Natural Language Processing (DS 5007)

Project Report

Submitted By: **Muhammad Raamish Alam(20k-1326), Muhammad Ashar Ansari (20k-1409), Syed Muzammil(20k-1394)**

Submitted To: **Sir. Raza Abbas**

Course Name: **NLP(DS 5007) - Fall22**

Dated: **Jan 4 2023**

Table of Contents:

Link To Project

https://github.com/786raamish/text-data_summarization_nlp_project2022

1 Introduction

Background:

If a book captures our interest, do we start reading it at once? Not likely. We are more inclined towards reading its "Blurb" or an "Review" to determine how the book likely is presented and if we should dedicate the necessary time to go through it or not.

Another example is when we open any online news sites, do we simply begin reading each news story? Presumably not. We commonly look at the short news summary and afterward read more details provided we are intrigued.
Short, informative outlines of the news are currently everywhere like magazines, news aggregator applications, research sites, and so on.

Indeed, it is feasible to make the synopses naturally as the news rolls in from different sources all over the planet.

The strategy for removing these outlines from the first tremendous text without losing fundamental data is called Text summarization. It is fundamental for the outline to be familiar, nonstop and portray the critical.

As a matter of fact, the Google news, the inshorts application and different other news aggregator applications exploit text synopsis calculations.

2 Literature Review

Text summarization is an emerging research field. H.P. Luhn in 1958(Luhn, 1958) presented this examination philosophy. He proposed a technique to remove the significant sentences from the text utilizing elements like expression and word recurrence (Allahyari et al, 2017).].

Interaction of Programmed Text Outline incorporates removing or assembling huge information from unique substances and displays that information as synopsis (Nitu et al, 2017). It decreases the powerful chance to get the core of the information. Need for synopsis should be visible for various reasons and in various spaces, for instance synopsis of news stories , messages, statistical surveying, data connected with government specialists, clinical history of patients and sicknesses and so on. Summarization has variations as it should be possible on a solitary record and furthermore on various archives on comparative topics.

Summarization tools are likewise accessible online in view of the sort of information to be handled for various fields like news article summarizers like Columbia News blaster (Saranya Mol and Sindhu, 2014) and clinical field related rundown devices like Total Essential (Gaikwad and Mahender, 2016).Classification of text rundown depends on different norms (Rani and Tandon, 2018). In view of these norms three rules are being set : input sort of report, reason criteria , archive yield measures (Aries et al, 2019). Early rundown was finished on the single report which produces the rundown of a solitary record (Khan and Salim, 2014). Be that as it may, as the information increases, a multi archive outline arose.

3 Methodology

Approach:

Within the bounds of NLP, Text summarization can be constituted within two major categories. **Extractive** and **Abstractive methods**.

When it comes to Extractive Text Summarization It is the conventional method developed first. The principal objective is to recognize the critical sentences of the text and add them to the summary being accumulated. We really want to take note of the summary that contains accurate sentences from the original data.

While Abstractive Text Summarization. It is a relatively advanced technique, numerous progressions continue to come out frequently. The approach is to identify the significant segments, decipher the specific situation and reproduce in another manner. This guarantees that the core data is passed on through the most limited text conceivable. Note that here, the sentences in outline are created, not simply separated from original text.

We will be relying on both approaches, and experiment to see which one produces the desired results that harmonizes with being displayed and worked on the web application that will be deployed on flask.

3.1 Motivating Experiments

Extractive Summarization:

The motivation of this system for extracting important information and presenting it to the end user as a summary. The text is pre-processed, and sentences are scored using various text features, including citations and synonyms. A neural network is used to classify which sentences should be included in the summary based on their scores. The system allows the user to specify the percentage of the original text that should be included in the summary, and it is found that using citations as a scoring feature results in the best performance.

Abstractive Summarization:

Abstractive summarization is a type of text summarization that involves generating a summary that is not just a rephrasing or subset of the original text, but rather a new and shorter version that captures the main points of the original text. One motivation for using abstractive summarization is that it allows for more flexibility and creativity in the summary, as it can include new phrases and sentences that are not present in the original text. This can make the summary more concise and easier to understand for the reader. Additionally, abstractive summarization can be more effective in summarizing texts that are long or complex, as it can capture the main ideas and concepts rather than just presenting a shortened version of the original text.

3.1.1 Datasets

Extractive Summarization:

- We use a tensor flow data set as the Opinosis Dataset like Opinosis is an abstractive text summarization dataset that aims to generate a summary that reflects the opinion or perspective of the original text. It is based on the idea that people often express their opinions and perspectives in natural language text, and that these expressions can be identified and used to generate a summary that reflects the opinion of the text. Opinosis has been applied to a variety of text types, including reviews, blogs, and news articles, and has been shown to be effective at generating concise and coherent summaries that accurately reflect the opinion of the original text.
- DUC 2002 Dataset: Document Understanding Conference (DUC) like The DUC 2002 dataset is a collection of English news articles and summaries that was used as part of the Document Understanding Conference (DUC) 2002. The dataset includes 500 news articles and their corresponding human-written summaries, which were produced by professional journalists. The summaries in the DUC 2002 dataset range in length from 1-4 sentences and are intended to capture the main points of the corresponding news article. The DUC 2002 dataset has been widely used in research on text summarization, and has served as a benchmark for evaluating the performance of automatic text summarization systems.

Abstractive Summarization:

- Dataset 1: Load a news article random sample to summarize.
- Dataset 2: Short Text Summary: Load the same 'canary deployment' sample text for a like-for-like comparison with Extractive Summary.
- Dataset 3: Long Text Summary: using “Harry Potter and the Sorcerer’s Stone” story as a test sample.

3.2 Implementation Details for Finalized Approach

We decide to implement our project for finalized approaches below:

Extractive Summarization:

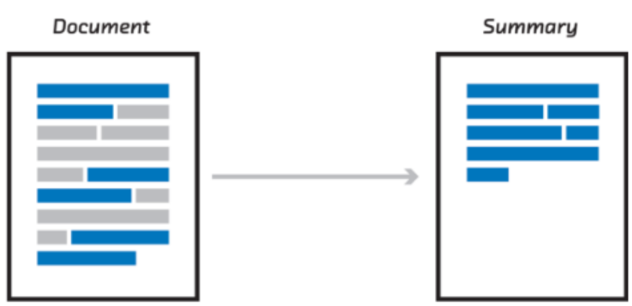
First we need to test with different extractive summarization and get the best approach from them. We will use different Summarization techniques in which include:

- Sumy.
- Gensim.

So in the above strategies we have different summarization mechanisms which we are using as per the best practices as per context/nature of text.

Sumy:

sumy is a Python library for automatic summarization of texts. It provides a simple and easy-to-use interface for extracting summaries from various kinds of texts such as HTML documents, PDF files, and plain texts. The library includes several algorithms for summarization, including ones based on the Luhn algorithm, the LexRank algorithm, and the TextRank algorithm. It also provides preprocessing tools for tokenizing and normalizing texts, as well as tools for evaluating the quality of summaries.



