Document Analysis & Summarization using Machine Reading Comprehension

Danish Ahmad Khan
MS in Engineering Science (Data Science)
SUNY Buffalo
danishah@buffalo.edu
Person Number - 50322584

Shreyas Addamane Pallathadka
MS in Engineering Science (Data Science)
SUNY Buffalo
saddaman@buffalo.edu
Person Number - 50320793

Snigdh Chandra
MS in Engineering Science (Data Science)
SUNY Buffalo
snigdhch@buffalo.edu
Person Number - 50322184

***Abstract* — The amount of data in our digital space was said to be around 18 zettabytes in 2018 according to IDC. The increase in this already colossal amount of data is said to be exponential and IDC has projected that by 2025 we will have 175 zettabytes of data. Most of us would need to open google to just understand how big is a "zettabyte". Now a good amount of these zettabytes of data are textual documents which are unstructured, redundant and do not provide us with meaningful information. In this project we have taken such documents or paragraphs from websites and tried to summarize the text in the documents/paragraphs so as we get a gist of what the document is about. Getting a summary of the textual content solves the problem of people having to go through entire set of documents to know what they are about. We then proceeded to incorporate a model which would help the user in getting concise answers about any questions asked about the information in the summarized data through Machine Reading Comprehension. We have also given a basic working and visualization of the Machine Reading Comprehension part using a WhatsApp bot GUI. The methods used and the results obtained have been discussed further in the report.**

***Keywords* — NLP, NLTK, BiDAF, AllenNLP, Twilio.**

I. INTRODUCTION

In today's fast paced world, not everyone has the time or patience to go through the articles/documents of textual content that they come across. It can be people on their way to work trying to read news articles online or students trying to understand a chapter or a specific chapter of the book for their assignments or intelligence agencies who have to go through enormous amounts of documents in a short time to get useful information out of it in cases of emergencies to name a few examples. And in the real world, data is available to us as unstructured, redundant and sometimes unhelpful format. So, if we need some information regarding a certain topic then we need to go through every line of every page of each and every document to find the specific piece of information that is useful to us. And this practice will take a lot of time and could end up in costing people time and probably money. All of these situations can be made easier if a concise summary of the articles or chapters or books or documents where made available to the people. And this is where text summarization comes into picture. In layman's terms, text summarization is a technique of pruning huge amounts of texts in order to get succinct and accurate summary of the texts at hand. So large documents can be shrunk to a small informative set of data. And this will reduce the time taken to read documents and the time taken to search for a specific set of information.

There are two mainly two types of text summarization in NLP:

1) *Extraction-based summarization,*
2) *Abstraction-based summarization.*

In extraction-based summarization, the keyphrases present in the document are extracted and combined which give us a summary of the document. We can think of it as marking important sentences and definitions in our books with a pen or pencil [1], [2]. The summary obtained has all the information that explains the context but these results may not be grammatically correct. In abstraction-based summarization, the use of deep learning techniques is done which rephrase and reduce the document which helps in solving the issue of grammatical mistakes which are observed in extraction-based summarization [1], [2]. Now we look at a short example of these techniques. Consider the text: "Chandler and Monica drove their car to attend an anniversary in NYC. While on their way, they got hungry and stopped the car to eat at a pizzeria". Extraction-based summarization would give us the following summary: "Chandler and Monica attend anniversary NYC. They hungry stopped pizzeria". As we can see that although we can use the summary to figure out the essence of the situation, the summary is still grammatical incorrect. On the other-hand abstraction-based summarization would give us the something like following summary: "They ate at a pizzeria while on their way to an anniversary". As we can clearly see this provides us with better and well-structured summary. We can conclude that abstraction-based technique is a better option, however the algorithms needed for this method are very difficult to implement as they involve deep learning and language modeling techniques which are quite complicated. And this is why the use of extraction-based summarization is more favored over abstraction-based. We will be using extraction-based technique in this project. Now that we have the summary of a text, we can go one step further and incorporate a module that will provide users with answers to the questions asked about any given document. So, people will be able to ask a question and get a response answering their query. And this task which seems difficult can be done with the help of "Machine Reading Comprehension". To give you an overview of what we are expecting the machine to do, let's take a basic example that we have or will face at some point in our life. In our English exams we were given articles or paragraphs and were then asked to answer a certain set of questions after reading and understanding the given text. And this is what we expect the machine to do at this point, to read a document and then correctly answer any question that we ask of it. This

