Report on Technical Work

Understandings from the Research Paper

Neural Networks have always been an easy way to develop Text Summarization models for both types of Summarizations. Summarization is of two types 1) Extractive Summarization 2) Abstractive Summarization. Extractive Summarization can be defined as the type of summarization where we extract the important statements from the text and Abstractive Summarization can be defines as the type of summarization where we rephrase the sentences such that the summary of the text remains the same but in lesser no of words. But with the use of Neural Networks there are always prone to either repeating themselves or produce factual irrelevant details from the text. So, to the baseline Sequence to Sequnce model we build an architecture that helps in improving the Neural Network. We use the Pointer generator + Coverage mechanism to improve the Neural Network.

Seq2Seq Attention Model: -

The model architecture is as follows "we convert the text to token and first feed it to the encoder in the model and produce the encoder hidden states from the encoder at the time stamp t then previous embedded token is feed into the decoder part of the architecture and produce the decoder hidden states. Now we use the = tanh ((weights in encoder) *(encoder hidden states) + (weights in decoder) *(decoder hidden states) + bias).

Then apply the SoftMax on this equation to get the attention distribution. Next, we produce the context vector using the encoder hidden states and the attention distribution. Then send the context vector to the two linear layers and the second layer has the SoftMax activation function that leads to building the Vocabulary distribution.

This architecture leads to the easy summarization of the text but it stills lead to 3 major problems. As discussed earlier it tends to be more repetitive, it does tend to produce factually incorrect details but also doesn't handle the OOV (Out of Vocabulary) words (Ex: - Mostly the Noun Words).

Pointer Generator + Coverage: -

The architecture which we use to improve the neural networks is Pointer generator + coverage mechanism. Here we use two sub mechanism one is the pointer and the second one is the Coverage mechanism. We use the pointer to make sure that every word in the given text is copied without missing and Coverage mechanism is used to keep track of the words that are being copied so that when the Neural Network is trained it does not loose the context of the text that is being given so that the proper attention is given to every word. The pointer generator helps in managing the OOV words and thus reduces the probability of generating incorrect details and the coverage mechanism gives proper attention to the words that leads to the reduction of the repetition of words.

$P(w) = P(gen)*P(vocab)(w)+ (1- P(gen)) *\sum i:wi=w*(attention\ distribution)$.

According to this if there is a OOV word then P(vocab)(w) is zero. Then we give attention to its attention distribution using the word without missing.

DEPLOYMENT OF MODELS

We build two models 1) Sequence to Sequence Attention model 2) BERT – Extractive Summarizer.

DATASET: -

The CNN / Daily Mail Dataset is an English-language dataset containing just over 300k unique news articles as written by journalists at CNN and the Daily Mail. The current version supports both extractive and abstractive summarization, though the original version was created for machine reading and comprehension and abstractive question answering, we use the version 3.0.0 for the training of the model.

EVALUATION METRIC: -

We used the ROUGE (Recall-Oriented Understudy for Gisting Evaluation) Score for the metric evaluation. ROUGE is a set of metrics used to evaluate the quality of summaries by comparing them to reference (human-generated) summaries. It is commonly employed in summarization to assess the effectiveness of automatic summarization systems. ROUGE measures various aspects of overlap and matching between the system-generated summary and the reference summaries.

ROUGE scores serve as benchmarks in research and competitions, offering a standardized measure for comparing the effectiveness of various summarization algorithms and approaches.

SEQUENCE TO SEQUENCE ATTENTION MODEL

The code begins by importing essential libraries, including TensorFlow, Keras, NLTK, and others, laying the foundation for building an advanced abstractive summarization model. Moving on to dataset management, the Kaggle API facilitates the seamless download of training, validation, and test datasets in CSV format. Pandas is then employed to read the data, with a strategic downsizing of the training dataset to enhance computational efficiency.

To ensure data quality and relevance, the code intelligently expands contractions in the text and performs comprehensive preprocessing. This involves tasks such as eliminating URLs, special characters, and, optionally, stopwords. Tokenization, a crucial step in natural language processing, is executed on both the input articles and target summaries using the Keras Tokenizer. A predefined threshold filters out infrequent words, and subsequent padding ensures uniformity in input sequence lengths.