Gensim:

Gensim is a Python library for topic modeling and document similarity analysis. It provides a high-level interface for working with large collections of unstructured text data, such as corpora and datasets for natural language processing tasks. Gensim includes implementations of popular topic modeling algorithms such as Latent Dirichlet Allocation (LDA) and Latent Semantic Analysis (LSA), as well as tools for calculating document similarity and creating document representations. It is designed to be efficient and scalable, making it suitable for working with large volumes of text data.

Abstractive Summarization:

For Abstractive Summarization we use different techniques in which includes:

- T5-Model (Text-to-Text Transfer Model).

T5-Model (Text-to-Text Transfer model):

A text-to-text transfer model is a type of machine learning model that is trained to perform a transformation on a given input text, producing an output text as a result. These models are often used for natural language processing tasks, such as translation, summarization, and text generation.

One common approach to training a text-to-text transfer model is to use a large dataset of input-output pairs, where each pair consists of an input text and the corresponding output text that the model should produce. The model is then trained to minimize the difference between the predicted output text and the ground-truth output text in the training data.

Text-to-text transfer models can be implemented using a variety of techniques, such as sequence-to-sequence models with attention mechanisms, transformer networks, and encoder-decoder architectures. These models can be trained using supervised learning, where the training data consists of labeled input-output pairs, or unsupervised learning, where the model is trained to learn a transformation based on the structure of the input and output texts.

3.3 Competing Techniques

Extractive Summarization:

First we use Sumy LexRank Techniques in our project so bascially the LexRank is the context of summarization Techniques, LexRank works by constructing a graph of the sentences in a document, where the nodes represent sentences and the edges represent the similarity between them. The algorithm then ranks the sentences based on their centrality in the graph, with the most central sentences being selected as the summary.
It provides a simple interface for extracting summaries from various kinds of texts using the LexRank algorithm. The library includes tools for preprocessing texts, such as tokenization and normalization, as well as tools for evaluating the quality of the summaries produced.

We will import essential libraries and also add files in our project for sample data and training data purpose.

First, we use summarization techniques with providing:

- A sample text file which contains a bunch of text.
- A Simple hard coded text (In different languages including English/German).
- A HTML link for summarization (https://en.wikipedia.org/wiki/Combat).

In the above competing techniques we are using tokenize as per the nature of the text/input data.

Also we use other summarization techniques in which includes: (LSA, Luhn, Edmundson).

LSA:
The idea that the meaning of a word can be inferred from the context in which it is used, and that the relationships between words in a text can provide insights into the meaning of the text as a whole.

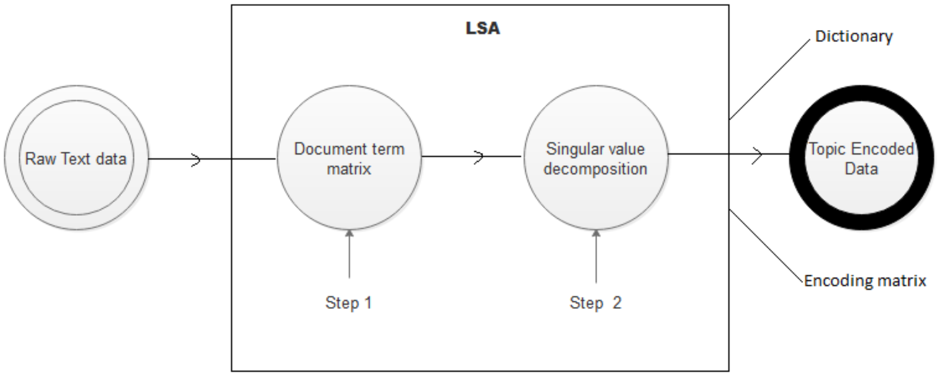
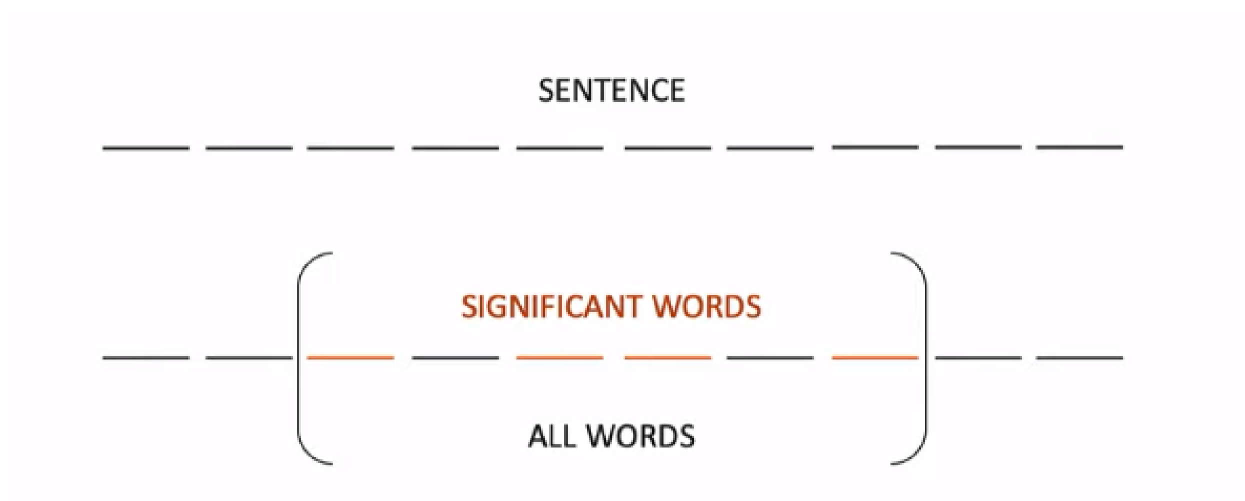


Fig. LSA processing

Luhn:

The ability of a user to determine the relevance and value of a piece of information based on the cues provided by its context. Luhn is also credited with developing the first automatic abstracting machine, which used statistical and linguistic techniques to extract important information from a text and generate a summary.



Edmundson:

This method uses the idea of defining bonus words and stigma words, words that are of high or low importance respectively.
Words in the document title are given additional importance.

