Name: Ashwini Shashikant Morde

Final Test-NLP

1. **Describe the overall use case of your summarization system.**

**Answer:**

*What is Text Summarization?*

Text summarization is the process of creating a concise, coherent, and fluent summary of a longer text document, which involves underlining the document’s key points.

There are two approaches to text summarization.

1. Extractive approaches
2. Abstractive approaches

Extractive Abstractive



Using an extractive approach, we summarize our text based on simple and traditional algorithms. For instance, we store all the key words and their frequency in the dictionary when we want to summarize our text using the frequency technique. We store the sentences that contain a high frequency word in our final summary based on their usage. This indicates that the words in our summary attest to their inclusion in the original text.

With Abstractive approach, we do not produce the summary by selecting sentences from the original text passage; instead, we create a paraphrase of the key points of a given text, using a set of words that differs from the original.

*Please find below use case of my summarization system:*

1. Detailed news may include several paragraphs and over 1000 words. It takes around several mins to read through the whole news. It is hard for people to digest a huge amount of local and international news that covers lots of topics such as financial, sports, etc. By using Summarization system, we will be able to generate a legible summary based on the context present in the text which will assist people to have a high-level understanding of it quickly. Instead of spending 5 mins reading news that may not be relevant to ourselves, we may just 30 seconds getting the rough idea from the news summary.
2. We can use this system on any other dataset or article to extract key information and to produce insights. For example, we can use this system on news article to get trends and news spotlights and generate news feed content. We can also use it on food review dataset to get concise summary of the reviews.
3. Assist the processing of documents to improve efficiency. Distill critical information from lengthy documents, reports, and other text forms and highlight key sentences in documents.
4. This summarization refers to compressing all the news data and turning it into abridged summaries that you can present in a meeting or use in other business processes. Since our system can get insight from text data, this makes it a perfect tool for keeping track of local or international news.
5. **Describe the data that you are using (please include the statistics about the data and provide an example).**

**Answer:**

The dataset can be downloaded from:

https://lil.nlp.cornell.edu/newsroom/explore/index.html

We are using CORNELL NEWSROOM for training and evaluating summarization system. It contains 1.3 million articles and summaries written by authors and editors in the newsrooms of 38 major publications. The summaries are obtained from search and social metadata between 1998 and 2017 and use a variety of summarization strategies combining extraction and abstraction.

This dataset has three strategy types assigned to each summary: extractive, mixed or abstractive. The dataset we are going to look at is filtered for extractive article-summary pairs only and truncated this selection to 5,000 samples.

The database can be obtained on request after agreeing to certain copyright conditions from Cornell.

Dataset features includes:

* text: Input news text.
* summary: Summary for the news. And additional features:
* title: news title.
* url: url of the news.
* date: date of the article.
* density: extractive density.
* coverage: extractive coverage.
* compression: compression ratio.
* density\_bin: low, medium, high.
* coverage\_bin: extractive, abstractive.
* compression\_bin: low, medium, high.

Let’s get a high-level overview of the length of articles and summaries as measured in sentences.

The descriptive statistics of the article and summary lengths as measured in number of sentences. The average and median article length is 33 and 2 whereas those for the summaries are 3 and 2 respectively. We can also see that at least 75% of the summaries are between 2 and 4 sentences. We also notice heavy outliers between the 99th and 100th percentile ranging from 14 to 56 sentences. and that the articles contain heavy outliers with the 95th to 100th percentile ranging from 151 to 843.

Table

Description automatically generated Graphical user interface, application

Description automatically generated

Let’s check an approximation of the % Breakdown of publications across the corpus. First find the domain name between "www." and ".com" to identify different publications. Here some publications will have different domains for different content or publications. for example, Fox would have "fox.com" as well as "foxsports.com". From below image, we see that the NY Times accounts for nearly 25% of the corpus and that the top10 account for 80% of the corpus.

Table

Description automatically generated

We can also build a visualization of the article embeddings labelled by subject domain using t-sne. Four subject domains were chosen based on common knowledge of news coverage: 'business', 'politics', 'entertainment' and 'crime'. The BERT embedding for each word was then calculated as a reference. The article embeddings were calculated as the eman of the sentence BERT embeddings and the cosine-similarity to each subject reference was computed. The article was then assigned a subject label based on which reference embedding was closest (highest cosine similarity).

Chart, scatter chart

Description automatically generated

Given the vast information loss of the representation (from n sentence embeddings to one article embedding to 2 dimensions) and the simple label assignment, we still clearly see regions of the embedding space dominated by different domains: bottom left clusters with politics, bottom right with crime, top right with entertainment and top left with business.

We might also suspect that apparent dispersion is influenced by imbalance in the sample. We can see that crime represents nearly 38% of the sample

Table

Description automatically generated

1. **Describe your summarisation algorithm in full. Include a description of all the components that are included in your algorithm.**

**Answer:**

For my implementation I am going to use TextRank Algorithm which is an extractive and unsupervised text summarization technique.

