STORY TITLING USING NLP

A Project Report in partial fulfilment of the degree

Bachelor of Technology

in

Computer Science & Engineering / Electronics & Communication Engineering

By

19K41A0532 B. Anu

19K41A0447 M. Nagasri

Under the guidance of

Mr. D. RAMESH

Assistant Professor School of CS & AI SRU

Submitted to

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

S.R.ENGINEERING COLLEGE (A), ANANTHASAGAR, WARANGAL

(Affiliated to JNTUH, Accredited by NBA) May-2022.

SR Engineering College

Innovation . Creativity . Entrepreneurship

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the Project Report entitled "STORY TITLING USING NLP" is a record of bonafide work carried out by the student(s) B.Anu, M.Nagasri bearing Roll No(s) 19K41A0532, 19K41A0447 during the academic year 2022-23 in partial fulfillment of the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering / Electronics & Communication Engineering** by the S.R. ENGINEERING COLLEGE, Ananthasagar, Warangal.

Supervisor **Head of the Department**

External Examiner

ABSTRACT

Natural Language Processing (NLP) is one of the most popular fields of Artificial Intelligence and its applications are diverse. NLP is often used for textual segregation activities such as spam detection and emotional analysis, text production, language translation, and text classification. Text data can be viewed in alphabetical order, word order, or sentence sequence. In general, text data is considered a sequence of words in most problems. This project is about generating a title for story. Automatic title generator is for generating title for any story in English language. The model takes story as input and produces the appropriate title after applying various approaches. The main idea behind the automatic title generator is to find suitable title from essential information of the story. It takes a story as input and produces the title after applying various approaches like frequency prioritizing, noun and adjective combination or idiom based title. Title-generation will be helpful to the editors of newspaper or technical writers to easily finding the central them without reading the whole article. Generating word embeddings for the story and story titles to train and test the model. Building an efficient model, using LSTM. Training the model on dataset and acquiring good accuracy.

Table of contents

1. INTRODUCTION

Artificial Intelligence is very vast field. This field is having many complex sub fields. One of the subfield is natural language processing which is difficult and complex to understand more over a challenge to implementing new ideas over it. Title means to understand a large article or any text document in two or three words. So that we can get the idea about article without reading it and after making an idea we can decide to read the whole article or not. Many times, only on the basis of tile we can decide that we should read the thing in detail or not. Almost all the expert systems based on knowledge base and inference engine. For the language processing, python I is the powerful language. Python provides a toolkit to process the language. This toolkit provides the good result. The training corpus is the most important element for language processing. AI is the challenge to build computational models and approaches of cognitive processes.

One of the significant contributions of AI has remained in Natural Language Processing (NLP), which glued together linguistic and computational techniques to assist computers in understanding human languages and facilitating human-computer interaction. Machine Translation, Chat bots or Conversational Agents, Speech Recognition, Sentiment Analysis, Text summarization, etc., fall under the active research areas in the domain of NLP. However, in the past few years, Sentiment analysis has become a demanding realm. Nowadays, Artificial Intelligence has spread its wings into Thinking Artificial Intelligence and Feeling Artificial Intelligence (Huang and Rust 2021). Figure 1 shows the sub domain of artificial intelligence. Thinking AI is de-signed to process information in order to arrive at new conclusions or decisions. The data are usually unstructured. Text mining, speech recognition, and face detection are all examples of how thinking AI can identify patterns and regularities in data. Machine learning and deep learning are some of the recent approaches to how thinking AI processes data.

At the core of any NLP task there is the important issue of natural language understanding. The process of building computer programs that understands natural language involves three major problems: the first one relates to the thought process, the second one to the representation and meaning of the linguistic input, and the third one to the world knowledge. An NLP system may begin at the word level to determine the morphological structure, nature (such as part-of-speech, meaning) etc. Of the word and then may move on to the sentence level – to determine the word order, grammar, meaning of the entire sentence, etc. and then to context and the

overall environment or domain. A given word or a sentence may have a specific meaning or connotation in a given context or domain, and may be related to many other words and/or sentences in the given context. the following seven interdependent levels, that people use to extract meaning from text or spoken languages, Phonetic or phonological level that deals with pronunciation., Morphological level that deals with the smallest parts of word, which carries a meaning, suffixes and prefixes, Lexical level that deals with the lexical meaning of words and parts of speech. Syntactic level that deals with grammar and structure of sentences. Semantic level that deals with the meaning of words and sentences, Discourse level that deals with the structure of different kinds of text using document structures and Pragmatic level that deals with the knowledge that comes from the outside world, i.e., from outside the contents of the document.