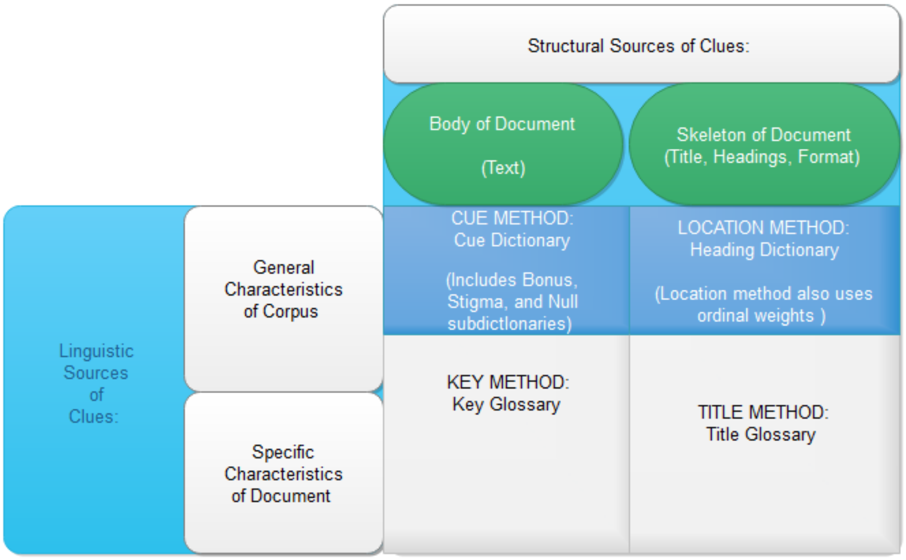
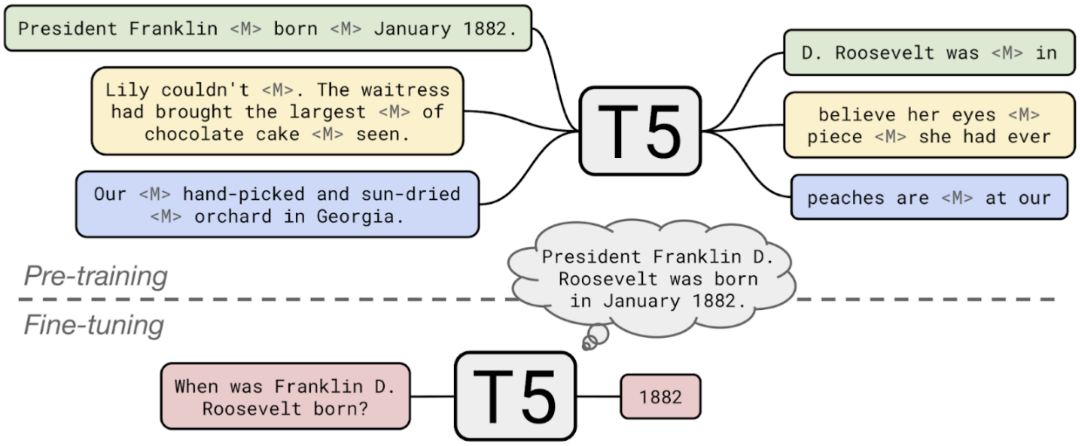


Fig. Rationale of the four basic methods

Abstractive Summarization:

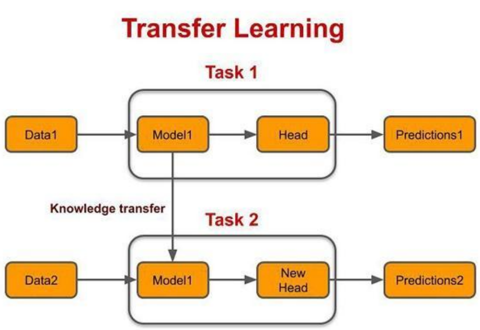
For Abstractive Summarization we use to import some essential libraries in which the transformer is one of the important for computing Abstractive summarization techniques.
it is a **deep learning - generative language model approach.**

T5 Model (Text-To-Text Transfer Transformer):
T5 Model is an encoder and decoder Pre-trained model which can be used for multi-task mixture for unsupervised and supervised problems in which the current task/problem converts into text-to-text format. So T5 model perform fine in the various NLP problems for out-of-the-box by prepending a different prefix for the input corresponding for every task.
We can use T5 Model for Translation and Summarization problems.



What is 'Transfer Learning'?

Transfer learning is a machine learning technique that involves using a pre-trained model on one task as the starting point for a model on a second, related task. The idea behind transfer learning is to take advantage of the knowledge gained from solving one problem and apply it to a different but related problem, thus reducing the amount of data and computation needed to train the second model. Transfer learning has been applied to a wide range of tasks in natural language processing, including language translation, text classification, and sentiment analysis. It has proven to be particularly effective when the second task has a limited amount of training data or when the two tasks are closely related.



3.4 Evaluation Metrics

We evaluate the original text and summarize results produced by different summerizars using different input scenarios on the basis of BLEU score and ROUGE score.

Sumy's lex rank:

For Text File:

BLEU SCORE	5.1304923275757076e-232
ROUGE SCORE	'rouge1': Score(precision=1.0, recall=0.177319587628866, f measure=0.3012259194395797), 'rougeL': Score(precision=1.0, recall=0.17731958762886(measure=0.3012259194395797)

For Text String:

BLEU SCORE	5.0454056966289525e-232
ROUGE SCORE	'rouge1': Score(precision=1.0, recall=0.4647058823529412, fmeasure=0.6345381526104418), 'rougeL': Score(precision=1.0, recall=0.464705882352941 fmeasure=0.6345381526104418)

For Web URL:

BLEU SCORE	8.444234747540162e-232
ROUGE SCORE	'rouge1': Score(precision=1.0, recall=1.0, fmeasure=1.0),'rougeL': Score(precision=1.0, recall=1.0, fmeasure=1.0)

Sumy's other summarizers (LSA, Luhn and Edmundson, etc.):

For Text File:

BLEU SCORE	5.1304923275757076e-232
ROUGE SCORE	'rouge1': Score(precision=1.0, recall=0.191752577319587 fmeasure=0.32179930795847755), 'rougeL': Score(precision=1.0, recall=0.19175257731958764, fmeasure=0.32179930795847755)

For Web URL:

BLEU SCORE	7.720899511627474e-232
ROUGE SCORE	'rouge1': Score(precision=1.0, recall=1.0, fmeasure=1.0), 'rougeL': Score(precision=1.0, recall=1.0, fmeasure=1.0)

Gensim's TextRank:

For Text File:

BLEU SCORE	5.1304923275757076e-232
ROUGE SCORE	'rouge1': Score(precision=1.0, recall=0.587628865979381 fmeasure=0.7402597402597402), 'rougeL': Score(precision=1.0, recall=0.5876288659793815, fmeasure=0.7402597402597402)

For Web URL:

BLEU SCORE	3.235229487165337e-232
ROUGE SCORE	'Rouge1': Score(precision=0.8105263157894737, recall=0.030771280138537364, fmeasure=0.059291581108829566), 'rougeL': Score(precision=0.4245614035087719, recall=0.016118289596376716, fmeasure=0.031057494866529773)

4 Results and Discussion

As we discussed earlier, like in this project, we use different techniques for summarization in both Extractive and Abstractive summarization, in which we found many reasons which make them differ from each others.

Extractive Summarization:

In Extractive Summarization we use Sumy Technique in which we apply Sumy with three techniques which are listed below.

