News Summarization: Mandate-4

R Prasannavenkatesh
IMT2019063
IIIT Bangalore
prasannavenkatesh.ramkumar@iiitb.ac.in

M Mani Nandadeep
IMT2019051
IIIT Bangalore
mani.nandadeep@iiitb.ac.in

***Abstract*—The goal of this project is to build a text summarizer and use it to summarize the news articles. People are looking for shortcut methods to learn ideas in less time. Text summarizers are also helping people to decide whether a book, a research paper, or an article is worth reading or not. A text summarizer is a NLP based tool that wraps up text to a short length. It condenses a long article to main points. The need for text summarizers are increasing in everyday life since people want news quick and to the point rather than long and shabby walls of text. Social media sites like reddit also use a text summarizer called autotldr which is a bot that uses SMMRY to automatically summarize long reddit submissions.**

I. Seq2Seq Models for Text Summarization With Keras

Some of the key steps in building Text Summarization with keras

- **Cleaning the Data**
  - We have used a text processing library named text-hammer for cleaning the data.
  - We have also used the pipe method of spaCy to load the data as batches, this ensures that all pieces of text and summaries possess the string data type.
- **Tokenizing the Text**
  - First split the data into train and test data chunks. Use the keras text processing tokenizer with num_words = none to find out how many words are rare in the vocabulary.
  - Then tokenize again with num_words approximately total number of words minus total number of rare occurings.
- **Creating the Model**
  - define the Encoder and Decoder networks
  - The input length that the encoder accepts is equal to the maximum text length. This is then given to an Embedding Layer of dimension (total number of words captured in the text vocabulary) x (number of nodes in an embedding layer).
    This is followed by three LSTM networks wherein each layer returns the LSTM output, as well as the hidden and cell states observed at the previous time steps.
  - In the decoder, an embedding layer is defined followed by an LSTM network.
    The output of the LSTM is given to a Dense layer wrapped in a TimeDistributed layer with an attached softmax activation function.
- **Training the Model**
  - In this step, compile the model and define EarlyStopping to stop training the model once the validation loss metric has stopped decreasing.
  - The model.fit() method is used to fit the training data where the batch size is defined to be 128.
  - optimizer used: rmsprop, loss used: sparse_categorical_crossentropy
- **Generating Predictions**
  - define the encoder and decoder inference models to start making the predictions.

II. Results

Taking the number of epochs as 10 the results are satisfactory but not perfect.
If the number of epochs are around 50 the results are better.

III. OUTCOMES MAPPING TO MANDATE CONTRIBUTIONS

- CO1: **
- CO2: ***
- CO3: ****
- CO4: ***
- CO5: ****
- C06: ***

IV. References

- Inspiration for Encoder-Decoder Network for creating the Model
- Text Hammer Library
- Tensor Flow Docs