% salary hike	Saurav Kant, an alumnus of upGrad and IIIT-B's PG Program in Machine learning and Artificial Intelligence, was a Sr Systems Engineer at Infosys with almost 5 years of work experience. The program ...
1	Delhi techie wins free food from Swiggy for one year on CRED	Kunal Shah's credit card bill payment platform, CRED, gave users a chance to win free food from Swiggy for one year. Pranav Kaushik, a Delhi techie, bagged this reward after spending 2000 CRED coi...
2	New Zealand end Rohit Sharma-led India's 12-match winning streak	New Zealand defeated India by 8 wickets in the fourth ODI at Hamilton on Thursday to win their first match of the five-match ODI series. India lost an international match under Rohit Sharma's capt...
3	Aegon life iTerm insurance plan helps customers save tax	With Aegon Life iTerm Insurance plan, customers can enjoy tax benefits on your premiums paid and save up to à¹46,800^ on taxes. The plan provides life cover up to the age of 100 years. Also, cust...
4	Have known Hirani for yrs, what if MeToo claims are not true: Sonam	Speaking about the sexual harassment allegations against Rajkumar Hirani, Sonam Kapoor said, "I've known Hirani for many years...What if it's not true, the [#MeToo] movement will get derailed." "I...

Figure - Rows of NEWS SUMMARY dataset

Word count inside the two columns (‘summary’ here is the ‘headlines’ column)-

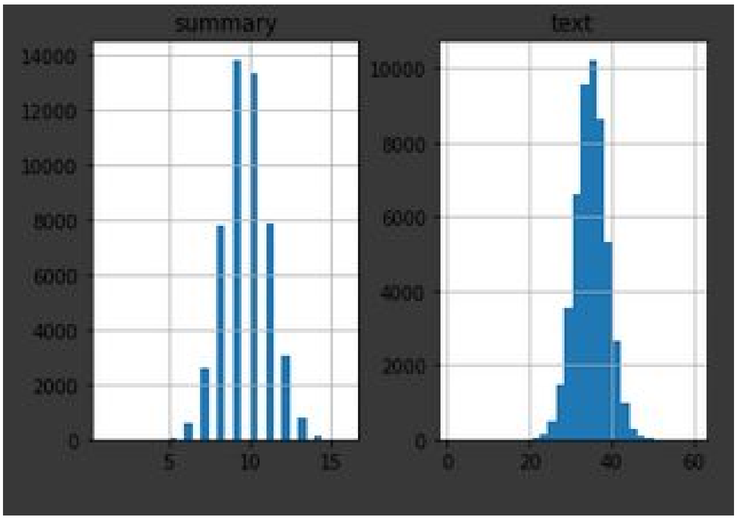
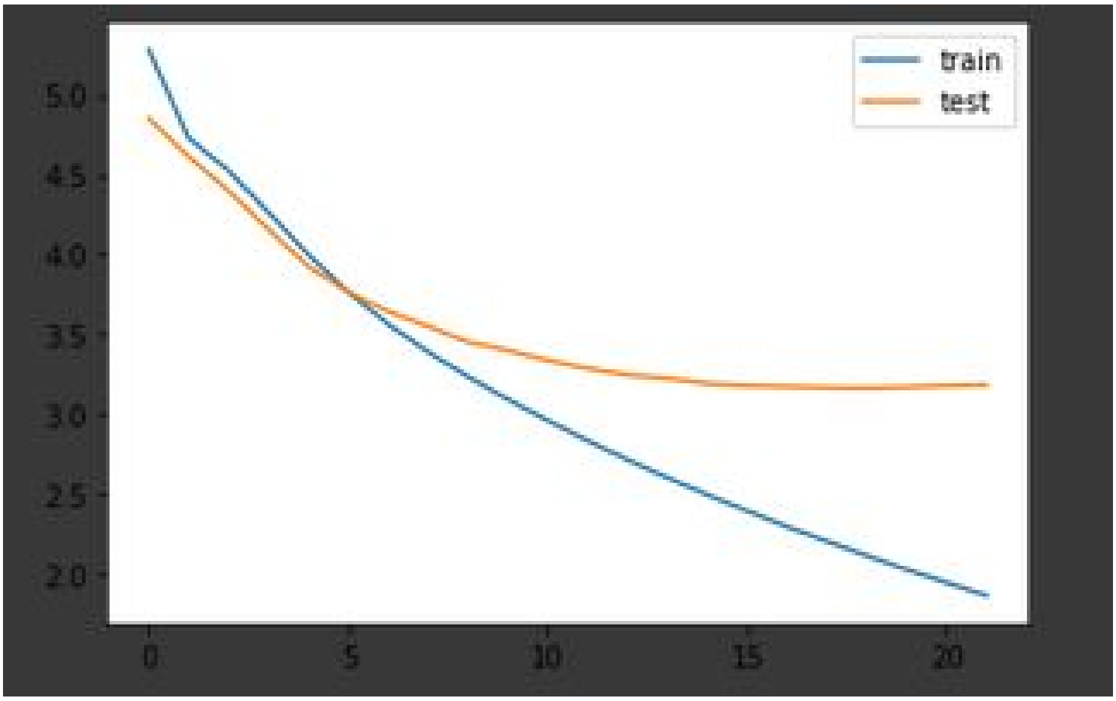


Figure - Word Count of NEWS SUMMARY dataset

We can see that for summary, most words come before word count of 15; and for text, most of them lie before 50. So, we fixed our model's prediction 'summary' word count limit to 15 and trained on the 'text' field.

Test vs Train loss after training -



'X' axis = number of epochs

'Y' axis = loss

Figure - Train vs test loss

We can see the **test** loss curve tends to go parallel after 22 epochs.

We found the dataset to be optimal for our project although processing and training this dataset does require good computational power.

The dataset was taken from kaggle.[3]

8.3 Deep Learning

Deep Learning is a subfield of machine learning concerned with algorithms inspired by the structure and function of the brain called artificial neural networks. Deep Learning has different algorithms relating to it being supervised, semi-supervised or unsupervised.