2. LITERATURE REVIEW

Based on the Research, There are several techniques that are used to generate a title for story few are:

[1] R.Jin (2002) describes a new probabilistic approach for title generation. He performs the probabilistic theories on text to extract the title. In very general terms they divide the process in two parts: finding the appropriate words and prepare a sequence. The words are extracted from documents and store in 'information source". "Information source" is like a temporary storage of analysed document. Now on the basis of these extracted words. Title can be generated. Now the order or sequence of these words is also very important step. It is like a pragmatic analysis.

[2] Paul E. Kennedy (2001) describes the non-extractive approach. The title can be generated without extracting the word from the document. In the model, the title and the document are considered as “the bag of words” Prepare a title vocabulary and a document vocabulary. Now, perform the estimation of probability of document word appear in the given document & title word appear in the corresponding title. This model consist the list of document word and tile word with assign the probabilities. They explain the EM (Estimate & Maximize) algorithm. They trained a word-pair model P (dw|tw) for 3 iterations with the corpus of 40000 transcripts of broadcast-news stories with human-assigned titles. They also build the language model. Extractive summarization is the most popular approach to generate titles.

[3] Cedric Lopez (2012) describes the survey on the titles. In title percentage of noun,

percentage of adjective, percentage of adverb etc. In all kind of document, in 90% cases noun is the title. This is the highest percentage. In their system they provide the percentage of all categories. More is the noun candidate leads to the quality improvement of the title. This is implemented for French language.

[4] Mario Barcala (2002) explains complex linguistic phenomena of proper noun reorganization. They explain the effectiveness of several methods. They show the result of several experiment perform to analyze the strategy of recognize proper noun. They propose a technique based on indexing. There are two sub parts: Proper noun trainer and Proper noun identifier. Proper noun trainer sub module use the trained dictionary to set the candidate proper nouns. It identifies the words begin with capital letter and its non-ambiguous position. These words include in the dictionary which is further used by next sub module. It also identifies sequences of capitalized words check that connectives are valid like the preposition of and definite articles. All possible segmentations of these sequences are measured. Based on the trained dictionary this is built in previous model proper noun identifier extracts the proper noun.

3. DESIGN

3.1 REQUIREMENT SPECIFICATION(S/W & H/W)

Hardware Requirements

- ✓ **System** : Intel Core i3, i5, i7 and 2GHz Minimum
- ✓ **RAM** : 4GB or above
- ✓ **Hard Disk** : 10GB or above
- ✓ **Input** : Keyboard and Mouse
- ✓ **Output** : Monitor or PC

Software Requirements

- ✓ **OS** : Windows 8 or Higher Versions
- ✓ **Platform** : Jupyter Notebook, Google Colab
- ✓ **Program Language** : Python

3.2 FLOW CHART

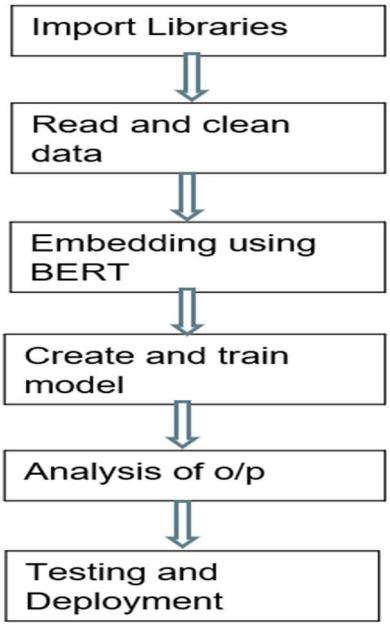

Figure 1: Flow chart

4. DATA SET

For story titling, we have created our own custom dataset. This dataset consists of 2 columns, Story and Story Title. For every story, the efficient and clear title is present in the column story title. These Short stories are collected from various story websites. This dataset consists of 100 unique stories with their corresponding tiltes. The data set was preprocessed and then it was model was trained using this data set. The test size is 0.20 and training size is 0.8 which means 20% used for testing and 80% for training.