looks like a very difficult task for our machine to execute because we expect the machine to interpret our query accurately and then provide us with the correct response having understood the essence of the question and the document currently under consideration. However, with use of NLP and machine learning models these tasks of reading comprehension have been possible and bit easier to execute. In this project we have attempted to answer questions related to a document by first getting the summary of the document using text summarization and then using machine learning models to get the answers for questions asked by a user.

II. METHOD AND PROCESS

A. *Text Summarization Workflow*

Now that a basic understanding of what Text Summarization has been provided and we know the essence of the process, we will now get into the step by step process carried out in our "Text Summarization" code. We start off by fetching the content or the textual content of the document from which we want the answers. We start by using the "urllib.request" module to open the online document and fetch the data (Figure 1). Then we use the "BeautifulSoup" package in python to parse the data obtained from the url and store it in a variable. And then fetch all values in the <p> tab of the parsed data. We have to do this because the data fetched from the url is in the format displayed in Figure 1 and Figure 2.

```
b'\n<!DOCTYPE html>\n<html class="client-nojs"
lang="en" dir="ltr">\n<head>\n<meta
charset="UTF-8"/>\n<title>Rahul Dravid -
Wikipedia</title>
\n<script>document.documentElement.className="cl
ient-js";RLCONF={"wgBreakFrames":!
1,"wgSeparatorTransformTable":
["",""],"wgDigitTransformTable":
["",""],"wgDefaultDateFormat":"dmy","wgMonthName
s":
["","January","February","March","April","May","
June","July","August","September","October","Nov
ember","December"],"wgRequestId":"Xq@K3gpAIEIAAL
jE2mcAAAAA","wgCSPNonce":!
1,"wgCanonicalNamespace":"","wgCanonicalSpecialP
ageName":!1,"wgNamespaceNumber":
0,"wgPageName":"Rahul_Dravid","wgTitle":"Rahul
Dravid","wgCurRevisionId":
947756884,"wgRevisionId":
947756884,"wgArticleId":252735,"wgIsArticle":!
0,"wgIsRedirect":!
1,"wgAction":"view","wgUserName":null,"wgUserGro
ups":["*"],"wgCategories":["Webarchive template
wayback links","Pages with citations lacking
titles","Pages with citations having bare
URLs","All articles with dead external
links","Articles with dead external links from
June 2016","Articles with dead external links
from December 2017","Articles with permanently
dead external links",\n"Use Indian English from
February 2013","All Wikipedia articles written
in Indian English","Use dmy dates from January
2013","Wikipedia indefinitely semi-protected
biographies of living people","Articles with
hAudio microformats","Pages including recorded
```

Figure 1: Unparsed data fetched from the url of our document.

As we can see from Figure 1 that the unparsed data (html code) will not give us any meaningful data. So we parse the data so that get information relevant to us like the title, headings, paragraphs, tables and such in a html tags format (Figure 2) so that we can get the textual content for us to start the summarization.

```
<li><a href="/wiki/A._V._Rama_Rao" title="A. V.
Rama Rao">A. V. Rama Rao</a></li>
<li><a href="/wiki/D._Nageshwar_Reddy" title="D.
Nageshwar Reddy">D. Nageshwar Reddy</a></li>
<li><a href="/wiki/
Dayananda_Saraswati_(Arsha_Vidya)"
title="Dayananda Saraswati (Arsha
Vidya)">Dayananda Saraswati</a></li>
<li><a href="/wiki/Barjinder_Singh_Hamdard"
title="Barjinder Singh Hamdard">Barjinder Singh
Hamdard</a></li>
<li><a href="/wiki/Ram_V._Sutar" title="Ram V.
Sutar">Ram V. Sutar</a></li>
<li><a href="/wiki/Tejomayananda"
title="Tejomayananda">Tejomayananda</a></li></
ul>
```

Figure 2: Parsed data obtained by using BeautifulSoup.