8.3.1 Recurrent Neural Networks

Neural Networks - They are a set of algorithms which closely resemble the human brain and are designed to recognize patterns. They interpret sensory data through a machine perception, labelling or clustering raw input.

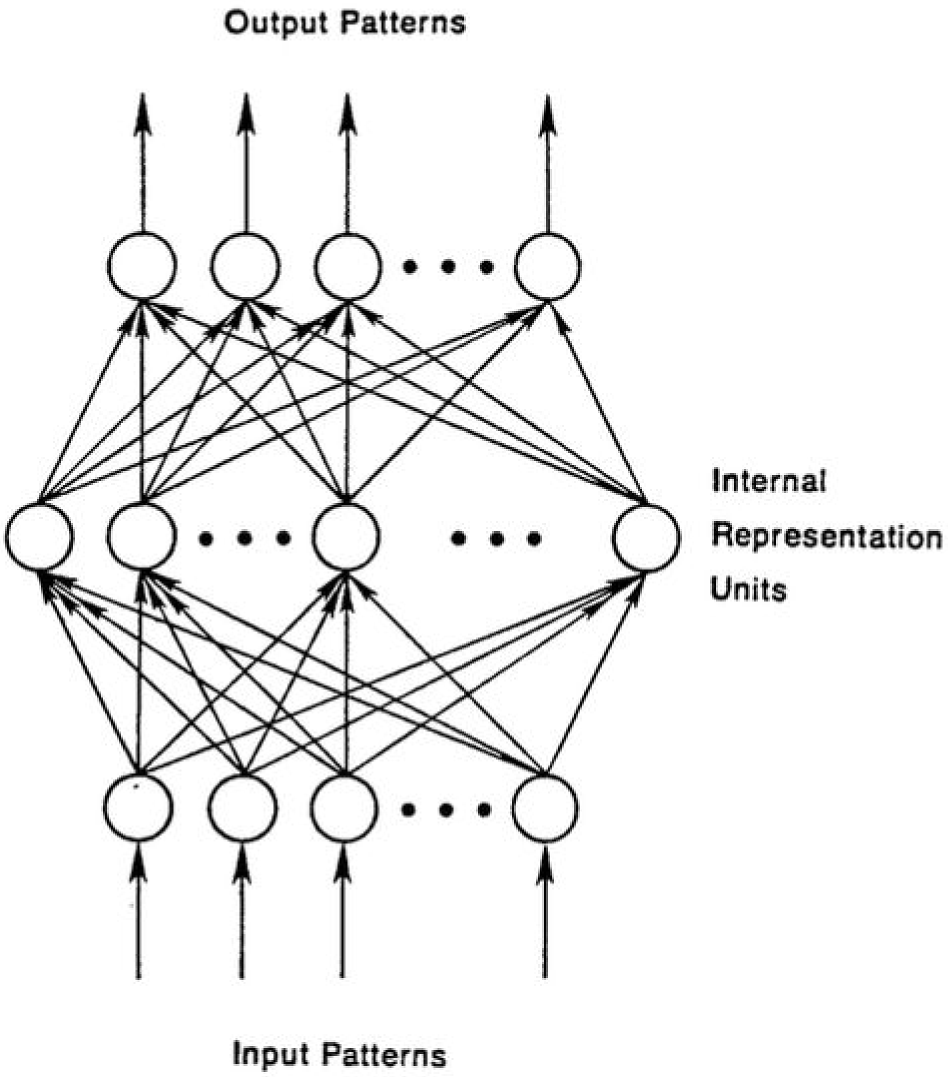


Figure - A neural network [4]

Recurrent neural networks (RNNs) is a class of artificial neural networks where connections between nodes form a directed graph along a temporal sequence. They are a class of neural networks that allow previous outputs to be used as inputs while having hidden states. That is, they have an internal memory.

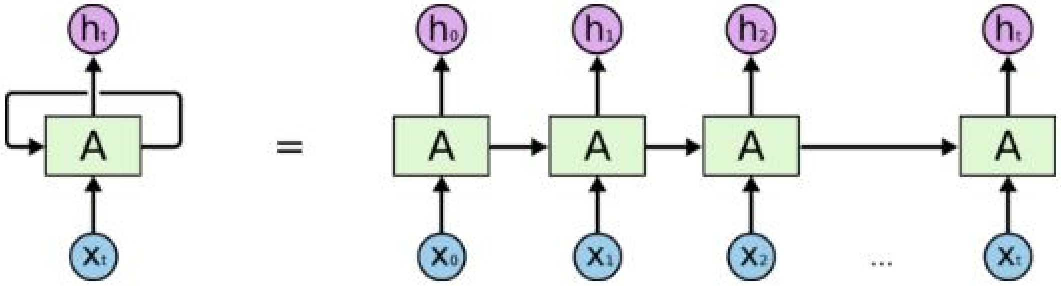


Figure - Recurrent Neural Network [5]

Here, Xi = input layers, A = hidden layers, hi = output layers.

We can see here output from the previous step is fed as input to the current step. In traditional neural networks, all the inputs and outputs are independent of each other, but in cases like when it is required to predict the next word of a sentence, the previous words are required and hence there is a need to remember the previous words. Thus RNN came into existence, which solved this issue with the help of a Hidden Layer.