	S.NO	STORY	STORY NAME
0	1	Once, there was a boy who became bored when he...	The Boy Who Cried Wolf
1	2	There once was a king named Midas who did a go...	The Golden Touch
2	3	One day, a fox became very hungry as he went t...	The Fox and the Grapes
3	4	Once upon a time, in a desert far away, there ...	The Proud Rose
4	5	One day, Molly the milkmaid had filled her pai...	The Milkmaid and Her Pail

Figure 2. Dataset Sample

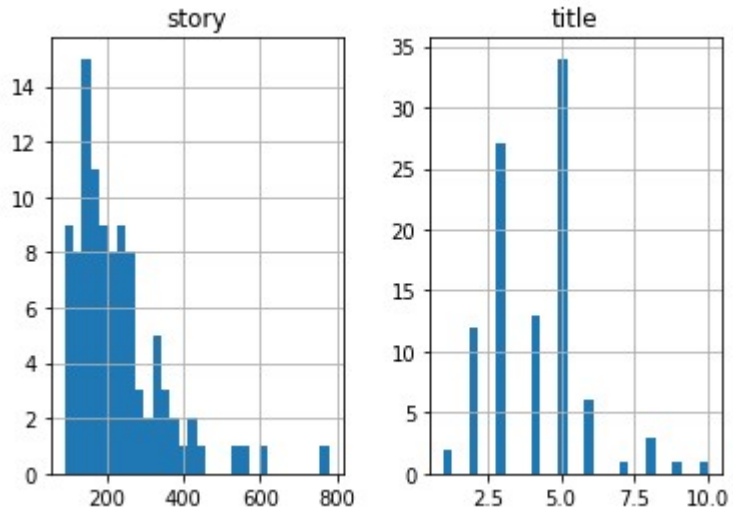

Figure.3 Word Distribution in Story and Title columns

5. DATA PRE-PROCESSING

We cannot give direct text data to the model, There will a lot of pre processing steps that need to be performed on data before giving it to the model.

```
[ ]  data = data.drop(["S.NO"],axis = 1)
```

```
[ ]  x = data['STORY']
     y = data['STORY NAME']
```

```
def text_cleaner(text):
    newString = text.lower()
    newString = re.sub(r'\([^)]*\)', '', newString)
    newString = re.sub('"','', newString)
    newString = re.sub(r"'s\b","",newString)
    newString = re.sub("[^a-zA-Z]", " ", newString)
    newString = re.sub('[m]{2,}', 'mm', newString)
    newString = re.sub("  "," ",newString)
    return newString
```

Figure.4 Data pre-processing

Steps:

- Drop unwanted columns like SNO
- Drop rows with NA values
- Drop NA may cause inconsistency in index so reset indexes
- Remove special characters in text
- Convert into lower letters
- Remove stop words

Split dataset:

Firstly split the dataset into features and target variable, then by using the train_test_ split method, split the data into a training set and test set.

The test_size = 0.20 that is 20% of data for testing and remaining 80% for training purpose.