Text rank is a graph-based ranking algorithm that works on words and sentences (word embedding or sentence embedding) and uses the similarity between sentences. Basically, TextRank finds how similar each sentence is to all other sentences and one which is most similar to other sentences are ranked higher.

*Flowchart of TextRank algorithm:*

Diagram

Description automatically generated

1. The initial step would be to concatenate all the text contained in the articles.
2. In second step, split the text into individual sentences
3. Then, we will find vector representation (word embeddings) for each and every sentence
4. Calculate the similarities between sentence vectors and store in a matrix
5. The similarity matrix is then converted into a graph, with sentences as vertices and similarity scores as edges, for sentence rank calculation
6. At the end, a certain number of top-ranked sentences form the final summary.

*How to use this model for my summarization project?*

1. Load data from JSONL to Pandas dataframe.
2. Filter for extractive summaries only:

Using the summary strategy "density\_bin" filter out all other article summary pairs and keep only extractive summaries.

1. Split each article and summary into a list of sentences using spacy:

The summaries and articles are read into the list of sentences using the large English model. These are then split into sentences using spacy's.sents method.

1. Clean the lists by excluding sentences shorter than a defined minimum:

Removal of stopwords, lemmatization and stemming are purposely ignored and this step solely focuses on removing very short sentences that would not be appropriate for the summary and ensuring that there are no empty items in the sentence list. The decision to ignore the other steps follows that these words and nuances provide semantic and contextual information that may be important for the NLP task. For example, the use of *and* versus *or* can give a sentence entirely different meanings. This is particularly true when using embeddings as sophisticated as BERT.

1. Embed each sentence using BERT:

Each sentence is embedded using the Huggine Face's sentence\_transformer library and specifying the bert-base-nli-mean-tokens model.

1. Add document label to each sentence:

Sentences need to be assigned to the document in order to recreate summaries and evaluate using ROUGE. Document labels are set using the index number of the original dataframe and looped through each document sentence to form a list of document labels that correspond to each sentence.

1. Similarity Matrix Preparation:

Next step is to find similarities between the sentences, and we will use the cosine similarity approach here. first, we will create an empty similarity matrix for this task and populate it with cosine similarities of the sentences.

1. first define a zero matrix of zero matrix of dimensions (n \* n). We will initialize this matrix with cosine similarity scores of the sentences. where, n is the number of sentences.
2. We will use Cosine Similarity to compute the similarity between a pair of sentences.
3. And initialize the matrix with cosine similarity scores.
4. Convert into Graph:

Before proceeding further, let’s convert the similarity matrix sim\_mat into a graph. The nodes of this graph will represent the sentences and the edges will represent the similarity scores between the sentences.

1. Label the article sentence with the highest similarity to each summary sentence as a positive event:

The highest cosine similarity of the matrix for each summary sentence is then identified and marked as a positive event (1) while the rest are assigned zeroes.

Using a custom function, we are going to train-test split at the document level in order to ensure that article level summaries can be sensibly concatenated for Rouge evaluation. We will implement TextRank on the test data here.

1. **Describe how you would you automatically evaluate your system. Please include the disadvantages and advantages of the evaluation. Describe how you would attempt to mitigate those disadvantages.**

**Answer:**

We can consider below evaluation techniques for our system:

The standard evaluation metrics for summarization tasks is the Rouge-N family of metrices that measure the n-gram overlap between the predicted and gold summaries.

* Rouge Recall: It is defined as the number of n-gram overlaps between the summaries normalized by the number of n-grams in the gold summary. One drawback here is that very long predicted summaries could contain all the words in the gold summary but also have many words that are not.
* Rouge Precision: It captures how concise the predicted summary is by changing the denominator to the number of words in the predicted summary.
* Rouge F1: It is defined as the harmonic mean between Rouge recall and precision and, striking a balance between recall and precision, is the primary measure typically reported for summarization tasks.
* ROUGE-L: It measures longest matching sequence of words using LCS. An advantage of using LCS is that it does not require consecutive matches, but in-sequence matches that reflect sentence level word order.
* ROUGE-S: Is any pair of words in a sentence in order, allowing for arbitrary gaps. This can also be called skip-gram concurrence. Skip-bigram measures the overlap of word pairs that can have a maximum of two gaps in between words

For our system specifically, we will use Rouge-1 and will also track Rouge-L. Rouge is not standard to the SK Learn library and we will use the rouge-score package to implement metric calculations.

*Calculate TextRank Rouge score*

* Consider the same test data on which we implemented TextRank.
* For our project, the sentences are ranked in descending order of predicted probability and the top k sentences are selected. We will take k =3. That is pick top 3 sentence by textrank score.
* Then convert list of sentences to string for each predicted summary
* convert cleaned gold summary sentence lists to string for each summary
* zip each predicted / gold summary pair together and store in another tuple
* calculate rouge scores
* Store results in dictionary.