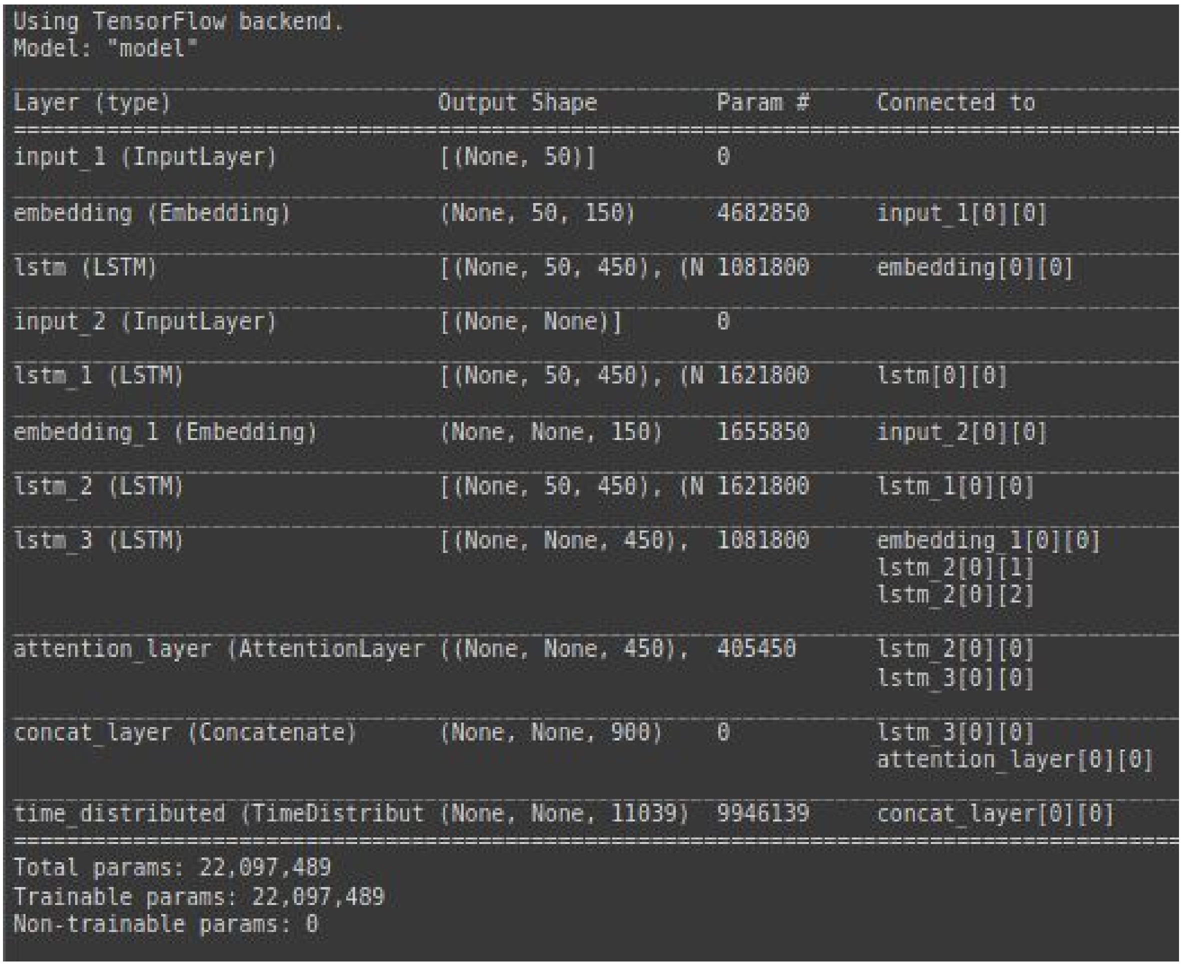
We have used supervised recurrent neural network - **LSTM**

8.3.2 LSTM -

Long Short-Term Memory (LSTM) networks are a modified version of recurrent neural networks, which makes it easier to remember past data in memory. The vanishing gradient problem of RNN is resolved here. LSTM is well-suited to classify, process and predict time series given time lags of unknown duration. It trains the model by using back-propagation.

A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell remembers values over arbitrary time intervals and the three gates regulate the flow of information into and out of the cell.

Below is the list of parameters used for encoding the deep LSTM model :



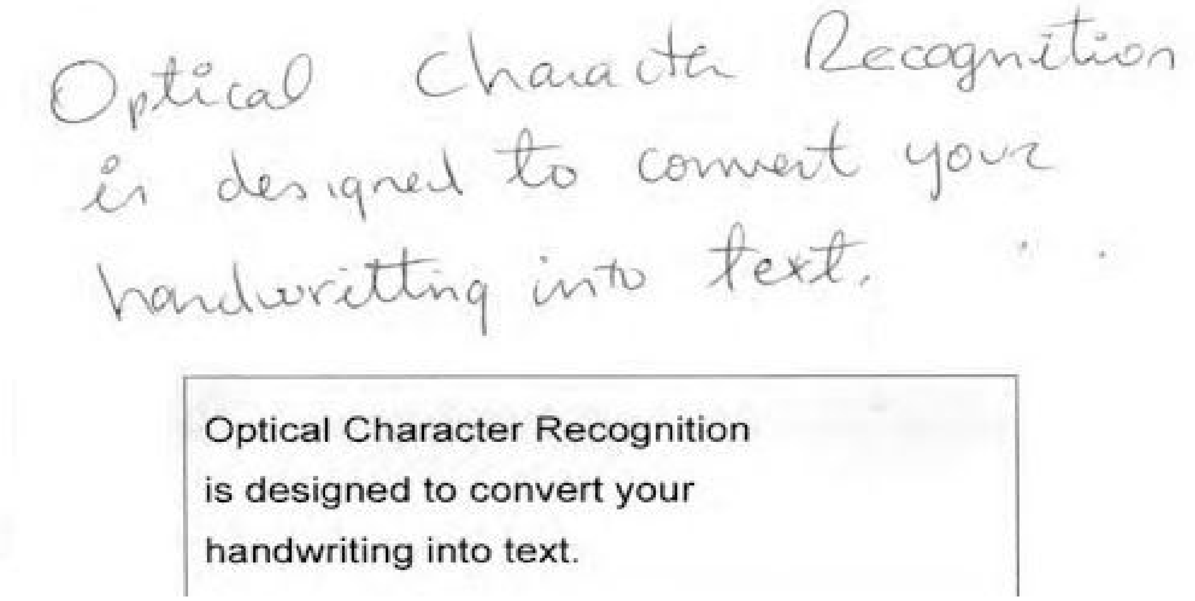
```
Using TensorFlow backend.
Model: "model"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            [(None, 50)]         0
__________________________________________________________________________________________________
embedding (Embedding)           (None, 50, 150)      4682850     input_1[0][0]
__________________________________________________________________________________________________
lstm (LSTM)                     [(None, 50, 450), (N 1081800     embedding[0][0]
__________________________________________________________________________________________________
input_2 (InputLayer)            [(None, None)]       0
__________________________________________________________________________________________________
lstm_1 (LSTM)                   [(None, 50, 450), (N 1621800     lstm[0][0]
__________________________________________________________________________________________________
embedding_1 (Embedding)         (None, None, 150)    1655850     input_2[0][0]
__________________________________________________________________________________________________
lstm_2 (LSTM)                   [(None, 50, 450), (N 1621800     lstm_1[0][0]
__________________________________________________________________________________________________
lstm_3 (LSTM)                   [(None, None, 450),  1081800     embedding_1[0][0]
                                                                 lstm_2[0][1]
                                                                 lstm_2[0][2]
__________________________________________________________________________________________________
attention_layer (AttentionLayer ((None, None, 450),  405450      lstm_2[0][0]
                                                                 lstm_3[0][0]
__________________________________________________________________________________________________
concat_layer (Concatenate)      (None, None, 900)    0           lstm_3[0][0]
                                                                 attention_layer[0][0]
__________________________________________________________________________________________________
time_distributed (TimeDistribut (None, None, 11039)  9946139     concat_layer[0][0]
==================================================================================================
Total params: 22,097,489
Trainable params: 22,097,489
Non-trainable params: 0
```