After parsing the data, we iterate through every data within the <p> tags (basically every paragraph in the document), concatenate them and store them in a variable. Now that we have the text we start by creating a dictionary which stores the frequency of every word that is used within the text. In order to do that we start by first tokenizing the words which is a list of every word occurring in our document (Figure 3).

```
['Rahul', 'Sharad', 'Dravid', '(', '/ˌrɑhuːl',
'drɑvɪd/', '(', 'listen', ')', ';', 'born',
'11', 'January', '1973', ')', 'is', 'a',
```

Figure 3: List of tokenized words.

Then we iterate through the above list of tokenized words and create the above-mentioned library which stores the frequency of occurrence of all words. While doing this, we have to be aware of two sets of word set. The first one is the set of common words such as "of", "or", "in" which do not really give us any information, so using the "nltk" we remove such words which are called as stopped words. The second set are words such as "drink" and "drinks", where these words have the same stem (drink) but with extra letters. So, using "PorterStemmer" we reduce all words to their root form. And then we obtain a dictionary similar to Figure 4.

```
'australia': 25, 'delhi': 2, 'posit': 6, 'three-
match': 1, 'africa': 11, 'ahmedabad': 2, '175':
2, 'modest': 1, '29.16': 1, 'week': 1, 'three-
```

Figure 4: Portion of the word-frequency dictionary.

Then we tokenize every sentence in the document just like we tokenized words (in Figure 3), which gives us a list of all the sentences in the document. We proceed to calculate the score of every sentence by the keywords present in it using the word-frequency table that we obtained earlier. Next, we calculate the average score of all the sentences, which we obtain the dividing the score of all the sentences with the number of sentences in the document. We will

use this average score of the sentences to multiply with a threshold value so as to eliminate the sentences which might have lower effect on the understanding of the article. So, such sentences will be eliminated from the summary of the text. Finally, we eliminate sentences which we do not want to display as part of our summary by using a certain threshold value. Now that we have our text summary, we move onto the Machine Reading Comprehension part of our project.

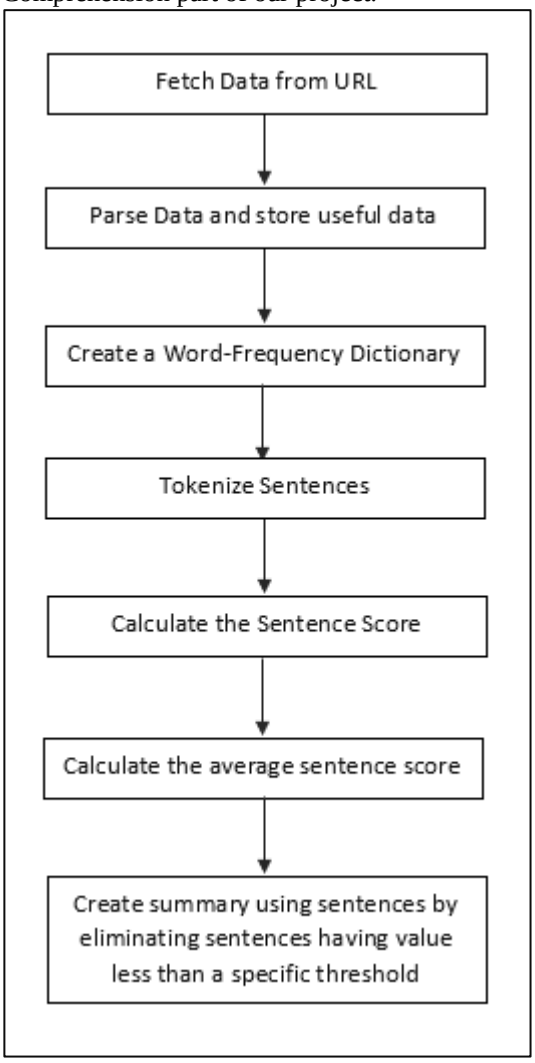

Figure 5: Flowchart of Text Summarization Process.