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2)
```

Figure.5 Splitting the dataset for training and testing

6. METHODOLOGY

In our proposed model, we used BERT and LSTM deep learning models. BERT (Bidirectional Encoder Representations) is used for word embeddings. When compared to other embedding techniques like Word2Vec and GloVe, BERT word embeddings are more efficient. It converts the text to number sequences without losing any semantics. And it also works efficiently in the case of ambiguous words. LSTM (Long Short Term Memory) is a recurrent neural network, that is capable of learning long term dependencies. This model is very much useful in the case of data which consists long sentences. A story consists of long sentences, which requires a model that can store long term dependencies. So, LSTM is the best for this kind of data.

BERT (BIDIRECTIONAL ENCODER REPRESENTATIONS) :

It is a transformer-based machine learning technique for natural language processing (NLP) pre-training developed by Google. It is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide

range of NLP tasks. BERT is pre-trained on a large corpus of unlabelled text including the entire Wikipedia(that's 2,500 million words!) and Book Corpus (800 million words). It also performs brilliantly with ambiguous words, which have similar spelling but meaning of the word changes with the context. If we look at an example: We went to river bank and I need to go to bank to make a deposit. In these two sentences, the word "bank" refers to various meanings in both the sentences, so these kind of words can be dealt with BERT.

The two model sizes for BERT:

- BERT BASE – Comparable in size to the OpenAI Transformer in order to compare performance
- BERT LARGE – A ridiculously huge model which achieved the state of the art results reported in the paper

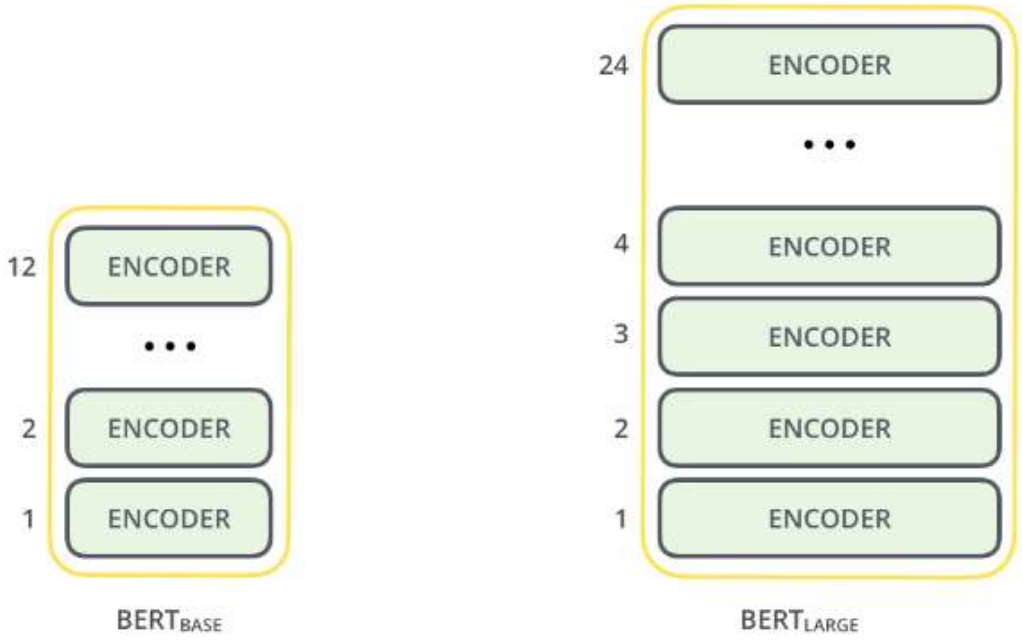

Figure.6 BERT Models based on architecture

BERT is basically a trained Transformer Encoder stack. This is a good time to direct you to read my earlier post The Illustrated Transformer which explains the Transformer model – a foundational concept for BERT and the concepts we'll discuss next.

Both BERT model sizes have a large number of encoder layers (which the paper calls Transformer Blocks) – twelve for the Base version, and twenty four for the Large version. These also have larger feedforward-networks (768 and 1024 hidden units respectively), and more attention heads (12 and 16

respectively) than the default configuration in the reference implementation of the Transformer in the initial paper (6 encoder layers, 512 hidden units, and 8 attention heads).

LSTM (LONG SHORT TERM MEMORY):

Long Short Term Memory is a kind of recurrent neural network. In RNN output from the last step is fed as input in the current step. It tackled the problem of long-term dependencies of RNN in which the RNN cannot predict the word stored in the long-term memory but can give more accurate predictions from the recent information. As the length increases RNN does not give an efficient performance. LSTM can by default retain the information for a long period of time. It is used for processing, predicting, and classifying on the basis of time-series data. LSTM model is used in applications like

- Translating the sentences from one language to other language.
- Predicting the next word, like Google search engine, before we type the whole sentence the next word will be displayed.
- Also used in applications, which are used to predict the names, places or any specific entities in a given sentence.

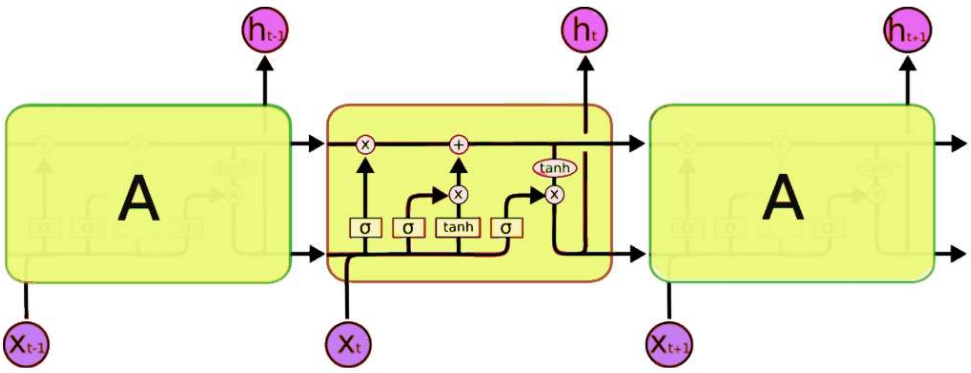

Figure.7 LSTM Model

Steps:

1. Import libraries and read excel file
2. Perform missing value treatment by dropping columns with NA values
3. Remove special Characters and convert upper case letters to lower case
4. Generate word embeddings for the story and story title using BERT.
5. Initializing model to sequential()

6. Adding embedding layer that can be used for neural networks on text data. It requires that the data to be integer encoded, so that each word is represented by unique value. It is initialized with random weights.

7. Adding 3 LSTM layers.

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 4144)]	0	[]
embedding (Embedding)	(None, 4144, 500)	14498500	['input_1[0][0]']
lstm (LSTM)	[(None, 4144, 500), (None, 500), (None, 500)]	2002000	['embedding[0][0]']
input_2 (InputLayer)	[(None, None)]	0	[]
lstm_1 (LSTM)	[(None, 4144, 500), (None, 500), (None, 500)]	2002000	['lstm[0][0]']
embedding_1 (Embedding)	(None, None, 500)	14498500	['input_2[0][0]']
lstm_2 (LSTM)	[(None, 4144, 500), (None, 500), (None, 500)]	2002000	['lstm_1[0][0]']
lstm_3 (LSTM)	[(None, None, 500), (None, 500), (None, 500)]	2002000	['embedding_1[0][0]', 'lstm_2[0][1]', 'lstm_2[0][2]']

Figure.8. Model Summary

7. RESULTS

For evaluating the performance of the proposed model training and testing accuracies are very useful. To get better accuracy the model needs to be trained using different epochs. We trained the data set using our model. We used 15 epochs to train the data.

Comparative result analysis:

```
predict_title("Finding that cranes were destroying his newly sown corn, a farmer one evening set a net in his field to catch the destructive birds. When he went
```

```
'The Farmer and the stork'
```

```
predict_title("There once was a poor boy who spent his days going door-to-door selling newspapers to pay for school
```

```
A Glass of Milk
```

```
predict_title("There was an old owl who lived in an oak tree. Every day, he observed incidents that occurred around him
```

```
A Wise Old Owl
```

```
predict_title("There once was a king named Midas who did a good deed for a Satyr. And he was then granted a wish by Dionysus,
```

```
The Golden Touch
```

```
predict_title("A mother dog and her pups lived on a farm. On the farm, there was a
```

```
The Dog At the Well
```

The above are the accurate results.

8. CONCLUSION

In this proposed system, Title generation of story using NLP is performed. This system can be used by scholars, technical writers, students and teachers. An improved way suggested to perform semantic analysis and get the main idea (theme) of the story. The model takes story as input and produces the appropriate title after applying various approaches. In this we used the BERT because it will continue revolutionizing the field of NLP because it provides an opportunity for high performance on small datasets for a large range of tasks.

REFERENCES

[1] Jin, Rong; Hauptmann, A.G.; "A new probabilistic model for title generation", In proceeding of the 19th international conference on computational linguistics-volume 1, (2002) pp.1-7

[2] Lopez, C., Prince, V., & Roche, M. "How to title electronic documents using text mining techniques", International Journal of Computer Information Systems and Industrial Management applications, vol 4, (2012) pp.562-569.

[3] Jin, Rong; & Hauptmann, A. G.; "Automatic title generation for spoken broadcast news",In

Proceedings of the first international conference on Human language technology research, (2001, March) pp. 1-3

[4] Barcala, Francisco-Mario; Miguel, A. Alonso; Jorge, Grana; "Tokenization and proper noun recognition for information retrieval", In Database and Expert Systems Applications, Proceedings. 13th International Workshop, IEEE (2002) pp. 246-250

[5] D.W.Patterson (1990) Introduction to AI & Expert Systems, Prentice Hall

[6] Mark Lutz (2009, September), Learning Python, O'Reilly Media, 4th edition.

[7]http://www.mind.ilstu.edu/curriculum/protothinker/natural_language_processing.php //accessed on 12th January 2016