Figure - LSTM Model Summary

Multiple layers of LSTM were used to improve the model.

8.4 Optical Character Recognition

Optical character recognition or optical character reader (OCR) is the electronic or mechanical conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo.



Optical Character Recognition
is designed to convert your
handwritting into text.

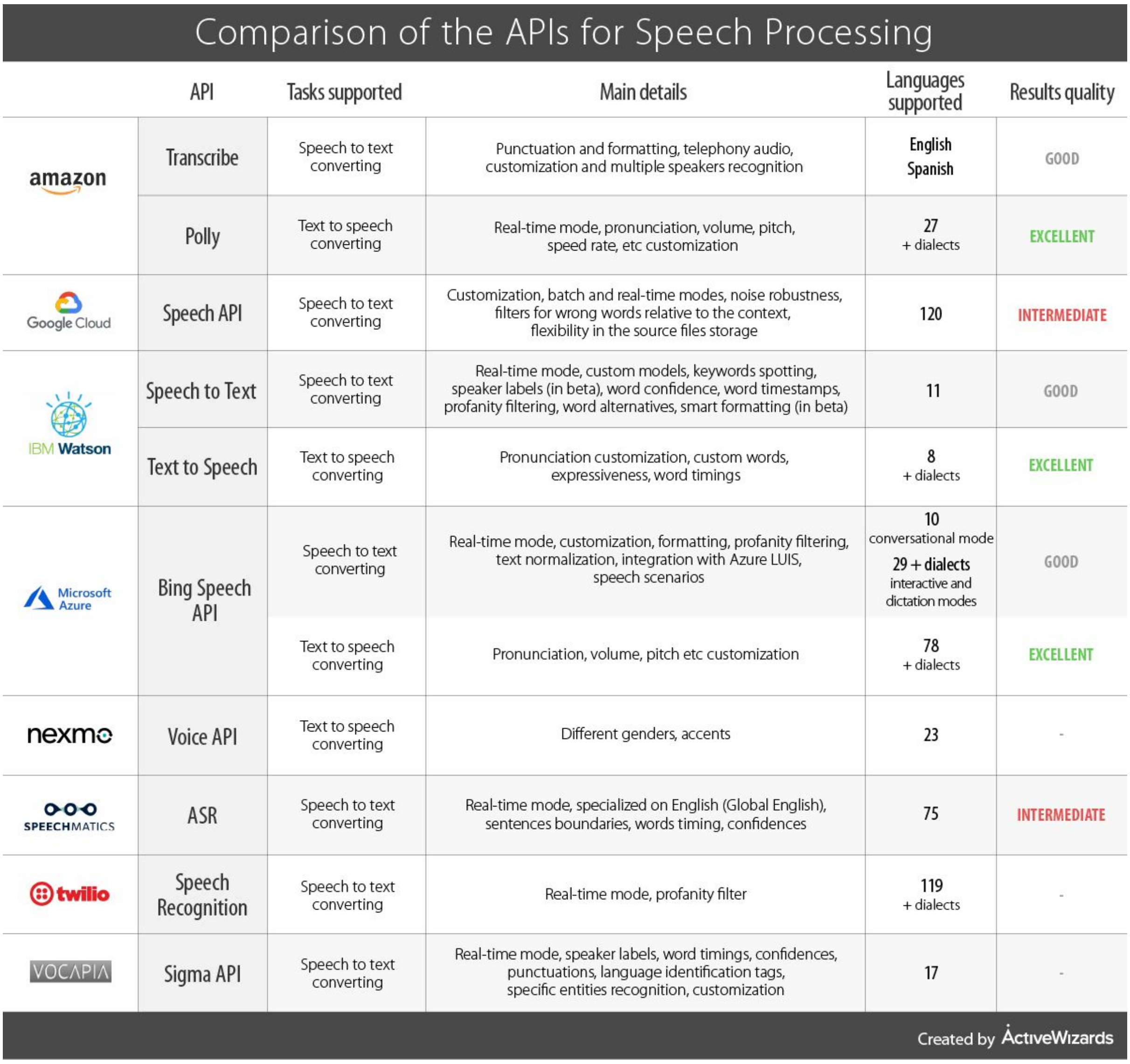
Optical Character Recognition
is designed to convert your
handwriting into text.