B. *Machine Reading Comprehension*

Before we utilized the concept of Machine Reading Comprehension, we tried a few other methods. One method which was the best of worst methods we tried, was to return the whole sentence from the text, which contained the most important(frequent) word from the question asked. Now this was a very naïve approach because the problem with this approach was that multiple sentences could have had that same word, so we couldn't choose the best sentence with accuracy. Then we included cosine similarity, mapping the question keywords with the sentences from the text, and returned the one which proved to be the most similar to the questions asked by the user. We had to rule out this approach as well because the outputs we got were too wordy to be considered and they lacked the basic human way of giving answers, they always started with useless information, most of the times. After researching for various efficient methods and going through lots of research papers and projects, possible solutions and intensive arguments, we finally found the optimal solution, AllenNLP [3]. It is a library which is developed by AllenAI and used by many huge tech companies such as Google, Facebook and Netflix to skim through immeasurable amounts of textual data and generate sentiments. We fine-tuned this library to fit a predictor model called BiDAF. BiDAF stands for Bi-Directional Attention Flow. We took it from a research paper [3] published by a professor, working in the Cornell University because it used a neural network which performed awesomely on the Stanford Question Answering Dataset (SQuAD) which was similar to the kind of performance we desired. The predictions were in the form of a JSON array, we filtered it to give us the best possible answer to the question. The best part about using this was that it was giving query-aware answers to our questions and not the whole trivial sentences. The basic workflow of the BiDAF model can be seen in Figure 6.

BiDAF network is a multi-stage hierarchical process that represents the context at different levels of granularity and uses bidirectional attention flow mechanism to obtain a query-aware context representation without early summarization [4]. To further explain this in layman's terms, BiDAF is a model which does not take into factor any pre-existing data/knowledge but the data that is currently under consideration (in this scenario, a textual document) and returns a part of the textual content that is most pertinent to the query currently at hand. Basically, the response received from this model will be found verbatim within the textual document [5]. It is important to remember that this model will only work with queries that have a sort factual question [5]. In short, BiDAF will always need a "Document/Context" to answer any query that is asked of it and the answer will always be a subset of the document at hand. To provide you with an example, let's consider the following text: "Rahul has a younger brother called Vijay" and we ask the system "Who is the Rahul's brother?". The response to this will simply be "Vijay". Now, we will go through the workflow of the BiDAF model, Figure 6 adapted from [4], to try and understand what the model's inner workings as explained by the author in [4]. As we can see in Figure 6, there are 6 layers in this model: Character Embed, Word Embed, Contextual Embed, Attention Flow, Modelling and Output Layer. The job of the character embed layer is to map each word to vector space using Character-Level Convolution Neural Networks (CNNs) [4]. The Word Embed Layer maps each word to a vector space but in this case, it uses a pre-trained word vector to get fixed word embedding of each word and the concatenation of these is passed through a 2-layer Highway Network which produces two matrices (1 each for Context and Query) [4]. The contextual Embed layer utilizes contextual cues from surrounding words to refine the embedding of the words [4]. The Attention Flow Layer combines the query and context vectors, then produces a set of query-aware feature vectors for each word of the context [4]. The Modeling layer scans the context using a Recurrent