- Use with text file using LexRank:

```
file = "/content/sample_data/plain_text_sample_sec.txt" #put the sample text file in the same folder as this notebook
#Read the original text file in Python:
with open(file) as f:
    contents = f.read()
    print("===Original Text===\n"+contents)
```

```
===Original Text===
Robotics, design, construction, and use of machines (robots) to perform tasks done traditionally by human beings. Robots are widely used in such industries as automobile manufacture to perform si
```

- Output with summarization:

```
#which languages are supported:
parser = PlaintextParser.from_file(file, Tokenizer("english"))
summarizer = LexRankSummarizer()
summary = summarizer(parser.document, 3) #Summarize the document with 3 sentences, a tunable hyperparam

print(f'\nSumy outcome type:{type(summary)}, element type:{type(summary[0])} \n{summary[:1]}')
allstr = '\n'.join(str(sentence) for sentence in summary)
print(f'\n===Extractive Summary using Sumy LexRank:===\n{allstr}')
```

```
Sumy outcome type:<class 'tuple'>, element type:<class 'sumy.models.dom._sentence.Sentence'>
(<Sentence: Robotics, design, construction, and use of machines (robots) to perform tasks done traditionally by human beings.>,)

===Extractive Summary using Sumy LexRank:===
Robotics, design, construction, and use of machines (robots) to perform tasks done traditionally by human beings.
A typical example is a humanoid robot, the building of which has been regarded as the final target of robotics for many years, especially in Japan where many robotics researchers have been strug
Pfeifer and Scheier's book from 1999 entitled Understanding Intelligence shows a constructive approach to understanding intelligence with a variety of robots from the viewpoint of embodied cogni
```

- Use with a hardCoded Text:

B. Summarize a text string

```
[8] ### Are you talking about multi-line strings? Easy, use triple quotes to start and end them.
    string = """Nearly ten years had passed since the Dursleys had woken up to find their nephew on the front step, but Privet Drive had hardly changed at all. The sun rose on the same tidy front ga

    parser = PlaintextParser.from_string(string, Tokenizer("english"))
    summary = summarizer(parser.document, 2) #Summarize the string with 2 sentences

    for sentence in summary:
        print(sentence)
```

```
Nearly ten years had passed since the Dursleys had woken up to find their nephew on the front step, but Privet Drive had hardly changed at all.
The sun rose on the same tidy front gardens and lit up the brass number four on the Dursleys' front door; it crept into their living room, which was almost exactly the same as it had been on the
```

- Use with a different Language Text:

Try another language

```
### Are you talking about multi-line strings? Easy, use triple quotes to start and end them.
string = """Près de dix ans s'étaient écoulés depuis que les Dursley s'étaient réveillés pour trouver leur neveu sur le perron, mais Privet Drive n'avait pratiquement pas changé. Le soleil se le

parser = PlaintextParser.from_string(string, Tokenizer("german"))
summarizer = LexRankSummarizer()
summary = summarizer(parser.document, 2) #Summarize the string with 2 sentences

for sentence in summary:
    print(sentence)
```

```
Près de dix ans s'étaient écoulés depuis que les Dursley s'étaient réveillés pour trouver leur neveu sur le perron, mais Privet Drive n'avait pratiquement pas changé.
Le soleil se levait sur les mêmes jardins de devant bien rangés et éclairait le numéro quatre en laiton sur la porte d'entrée des Dursley ; elle se glissa dans leur salon, qui était presque exact
```

- Use with the HTML link (https://en.wikipedia.org/wiki/Combat):

```
Test 1: simply using LeRank w/o stemming and not removing stop words:

Combat is sometimes resorted to as a method of self-defense, or can be used as a tool to impose one's will on others.
An instance of combat can be a stand-alone confrontation or a small part of a much larger violent conflict.
Instances of combat may also be benign and recreational, as in the cases of combat sports and mock combat.
Wikibooks has a book on the topic of: Fighting
Propensity to Serve and Motivation to Enlist among American Combat Soldiers."

Test2: using LexRankSummarizer with stemming and removing stop words:

Purposeful violent conflict
Combat( French for fight) is a purposeful violent conflict meant to physically harm or kill the opposition.
An instance of combat can be a stand-alone confrontation or a small part of a much larger violent conflict.
Combat Motivation in Today's Soldiers: U.S. Army War College Strategic Studies Institute."
Cohesion during Military Operations: A Field Study on Combat Units in the Al-Aqsa Intifada."
```

Result for other Algorithms for Extractive Summarization.

We are using Wikipedia url for summarization (https://en.wikipedia.org/wiki/Combat).

```
--LexRank Summarizer--
Purposeful violent conflict
Combat( French for fight) is a purposeful violent conflict meant to physically harm or kill the opposition.
An instance of combat can be a stand-alone confrontation or a small part of a much larger violent conflict.
Combat Motivation in Today's Soldiers: U.S. Army War College Strategic Studies Institute."
Cohesion during Military Operations: A Field Study on Combat Units in the Al-Aqsa Intifada."

--LSA Summarizer--
Combat( French for fight) is a purposeful violent conflict meant to physically harm or kill the opposition.
An instance of combat can be a stand-alone confrontation or a small part of a much larger violent conflict.
^ North Atlantic Treaty Organization, NATO Standardization Agency AAP-6 - Glossary of terms and definitions, p. 80
Martin van Creveld: The Changing Face of War: Lessons of Combat, from the Marne to Turkey.
Combat Motivation in Today's Soldiers: U.S. Army War College Strategic Studies Institute."

--Luhn Summarizer--
Armed Forces & Society, vol.
Armed Forces & Society, vol.
Armed Forces & Society, Jul 2006; vol.
Armed Forces & Society, vol.
Armed forces & Society, vol.

--Edmundson Summarizer--
Combat( French for fight) is a purposeful violent conflict meant to physically harm or kill the opposition.
Instances of combat may also be benign and recreational, as in the cases of combat sports and mock combat.
Hand-to-hand combat can be further divided into three sections depending on the distance and positioning of the combatants:
Undermining Combat Readiness in the Russian Military, 1992-2005."
Cohesion during Military Operations: A Field Study on Combat Units in the Al-Aqsa Intifada."
```

Evaluation for the Extractive Summarization above results:

During Observation of output: LexRank seems to give the most concise and coherent summary, closely followed by the Edmundson module.

4.1 Metric based results

Topic	LexRan	Luhn	LSA	Edmundson
Analyze word on a large corpus	Yes	Yes	Yes	Yes
Maintains redundancy (Frequent words are present in summary).	Yes	Yes	No	Yes
Improves coherency.	Yes	No	NO	No
Reduce the dimensionality of the original text -based dataset.	Yes	No	Yes	No
understand what each topic is encoding.	No	No	Yes	Yes
Find Relationship between them	No	No	Yes	No
Removing stopwords	No	Yes	No	No
Accuracy	Yes	No		
Single Document	Yes	Yes	Yes	Yes
Multiple Document	Yes	No	yes	No

Abstractive Summarization:

T5-Model Output:

```
tokenizer = T5Tokenizer.from_pretrained("t5-base")
model = T5ForConditionalGeneration.from_pretrained("t5-base")

print(create_summary(model, tokenizer, article, 512, 120, 5.5, 5.5, 3))
```

```
<pad> robots can be discriminated from other species by three distinctive features or capabilities. the first two capa
None
```

T5-base-finetuned-summerize-news:

```
tokenizer_t5news = AutoTokenizer.from_pretrained("mrm8488/t5-base-finetuned-summarize-news")
model_t5news = AutoModelWithLMHead.from_pretrained('mrm8488/t5-base-finetuned-summarize-news')

print(create_summary(model, tokenizer, article, 512, 120, 5.5, 5.5, 3))
```

```
<pad> robots can be discriminated from other species by three distinctive features or capabilities. the first two capabilities have been attacked by robotics researchers as challenges in locomoti
None
```

Generally, It looks like for a specific NLP task (e.g. 'text summarization') for a specific domain (e.g. 'news summary' here), the fine-tuned model of news summarization t5-base-finetuned-summerize-news produces a bit better and more concise summary than T5-Base model, thanks to the power of transfer learning using this specific news domain training data!