Figure - OCR [6]

In our project, we are using the Tesseract library (in python) to perform OCR on our extracted frames. This helps to convert data in video into readable text format.

8.5 Speech-to-text

In our project, we converted our original video file into an audio file using a cloud service. Then we used NLP to process audio to text so that we can perform operations like stemming, removing stopwords, normalization, etc.



Comparison of the APIs for Speech Processing

	API	Tasks supported	Main details	Languages supported	Results quality
amazon	Transcribe	Speech to text converting	Punctuation and formatting, telephony audio, customization and multiple speakers recognition	English Spanish	GOOD
	Polly	Text to speech converting	Real-time mode, pronunciation, volume, pitch, speed rate, etc customization	27 + dialects	EXCELLENT
Google Cloud	Speech API	Speech to text converting	Customization, batch and real-time modes, noise robustness, filters for wrong words relative to the context, flexibility in the source files storage	120	INTERMEDIATE
IBM Watson	Speech to Text	Speech to text converting	Real-time mode, custom models, keywords spotting, speaker labels (in beta), word confidence, word timestamps, profanity filtering, word alternatives, smart formatting (in beta)	11	GOOD
	Text to Speech	Text to speech converting	Pronunciation customization, custom words, expressiveness, word timings	8 + dialects	EXCELLENT
Microsoft Azure	Bing Speech API	Speech to text converting	Real-time mode, customization, formatting, profanity filtering, text normalization, integration with Azure LUIS, speech scenarios	10 conversational mode 29 + dialects interactive and dictation modes	GOOD
		Text to speech converting	Pronunciation, volume, pitch etc customization	78 + dialects	EXCELLENT
nexmo	Voice API	Text to speech converting	Different genders, accents	23	-
SPEECHMATICS	ASR	Speech to text converting	Real-time mode, specialized on English (Global English), sentences boundaries, words timing, confidences	75	INTERMEDIATE
twilio	Speech Recognition	Speech to text converting	Real-time mode, profanity filter	119 + dialects	-
VOCAPIA	Sigma API	Speech to text converting	Real-time mode, speaker labels, word timings, confidences, punctuations, language identification tags, specific entities recognition, customization	17	-

Created by ActiveWizards

Figure - Comparison of different speech to text api [7]

In our project, we have mainly used **IBM watson's** speech-to-text api which consistently performed well. We can also use **google cloud speech-to-text** as a free alternative.

8.6 Text Summarization

Text summarization refers to the technique of shortening long pieces of text. The intention is to create a coherent and fluent summary having only the main points outlined in the document. In addition to adding topics, we also provided a summary of what is happening in that timeframe.

There are two main types of how to summarize text in NLP:

1. Extraction-based summarization - a subset of words that represent the most important points is pulled from a piece of text and combined to make a summary.
2. Abstraction-based summarization - deep learning techniques are applied to paraphrase and shorten the original document. The sentences or words produced by abstraction based summarization may not be part of the source document.

We have used a deep learning LSTM model for **abstraction based summarization** in our project.

8.7 Abstractive Summarizer using Seq2Seq model

8.7.1 Seq2Seq model

Seq2Seq is a method of encoder-decoder based machine translation that maps an input of sequence to an output of sequence with a tag and attention value. The idea is to use 2 RNN that will work together with a special token and try to predict the next state sequence from the previous sequence.

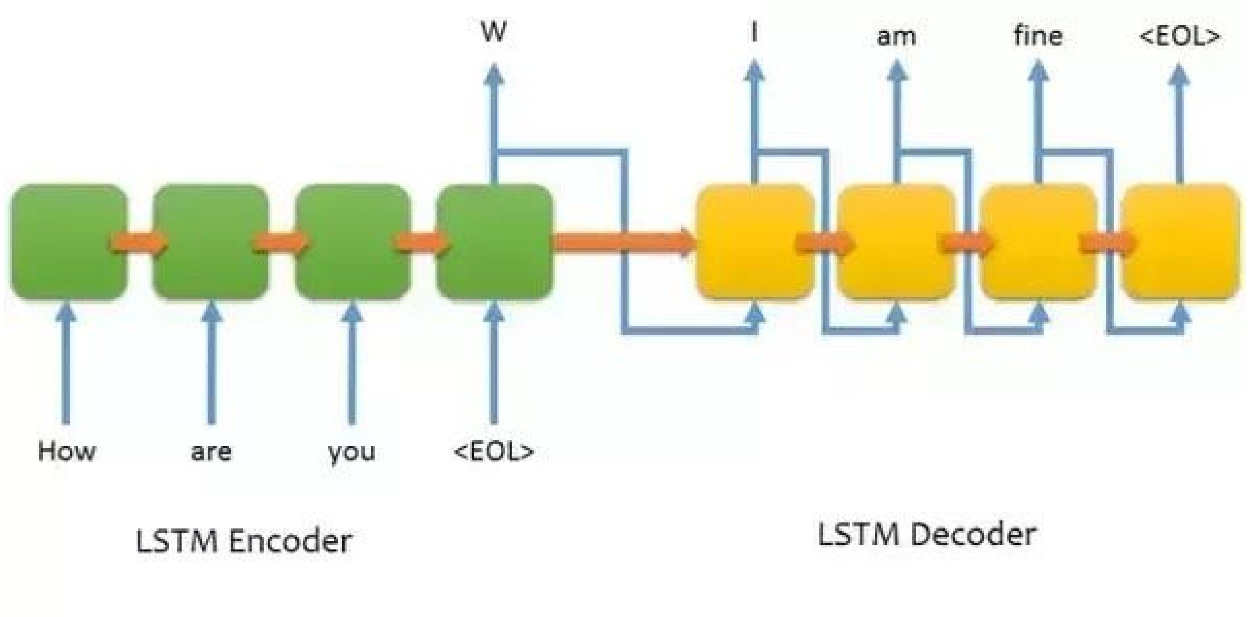


Figure - Seq2Seq model [8]

8.7.2 Attention Layer

Attention is simply a vector, often the outputs of a dense layer using softmax function. It allows the machine translator to look over all the information the original sentence holds, then generate the proper word according to the current word it works on and the context.