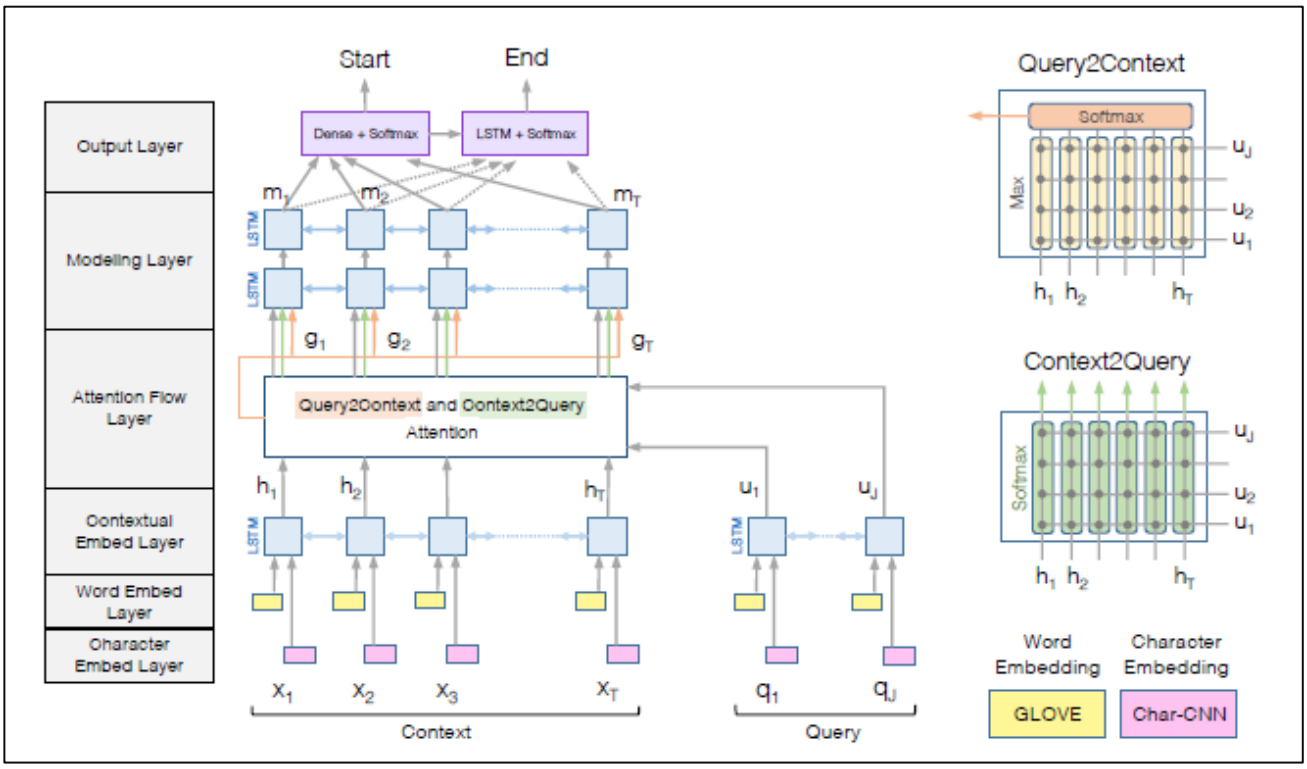

Figure 6: Workflow of the BiDAF Model.
Source: Adapted from [4].

Neural Network [4]. And finally, the output layer provided the answer to query at hand [4]. In this workflow, the first 3 layers were applied to the Context as well as the query. An in-depth explanation can be found in Section 2 of [4] which is highly interesting.

C. *Twilio and WhatsApp*

Now that we have module for text summarization and module for answering the questions asked by the user, we need an Interactive UI which will make it easier for the user to ask questions and receive the response. The user will interact with the system via WhatsApp and the system receives and responds to the user using Twilio, which is a cloud communications platform. Twilio allows developers to send and receive WhatsApp messages programmatically, and perform various communication functions with the help of its web service APIs.

The code for the setup of Twilio is within the "qabot.py" python file. Using the bot function within this file, we receive the questions from the user one at a time as an incoming message and using the predict function which has the parameters: Summarized Passage and incoming question, returns the appropriate response to the user. We open our python files in Spyder (Python) IDE using Anaconda. We then open the Anaconda prompt and run our flask app. Once a connection has been established the window will look something like Figure 7. Then, we execute a "ngrok" file and temporary connection has been established which we have to use to connect our system and the user explained in the next. The ngrok execution results looks like as it is in Figure 8 (Ngrok is a multiplatform tunneling, reverse proxy software that establishes secure tunnels from a public endpoint such as internet to a locally running network service while capturing all traffic for detailed inspection and replay [6]). Once we have done that, we get a url ("https://7d472ba2.ngrok.io" in Figure 8) which we use for configuration of our sandbox for when the system receives a question from the user (The installation docs for Twilio Sandbox for WhatsApp is available on the Twilio documentation site).