Now testing with custom data text using text file using **T5-base** and **T5-Base finetuned-summerization-news**.

```
with open('/content/sample_data/plain_text_sample_sec.txt', 'r') as f:
    text = f.read()
text
```

```
'Robotics, design, construction, and use of machines (robots) to perform tasks done traditionally by human beings. Robots are widely used
ies where work must be performed in environments hazardous to humans. Many aspects of robotics involve artificial intelligence; robots may
e temperature. Some are even capable of simple decision making, and current robotics research is geared toward devising robots with a degr
ment. Today's industrial robots do not resemble human beings; a robot in human form is called an android.\nCompared to other artifacts wit
n has an important role. A typical example is a humanoid robot, the b...'
```

3ase

```
print(create_summary(model, tokenizer, text, 512, 120, 5.5, 5.5, 3))
```

```
<pad> many aspects of robotics involve artificial intelligence. robots may be equipped with the equivalent of human senses such as vision,
None
```

ase-finetuned-summerize-news

```
print(create_summary(model_t5news, tokenizer_t5news, text, 512, 120, 5.5, 5.5, 3))
```

```
<pad> robotics is the design, construction, and use of machines to perform tasks traditionally done by human beings. While many aspects of
None
```

It can be seen that the DL-enabled abstractive summary looks a bit **more natural, more coherent and more like human-writing summary** than the extractive summary.

4.2 Tuning Hyperparameters:

Extractive Summarization:

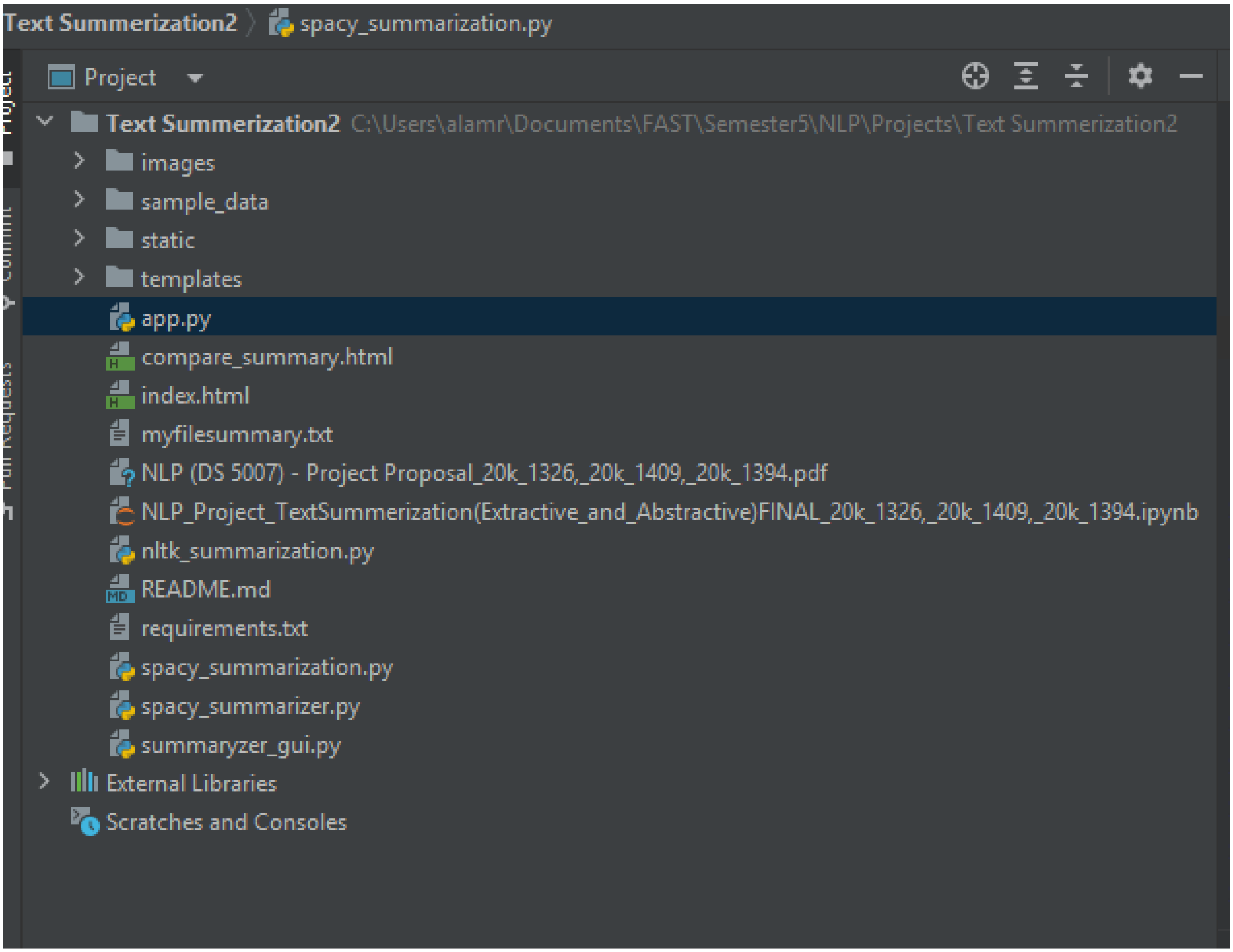
- In Sumy technique we SENTENCES_COUNT is a tunable hyperparameter.
- We can define language params as per type of context in the Tokenizer parameter.
- We can also define how many lines of sentence you are looking for using hyperparameters in the summarizer constructor.
- KEYWORD_RATIO, KEYWORD_SUMMARY, it is also a tunable param which would be responsible for the keyword Ratio at define in ratio param at summary and keyword methods.

Abstractive Summarization:

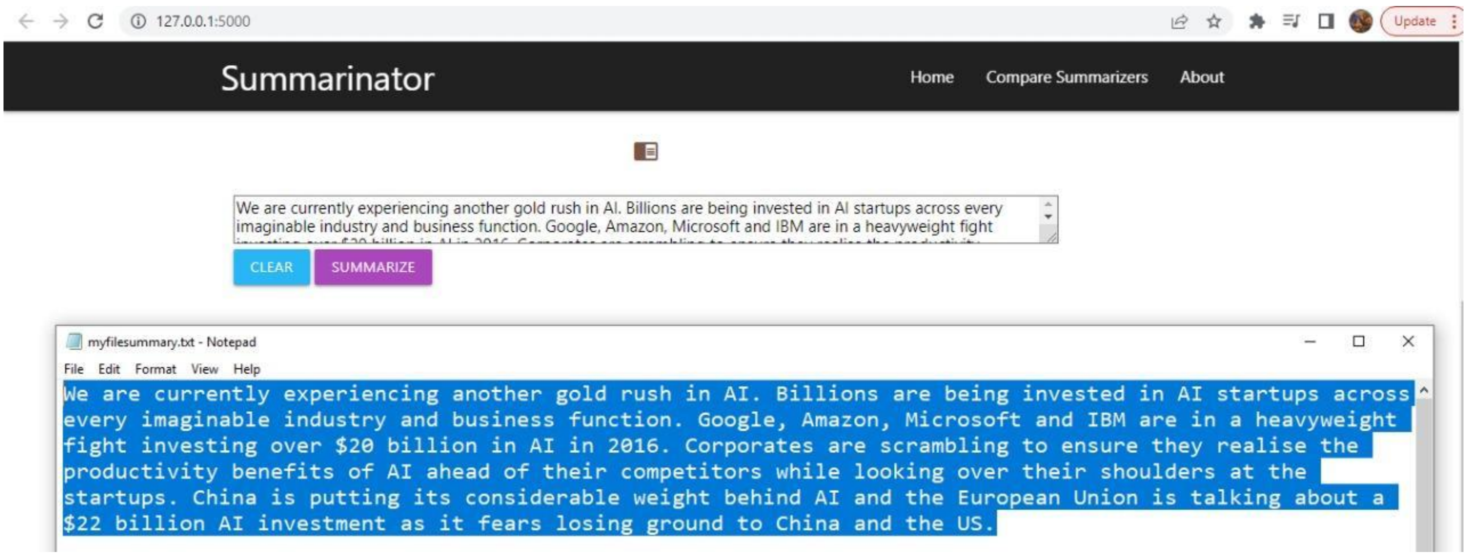
In Abstractive Summarization we have many hyperparameters while creating a summary function in which max_length, min_length, length_penalty, repetition_penalty, num_beams, truncation etc.