We have used a third party implementation of the attention layer in our project.

9. Result

9.1 Topic Modelling

The results show that the data at first was comparatively dense and became sparse over different iterations. Some of the topics identified are interpretable and some are not.

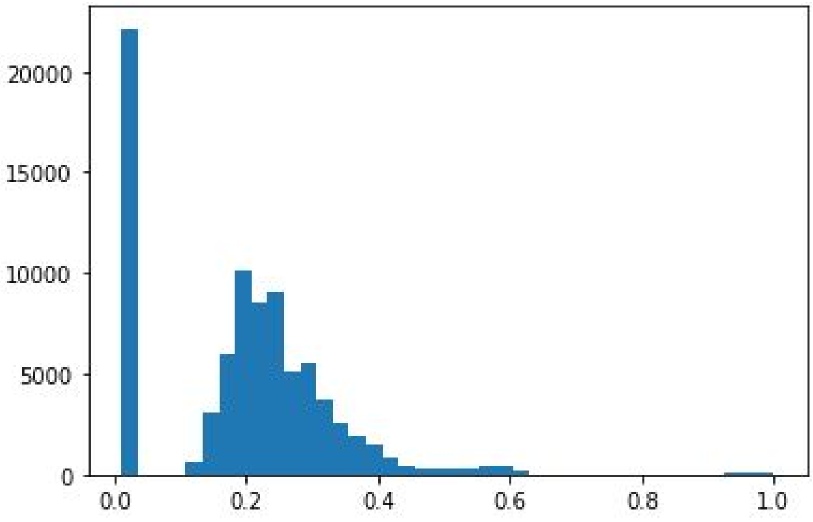


Figure - Data during starting of iterations

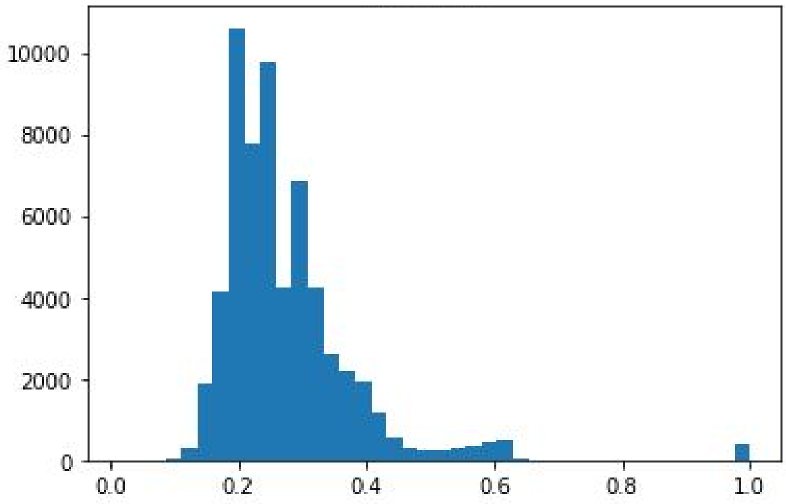


Figure - Data during mid of iterations

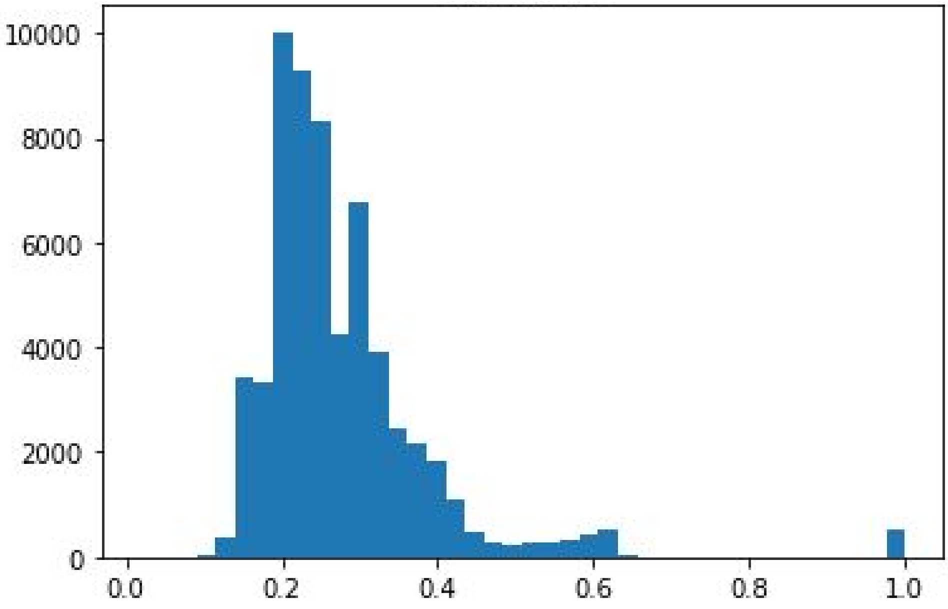
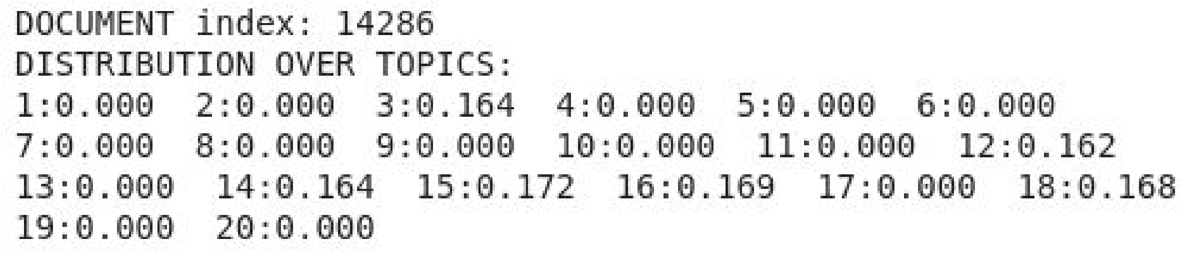
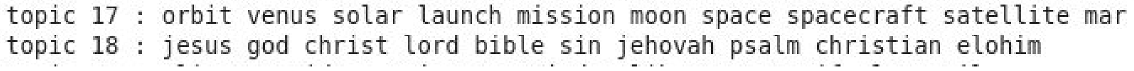