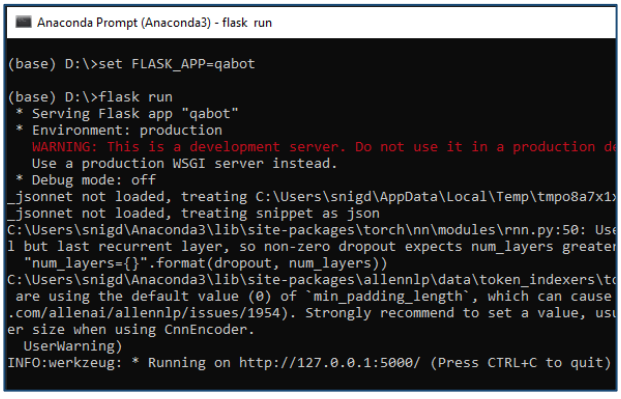

Figure 7: Flask app run window

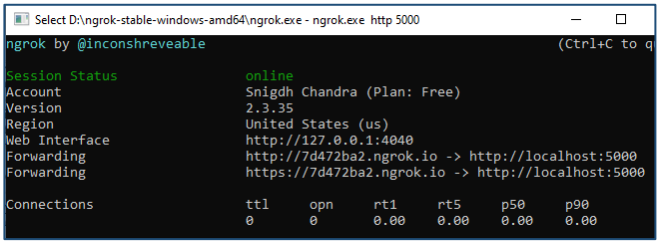

Figure 8: ngrok executable file

Once we are done with making the connection, we will have effectively created a two-way communication line where the user can ask questions and the system will be able to respond to the questions.

III. RESULTS

The first set of results we would like to present are the Summarized passage that we obtained using our Text Summarization code. Below (Figure 9) is a section of summarization of the Wikipedia Page of Machine Learning.

```
 The study of mathematical optimization
delivers methods, theory and application
domains to the field of machine learning.
Some statisticians have adopted methods
from machine learning, leading to a
combined field that they call statistical
learning. If the hypothesis is less
complex than the function, then the model
has under fitted the data. If the
complexity of the model is increased in
response, then the training error
decreases. [27] The data is known as
training data, and consists of a set of
training examples. [14]
Types of supervised learning algorithms
include Active learning , classification
and regression. The algorithms, therefore,
learn from test data that has not been
labeled, classified or categorized. In
machine learning, the environment is
typically represented as a Markov Decision
Process (MDP). In supervised feature
```

Figure 9: Section of summarization of ML Wikipedia Page

Below (Figure 10) is a section of the Wikipedia Page of the famous former football player David Beckham.

```
Beckham played a full 90-minutes of the
fixture which ended 3–2 in favour of
Manchester United and 6–3 on aggregate.
[36]
Manchester United again reached the final
of the FA Youth Cup, where they faced
Leeds United. In the second leg on 13 May
1993, Beckham played the full 90 minutes
of Manchester United's 2–1 defeat, which
gave Leeds United a 4–1 aggregate score.
However, United recovered from this early
season defeat and the young players
performed well. He is such a big
```

Figure 10: Section of summarization of David Beckham Wikipedia Page.

Now we move on to the Interactive UI results where the user asks questions from the system. Figure 11 displays a User interaction with the system, where the user asks the system questions about a famous former Indian cricket player “Rahul Dravid”.

Figure 11: User interaction with system via WhatsApp.

As we can see in the Figure 11, the system is able to provide the user with correct answers related to Rahul Dravid. These results have been verified by cross-checking with the Wikipedia page of Rahul Dravid. As we can see that we are not getting long and explained answers with proper grammar, this because of the fact that we are using extraction-based summarization which does not provide us with proper sentences which considers the grammatical syntactics as discussed in the Introduction section of the report.