Harnessing the power of pre-trained GloVe word embeddings, the code initializes embedding layers for both the encoder and decoder. This not only optimizes model performance but also showcases efficiency in loading and creating embedding matrices for the input and target languages.

The model architecture, a pivotal aspect, features a bidirectional LSTM-based encoder and a unidirectional LSTM-based decoder. Attention mechanisms are seamlessly integrated within the LSTM layers, elevating the model's ability to discern and capture essential information. The use of a TimeDistributed Dense layer for the decoder output underscores the sophistication of the architecture. The model's optimization is achieved through the compilation process, employing categorical cross-entropy loss and the RMSprop optimizer.

In the training phase, the fit method takes center stage, with early stopping based on validation loss providing a mechanism for efficient convergence. The model is fed with training and validation data using meticulously crafted source and target sequences.

Post-training, the code takes a forward-looking approach by saving both encoder and decoder models, alongside tokenizers, for future use. The capability for text generation is encapsulated in the

generate_summary function, demonstrating the model's practical application in generating succinct summaries for input texts.

Evaluation is a critical component, and the code leverages the ROUGE package for an automated assessment of generated summaries. ROUGE-1, ROUGE-2, and ROUGE-L scores are meticulously calculated and printed, providing a comprehensive measure of the model's summarization effectiveness.

BERT – EXTRACTIVE SUMMARIZER

The code employs the `bert-extractive-summarizer` library to conduct extractive summarization on the test dataset extracted from the CNN/DailyMail dataset. This approach utilizes a pre-trained BERT model to generate summaries, and the quality of these summaries is evaluated using the ROUGE metric, specifically ROUGE-1, ROUGE-2, and ROUGE-L. The decision to exclusively focus on the test dataset is rationalized by the utilization of a pre-trained BERT model, eliminating the necessity for additional training or validation datasets.

The BERT-based summarization model is instantiated using the `Summarizer` class, and for each article in the test dataset, the model produces a summary by extracting key sentences. The minimum length constraint is set to 20 characters. The resulting summaries are then collected in the `Summaries` list.

To evaluate the quality of the generated summaries, the ROUGE metric is employed. This metric provides a quantitative measure by comparing the generated summaries to the ground truth highlights in the test dataset. The ROUGE-1, ROUGE-2, and ROUGE-L scores are calculated, offering a comprehensive evaluation that considers unigram, bigram, and longest common subsequence overlaps, respectively.

The ROUGE scores serve as valuable indicators of the effectiveness of the BERT-based extractive summarization model. They provide insights into how well the model captures essential information from the articles, aligning with the human-generated highlights. These evaluation metrics are essential for gauging the summarization model's performance and can guide further experimentation and tuning to enhance its effectiveness.

RESULTS: -

Model Name	**Rouge – 1 (Uni gram Score)**	**Rouge – 2 (Bi gram Score)**	**Rouge – L (Longest Common Subsequence Score)**
Sequence to Sequence Attention model	0.1318	0.0188	0.1318
BERT – Extractive Summarizer	0.1852	0.0639	0.1739

OBSERVATIONS: -

Both models exhibited strong performance on the dataset; however, a detailed comparison of their ROUGE scores revealed that the BERT Extractive Summarizer consistently outperformed the Sequence-to-Sequence Attention model across all metrics. Both models demonstrated proficiency on the Daily Mail dataset, with noticeable reductions in the loss function, indicating robust model sustainability.
Interestingly, when presented with text outside the dataset, both models struggled to generate definitive and clear summaries. The Sequence-to-Sequence model displayed a tendency towards repetition and struggled with out-of-vocabulary (OOV) words. In contrast, the BERT model adeptly extracted crucial lines from the text, showcasing its ability to address challenges faced by the Sequence-to-Sequence Attention model.

In summary, while both models excelled on the dataset, the BERT Extractive Summarizer exhibited superior performance in handling text outside the dataset and addressing challenges related to repetition and OOV words, as evidenced by its higher ROUGE scores across all metrics.