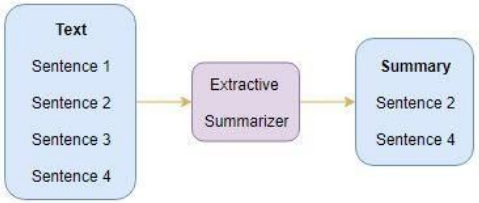
Different Summarization Algorithms tested

Two types of summarizations

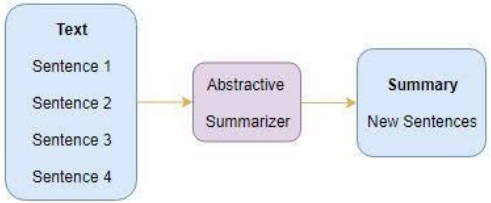
Extractive Summarization:

In Extractive Summarization, we are identifying important phrases or sentences from the original text and extract only these phrases from the text. These extracted sentences would be the summary.



Abstractive Summarization:

In the Abstractive Summarization approach, we work on generating new sentences from the original text. The abstractive method is in contrast to the approach that was described above. The sentences generated through this approach might not even be present in the original text.



Classical **Extractive** Summarization Approaches

The following methods were tested out -

- ‘SUMY’
  - LSA
  - Lex-rank
- Frequency Summarizer using NLTK.

SUMY -

‘sumy’ is a python library and can be accessed using command line to generate summaries. Summary can be generated from an html website by providing url or from a text document.

Implemented summarization methods:

- Latent Semantic Analysis
- LexRank

I have summarized all documents in which summarization percentage is 30%.

Relevant Observations (LSA):

1. Full sentences are extracted from the original document and inserted into the summary.
2. The sentences appear in the order they do in the original document.
3. Sometimes ‘.’ is considered as a full stop when it is not used in the context of a full stop.

LexRank had similar observations, just that the sentences chosen for the summary differ from LSA. Also if the original passage contains numbered list such as (a),(b),..., In LSA, the summaries contain the list markers but in LexRank they do not.

Frequency Summarizer using NLTK-

The target of the automatic text summarization is to reduce a textual document to a summary that retains the pivotal points of the original document.

This unsupervised model works by specifying how many sentences do we want in the summary.

The model then extracts that many number of sentences from the summary and places them according to a rank given by the model

The final output is not as satisfactory.

Legal Document specific **Extractive** Summarization

These models are built to specifically summarize case documents. Two of these models are supervised models and one is an unsupervised model.

The unsupervised model is ‘CaseSummarizer’ model.

The supervised models are -

- ➔ Graphical Model
- ➔ LetSum Model

CaseSummarizer Model

Observations:

1. It is an extractive summarization model.
2. Period ‘.’ Is considered as full stop even when it is not supposed to be.
3. Sentences are always included in the summary from end to end in the reference document. Sentences are never cut in half to mention only relevant information. If a sentence is chosen to be included, it will be included in its entirety.
4. According to the review paper, the model has a large execution time and the ROUGE scores are not as good as the supervised models.

Supervised Graphical Model

The model works on the basis of extracting and ranking sentences that contain important information.

- The model will first decide which sentences are important for the summary and give them a rank. Ranking is given by employing k-mixture model.

  order = ['F', 'I', 'A', 'LR', 'SS', 'SP', 'SO', 'R', 'S', 'B']

- Next, the summary will be prepared by putting the ‘F’ ranked sentence first, then the ‘I’ ranked sentence and so on. For each rank there can be more than 1 sentence. The chosen length for the summary is 34% of the length of the original document.
- ‘.’ is considered as a full stop, however it is not as bad as the CaseSummarizer model.
- The sentences do not appear in the order they appear in the document. This could tamper the meaning of the document sometimes.
- This model is much better than the second supervised model, the LetSum model as I could not make sense of the generated summaries of that model.

Abstractive Summarization based on Transformer Model

T5 model:

- The model was implemented on jupyter notebook and it works using pytorch.
- Minimum and maximum length of the summary can be stated
- The summary length produced, however, does not change much even upon altering the parameters. Essential information is picked up and paraphrased and put into the final output.
- The sentences make good sense.

Bart-large-CNN model

- The model was implemented on jupyter notebook.
- Minimum and maximum length of the summary can be stated
- The summary length produced, however, is very small and not as good as the T5 model.
- Because of the small length, the summary feels incomplete.

Conclusion

The six models which give satisfactory outputs which make sense are:

1. Latent Semantic Analysis
2. LexRank
3. CaseSummarizer model
4. Graphical Model
5. T5 model
6. Bart-large-CNN model

The outputs of all these models on six inputs can be found on this link - https://auspicious-flame-69f.notion.site/b069d7c419e14da0b86908cc838eec2b?v=07fd600e531e439d97fc16c083d887c7

Roadblocks -

Tried implementing a deep learning model on this link - https://github.com/abisee/pointer-generator

The model gives outputs based on a inputs fed as test set in form of .story or .bin files. It also needs a large dataset to train on. Since a large dataset from westlaw was inaccessible, the model could not be trained or tested. The model divides the dataset into train and test and hence, training on a non legal dataset does not work as providing input externally on a trained model is not possible. The input should be included in the test files.