Figure - Data during ending of iterations



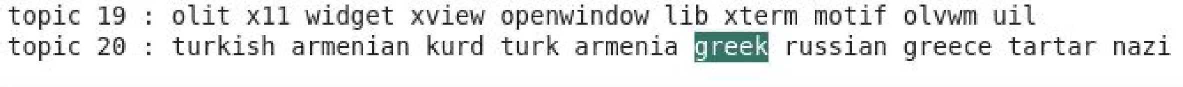
```
DOCUMENT index: 14286
DISTRIBUTION OVER TOPICS:
1:0.000  2:0.000  3:0.164  4:0.000  5:0.000  6:0.000
7:0.000  8:0.000  9:0.000  10:0.000  11:0.000  12:0.162
13:0.000  14:0.164  15:0.172  16:0.169  17:0.000  18:0.168
19:0.000  20:0.000
```

Figure - End result of particular document



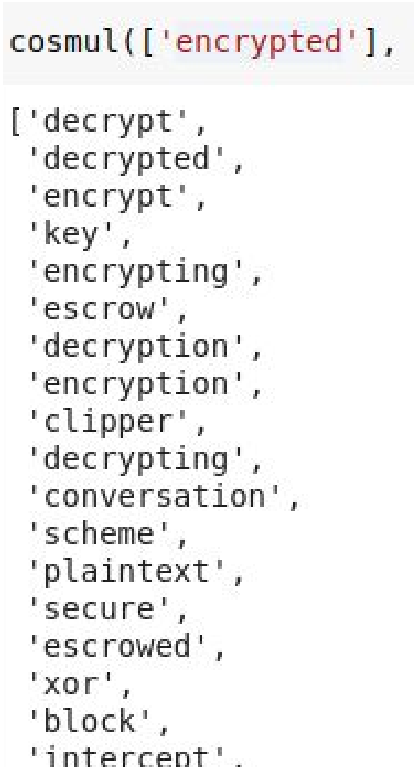
```
topic 17 : orbit venus solar launch mission moon space spacecraft satellite mar
topic 18 : jesus god christ lord bible sin jehovah psalm christian elohim
```

Figure - Some topics that are easy to understand



```
topic 19 : olit x11 widget xview openwindow lib xterm motif olvwm uil
topic 20 : turkish armenian kurd turk armenia greek russian greece tartar nazi
```

Figure - Some topics that are incomprehensible



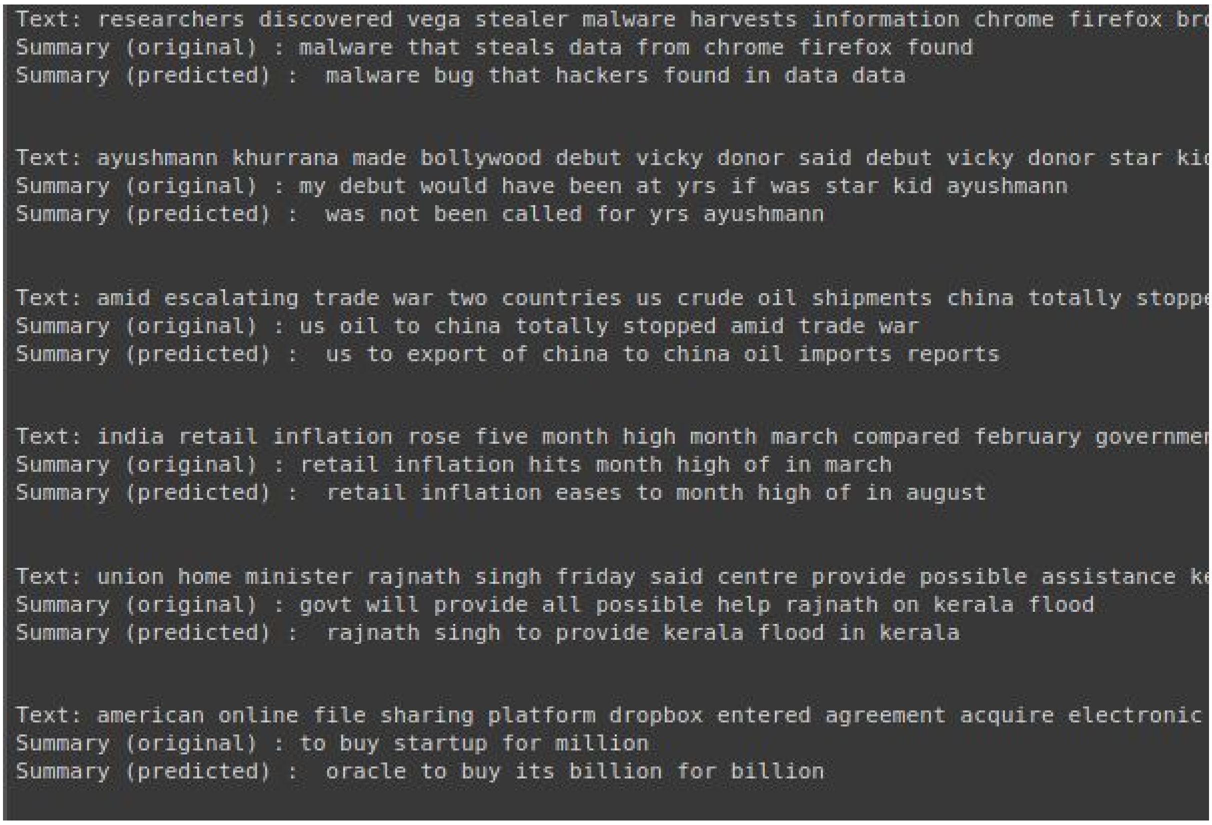
```
cosmul(['encrypted'],

['decrypt',
 'decrypted',
 'encrypt',
 'key',
 'encrypting',
 'escrow',
 'decryption',
 'encryption',
 'clipper',
 'decrypting',
 'conversation',
 'scheme',
 'plaintext',
 'secure',
 'escrowed',
 'xor',
 'block',
 'intercept',
```