5 Summary and Conclusion

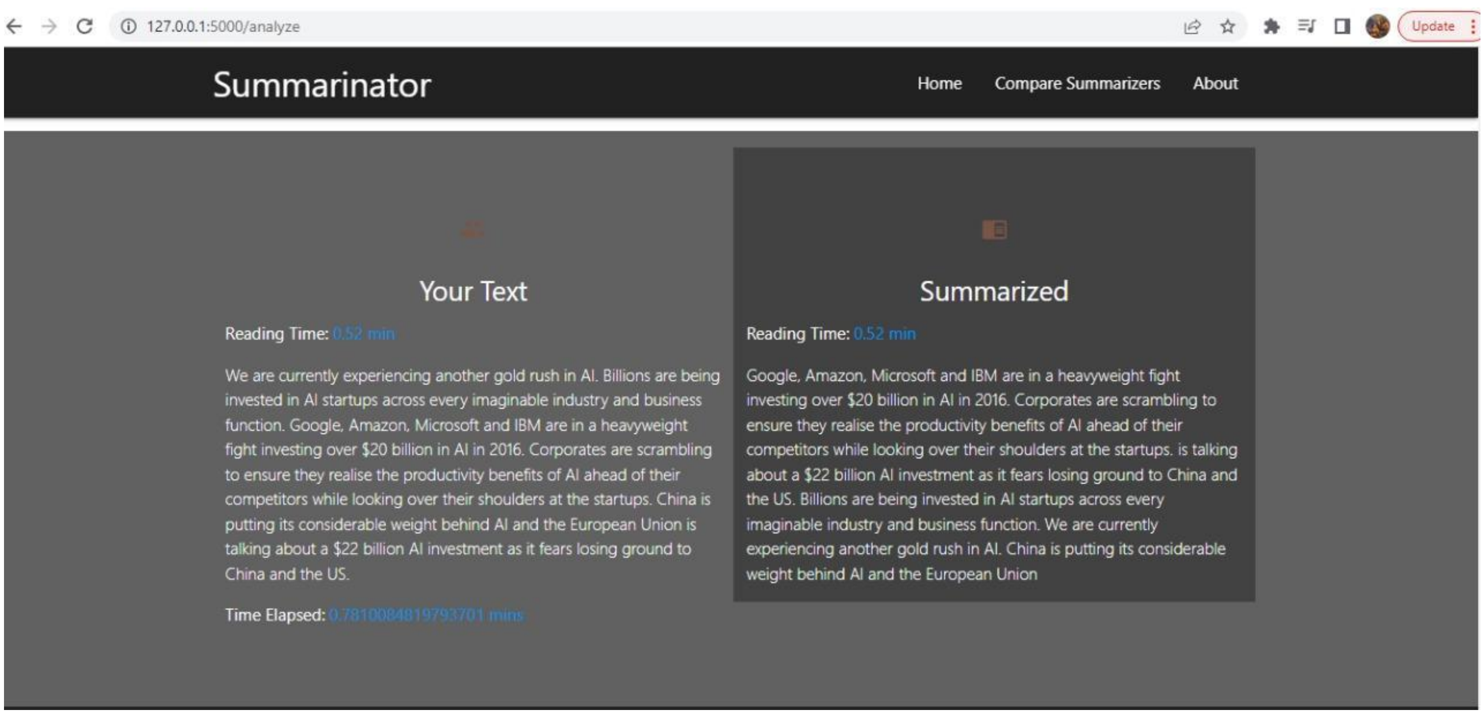
Finally we implement the Spacy, NLTK and Gensim code in different modules and attach it to the main “app.py” file which we deploy using Flask.



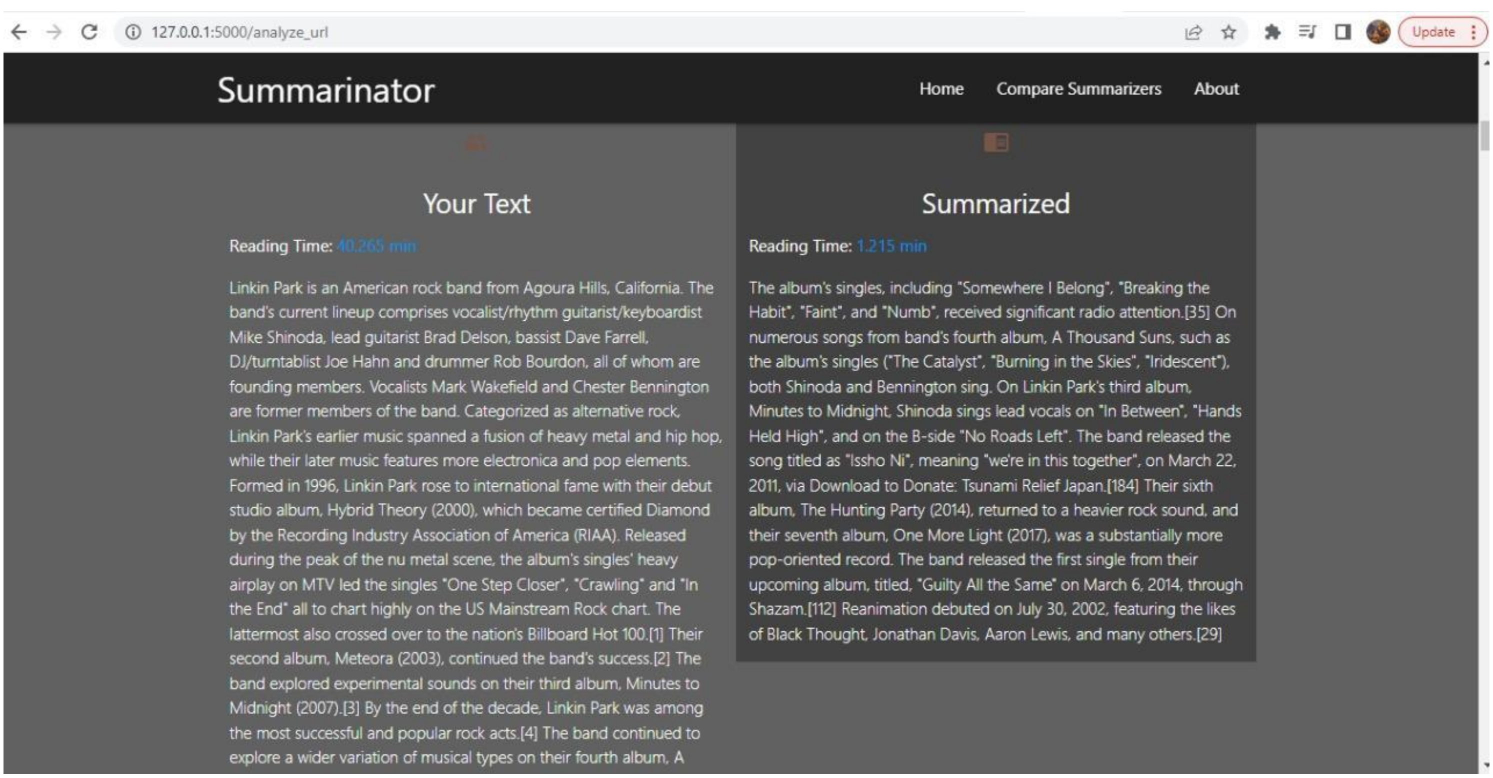
Input:



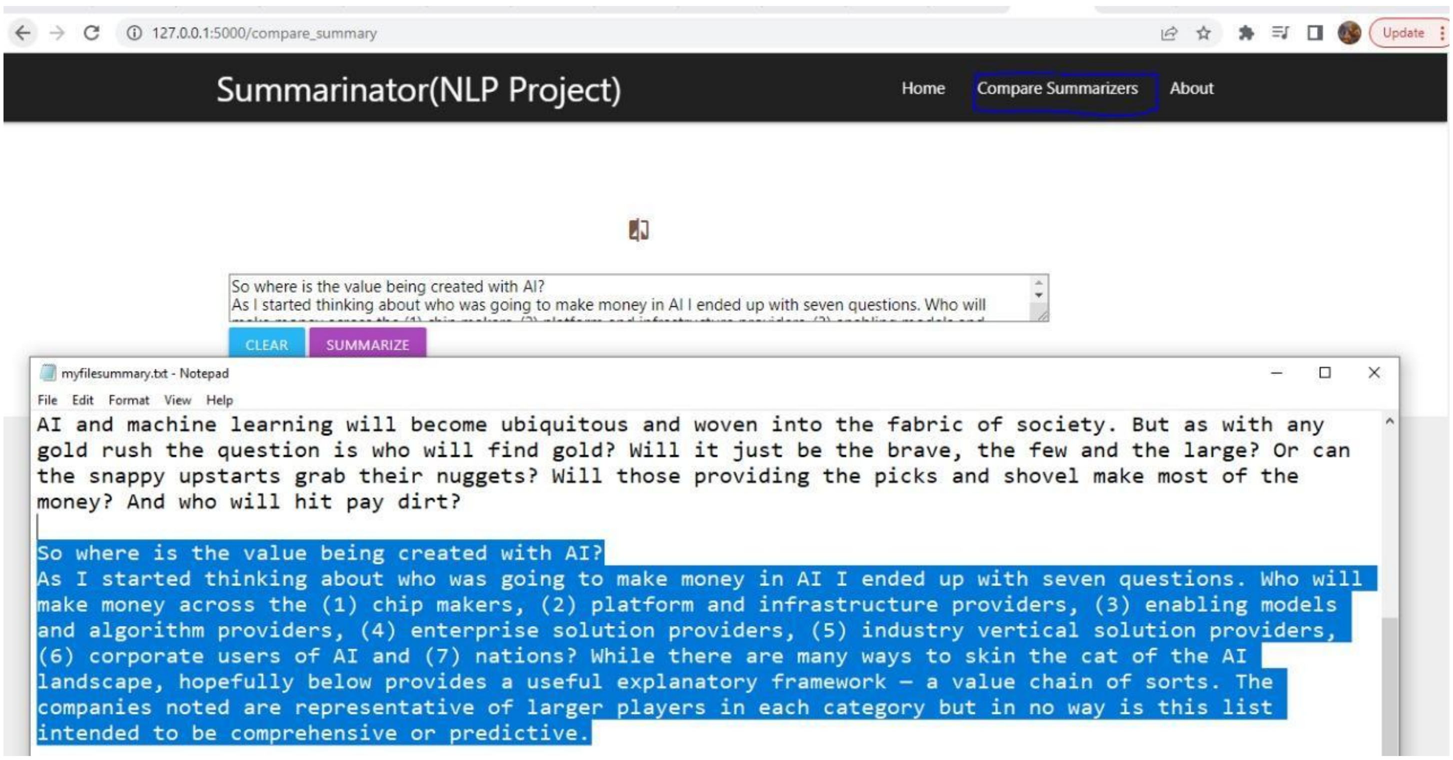
Output:



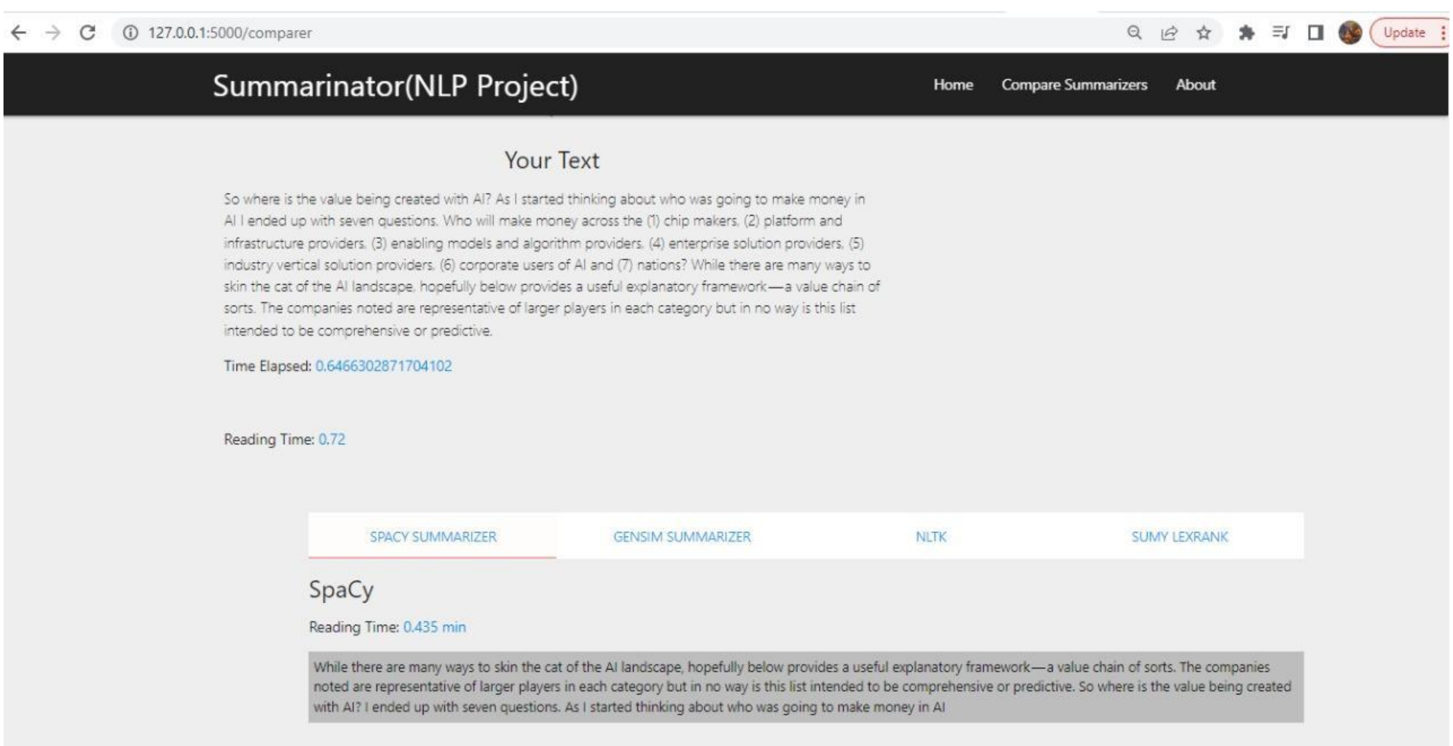
Summarizing by providing the link:



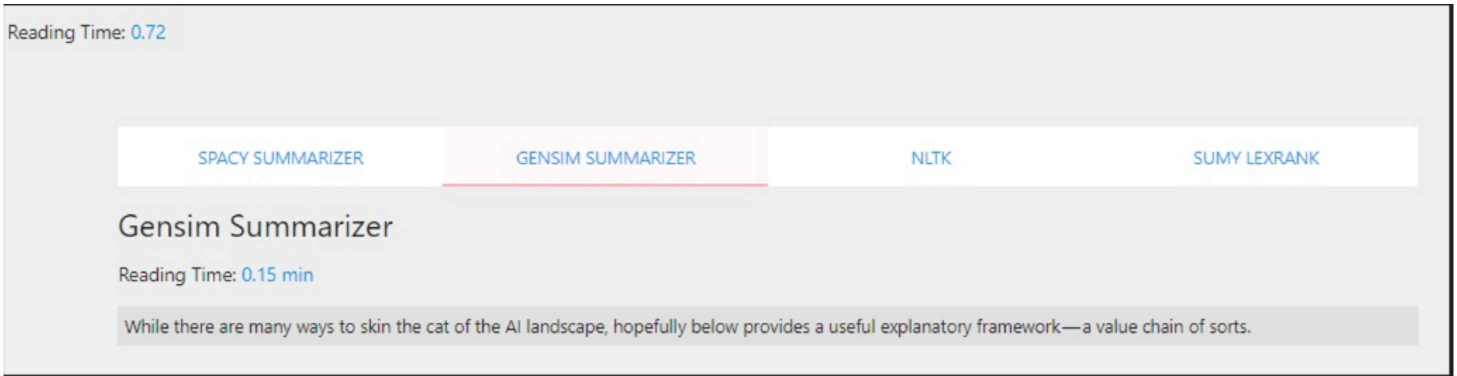
Text Summarization Interface(For comparing summaries):



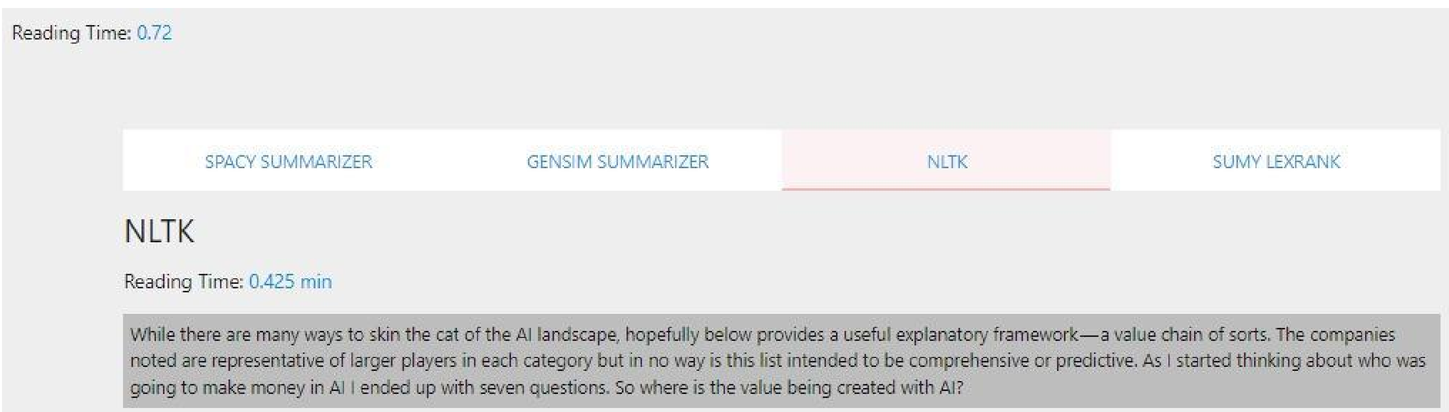
OUTPUT(Spacy Summarizer):



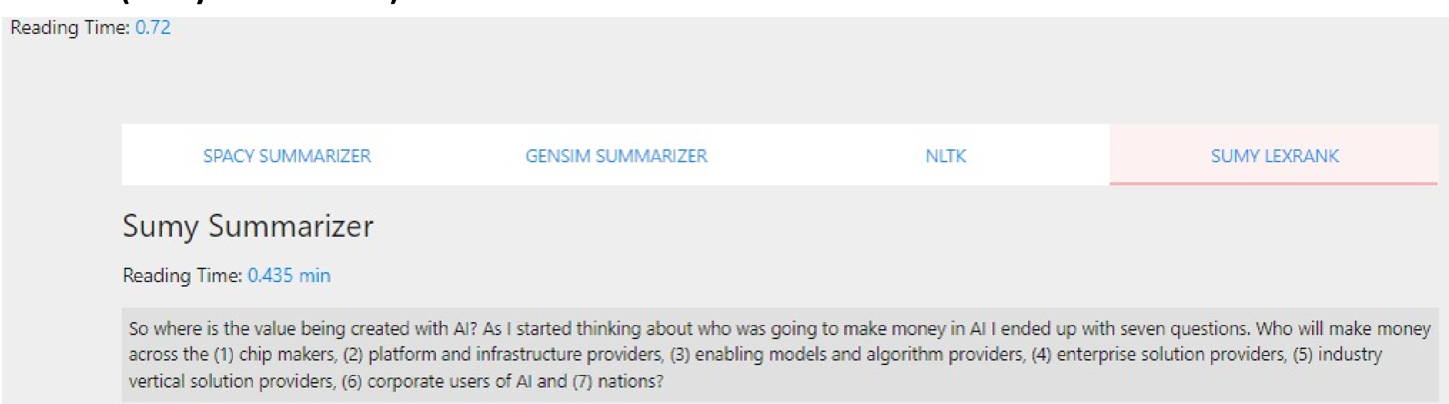
OUTPUT(Gensim Summarizer):



OUTPUT(NLTK Summarizer):



OUTPUT(Sumy Summarizer):



And Hence we successfully explored both Extractive and Abstractive methods of text summarization and deployed them using the Flask web application.