*Advantages of Rouge Evaluation:*

* ROUGE counts the overlap of word or word units between the candidate summary and reference summary with respect to the word units in the reference summary. A higher ROUGE score means a larger overlap of word units while a lower ROUGE score means a smaller overlap of word units.
* Using Rouge 1, shows the fluency of the summaries or translation. Basically, when we follow the word orderings of the reference summary, then our summary is actually more fluent. Whereas ROUGE-L doesn’t compare n-grams; instead treats each summary as a sequence of words and then looks for the longest common subsequence.
* ROUGE is case insensitive, meaning that upper case letters are treated the same way as lower-case letters.

*Disadvantages of Rouge Evaluation:*

* lack of semantic understanding: if the human-written summary includes more novel words than the original document, ROUGE will provide a poor score to extractive summaries due to a lack of semantic awareness.
* Here we are implementing extractive summarization task that is framed as a sentence ranking problem, the ROUGE metric was not originally proposed for evaluating the quality of a ranker.
* The widely used technique behind extractive summarization is to rank sentences from the original document according to how well they reflect the overall description and then create a summary by concatenating the top-ranked sentences. Thus, the right evaluation metric for the extractive summarization task should also consider the quality of the sentence ranker. Consider a human-written summary which is highly abstractive in nature. A good ranker that ranks the most informative sentences at the top may still suffer from low ROUGE scores due to fewer direct lexical overlaps between the system summary and human-written summary.
* ROUGE is not Robust to Perturbation: One of the major drawbacks of ROUGE is that it is not semantic-aware.

*How to mitigate these disadvantages:*

* As an alternative option to mitigate these disadvantages we can use gain-based evaluation metric, Sem-nCG , which is both 1) semantic-aware and 2) rewards a system-generated summary based on some groundtruth ranking of sentences from original document. Let’s see how it works: given an original document and a human-written summary for evaluation purposes, we can use several sentence embedding techniques (including InferSent, Sentence Transformer, Elmo, Google Universal Sentence Encoding and their ensemble) to prepare groundtruth ranking of sentences from original document by computing semantic similarity between each individual sentence of original document and entire human written summary. Finally, this groundtruth ranking is compared against model-inferred ranking to compute Sem-nCG score, where a higher number means a better extractive summary.
* The main idea is to ensure a fair evaluation of the extractive summarization task where the metric is both semantic-aware as well as captures the ranking quality of the extractive summarizer. Indeed, for extractive summarization, sentences in the original document are ranked based on how well they reflect the overall description, and thus, evaluating it with a rank-aware metric like Sem-nCG is more equitable.
* While evaluating extractive summaries, mitigate the limitations of the ROUGE metric by reporting additional metrics which are semantic-aware and can generate reliable gains from human references (e.g., Sem-nCG), especially when the human-references are more abstractive in nature.

1. **Describe the limitations of your system.**

**Answer:**

1. One of the limitations of summarization is that evaluation requires a database consisting of text and their human written or curated summaries. These gold summaries are difficult to obtain in the wild and research tends to focus on news articles and scientific papers where these are the most readily available. The abstract serves as the summary for scientific papers and newsrooms typically have a teaser or summary of their articles for online banners or updates.
2. Can't Always Tell What Is Important:

Another drawback of text summarization is its inability to tell what is most important about a particular article. For example, if a person wants to know more about specific examples given in the full news article, he or she would have to go back and look at the full article.

1. Doesn't Work Well With Long Documents:

One of the limitations of summary generators are the lack of support for longer documents. Machine learning algorithms haven’t been able to produce high-quality summaries for longer pieces of text because of hardware limitations. Training the models with considerably more data than the 5,000 articles used here may be a challenge.

1. ROUGE doesn't try to assess how fluent the summary: ROUGE only tries to assess the adequacy, by simply counting how many n-grams in your generated summary matches the n-grams in your reference summary. If there are multiple references, the ROUGE-1 scores are averaged. Because ROUGE is based only on content overlap, it can determine if the same general concepts are discussed between an automatic summary and a reference summary, but it cannot determine if the result is coherent or the sentences flow together in a sensible manner.
2. ROUGE is unable to differentiate summaries that are accurate and summaries that are inaccurate. Using ROUGE as a means to measure content coverage is not sufficient for the evaluation of opinion summaries.
3. May Miss Relevant Information:

A limitation of text summarization is the possibility of missing relevant information from news.

1. Language support: Here, most of the training data is in English. The trained models might not perform as well on news in other languages, because these languages are less represented in the training data.

**References:**

* <https://ieeexplore.ieee.org/document/8336568>
* <https://towardsdatascience.com/how-to-create-simple-news-summarization-from-scratch-using-python-83adc33a409c>
* <https://towardsdatascience.com/extractive-summarization-using-bert-966e912f4142>
* https://lil.nlp.cornell.edu/newsroom/explore/index.html