Figure - Words related to encrypted across different documents

The understandable topics generally match the overall topics that are present in the documents. But the very small number of such topics pose a problem. Getting a larger will result in better understanding as it is more likely to cover topics understandable by humans.

9.2 Extractive Summarization (Seq2Seq model)

Final predicted results looks as follows -



```
Text: researchers discovered vega stealer malware harvests information chrome firefox bro
Summary (original) : malware that steals data from chrome firefox found
Summary (predicted) :  malware bug that hackers found in data data


Text: ayushmann khurrana made bollywood debut vicky donor said debut vicky donor star ki
Summary (original) : my debut would have been at yrs if was star kid ayushmann
Summary (predicted) :  was not been called for yrs ayushmann


Text: amid escalating trade war two countries us crude oil shipments china totally stoppe
Summary (original) : us oil to china totally stopped amid trade war
Summary (predicted) :  us to export of china to china oil imports reports


Text: india retail inflation rose five month high month march compared february governme
Summary (original) : retail inflation hits month high of in march
Summary (predicted) :  retail inflation eases to month high of in august


Text: union home minister rajnath singh friday said centre provide possible assistance ke
Summary (original) : govt will provide all possible help rajnath on kerala flood
Summary (predicted) :  rajnath singh to provide kerala flood in kerala


Text: american online file sharing platform dropbox entered agreement acquire electronic
Summary (original) : to buy startup for million
Summary (predicted) :  oracle to buy its billion for billion
```

Figure - Predicted result

Predicted results have some inconsistencies. This is mainly because the dataset is not quite big. It had only 98360 processable rows, which is not quite big for deep learning algorithms, but is good enough to make a decent model.

10. Limitations

10.1 Text Modelling

- **Not using bigger dataset** - The main factor for not using a bigger dataset is that it would take expensive hardware to process the data in considerable time.
- **Hardware** - We did not have access to a very expensive GPU for faster training.
- **Result Topics** - The topics that we get in result are not all easily understood by humans. With a bigger dataset this problem can somewhat be reduced.

10.2 Seq2Seq Model

There are few limitations faced by us when modelling seq2seq model -

- **Not using a huge dataset** - The main factor for not using a bigger dataset is that it would take a very expensive hardware to process the data and the model would run for multiple hours if not days. Using a much bigger dataset would help generate a better summary.
- **Hardware** - We did not have access to a very expensive GPU for faster training. Also, we used a batch size of 1024 which used all the VRAM available in the GPU. Therefore, we would require more VRAM for faster training
- **Hyperparameters** - If we decide to feed more data into the training set, we might have to change some hyperparameters to get optimal results. This may include changing optimizer, learning rate, epoch or adding dropout in LSTM layers.

11. Scope for Future Work

The project was completed in modules, with each module having significant space for growth.

- There have been developments made in developing new technologies for topic modeling. Also there are many technologies available which are different from the one used. This makes it very easy to change the algorithm used and get different results. There is a need for understanding which other technology or group of technologies can be used to get more useful results.
- As for other improvements, we can use a bigger dataset (for seq2seq model) to get better summaries. One such dataset is -**"BIGPATENT: A Large-Scale Dataset for Abstractive and Coherent Summarization"**.

  BIGPATENT, consists of 1.3 million records of U.S. patent documents along with human written abstractive summaries. Each US patent application is filed under a Cooperative Patent Classification (CPC) code. There are nine such classification categories: A (Human Necessities), B (Performing Operations; Transporting), C (Chemistry; Metallurgy), D (Textiles; Paper), E (Fixed Constructions), F (Mechanical Engineering; Lightning; Heating; Weapons; Blasting), G (Physics), H (Electricity), and Y (General tagging of new or cross-sectional technology).

  The dataset [9] is of around 6GB in size. This dataset would give much better results than any other dataset but would require an extremely high end hardware (CPU, GPU and RAM) for deep learning. Even then the model might take days to train on the data.

12. References

[1] https://themathbehindyou.wordpress.com/2019/01/22/using-cudnn-in-kera

[2] https://developer.nvidia.com/discover/lstm

[3] https://www.kaggle.com/sunnysai12345/news-summary

[4] https://pathmind.com/wiki

[5] https://towardsdatascience.com/understanding-rnn-and-lstm-f7cdf6dfc14e

[6] https://www.rsipvision.com/deep-learning-for-ocr

[7] https://medium.com/activewizards-machine-learning-company/comparison-of-top-10-speech-processing-apis-2293de1d337f

[8] https://www.ideas-engineering.io/blog/2020/2/headliner-easy-training-and-deployment-of-seq2seq-models

[9] https://arxiv.org/abs/1906.03741

[10] https://arxiv.org/abs/1301.3781

[11] https://medium.com/nanonets/topic-modeling-with-lsa-psla-lda-and-lda2vec-555ff65b0b05

[12] https://www.youtube.com/

[13] https://www.kdnuggets.com/2018/08/topic-modeling-lsa-plsa-lda-lda2vec.html

[14] https://medium.com/wisio/a-gentle-introduction-to-doc2vec-db3e8c0cce5e

[15] https://www.geeksforgeeks.org/seq2seq-model-in-machine-learning

[16] https://monkeylearn.com/blog/introduction-to-topic-modeling