IV. INDIVIDUAL MEMBER CONTRIBUTION

We carried out our project in 3 modules. First, the text summarization module, then the machine learning comprehension module and finally, the creation of the UI. The extensive research for Text Summarization and Machine Reading Comprehension was carried out by all the 3 members throughout the duration of the project to get as much information as possible from alternate sources. The implementation of Text Summarization was carried out by Shreyas and Snigdh. The implementation of Machine Reading Comprehension was carried out majorly by Danish with a bit of help from the other team members. The UI implementation was carried out majorly by Shreyas with the help of Snigdh due to their familiarity with using Twilio, WhatsApp and Chatbot. The milestones 1 and 2 were written by Danish with inputs regarding the Methods and Preliminary Results section by Shreyas and Snigdh. The project report was majorly written by Snigdh with inputs from Danish and Shreyas. Throughout the project, all team members understood every step of the implementation and the reasoning behind the implementation. The execution of every module was carried out all three members on their respective machines with suggestions so as to get suggestions for any improvement throughout the project.

V. RELEVACNE AND SIGNIFICANE

We believe that **"Text Summarization (TS)"** and **"Machine Reading Comprehension (MRC)"** can be a helpful module in various domains of the world. It can be helpful for students and professors in schools or colleges, medical professionals in health sector, people who are looking to write research papers and professionals in IT or Finance or Government sector. Let's go through a few examples. Students who are looking for an answer to a query can use **MRC** to get the answer and then use **TS** to get a summary of the chapter or section of the chapter to understand the reasoning behind the answer. Teachers can use **TS** to go through answer sheets to find out if any keywords have been used which they would expect in a solution. People who are writing research or any other kind of papers have to go through a lot of documents to find answers to their queries like us and many our classmates for this project. All of us can benefit from **TS,** if we had summaries of the papers available to us and not just an abstract of the paper because then everyone will have to read through the paper if the paper gives even an inkling of what we are looking for. Professionals in other sectors can use **TS** and **MRC** to summarize documents they have to read and get quick short answers to their queries from a document. Now it is important to remember that we have implemented an extraction-based summarization so the summary and answers will be short summaries and answers which look like a concatenation of majorly keywords and key-sentences which will be helpful in the above scenarios. A possible improvement to this has been discussed in the Future Work section.

Throughout the duration of the project, we came into topics such as Unsupervised Learning, Tensors, Tensor flow, Neural Networks and Convolution while going through various papers for implementation and in-depth understanding of Test Summarization and Machine Reading Comprehension.

VI. CONCLUSION

In this project we have executed 3 modules. First module deals with the execution of a Text Summarization whose task was to take any textual document as an input and then give a summary of the document as the output. The second module deals with the execution of Machine Learning Comprehension whose task was to take in two inputs, the summary of the document from the first module and a question regarding the document from the user, and then give out a response that would answer the query of the user as accurately as possible. In the third module, we created and implemented a UI which would let a user ask the system questions and in return get the correct response. Throughout the duration of the project, we gained knowledge regarding the difficulties faced while attempting to solve a ML problem. We also realized how closely a ML solution is a "black box" and the beauty of the logic, reasoning and implementation required to turn a set of inputs to just a single output.

VII. FUTURE WORK

As we discussed in the Section V about the possible uses of our project in the real world. We also mentioned that improvements can be made to the modules implemented. Our final output would give the user mostly a sentence which would answer their queries but the response would be a sentence having the keywords to answer the query but not in the correct grammatical syntax. However, with the use of abstraction-based summarization this issue can be solved which would provide the user with a response which would capture the essence of the query rather than just answering the query with an assortment of words. If this system would be implemented with more than 90% accuracy, then this could be used by professionals in Intelligence agencies or Finance sector or Medical sector to get precise and accurate answers regarding a document quickly which would be really helpful for them as they work in a very high-paced environment and getting precise solutions in a quick space of time can help them in saving a life (Intelligence or Health) or make a substantial profit (Profit).

VIII. REFERENCES

[1] [1] J. Brownlee, "A Gentle Introduction to Text Summarization", *Machine Learning Mastery*, Nov. 29, 2017. [Online]. Available: https://machinelearningmastery.com/gentle-introduction-text-summarization/. [Accessed: Feb. 15, 2020]

[2] M.J. Garbade, "A Quick Introduction to Text Summarization in Machine Learning", Towards Data Science, Sept. 18, 2018. [Online]. Available: https://towardsdatascience.com/a-quick-introduction-to-text-summarization-in-machine-learning-3d27ccf18a9f. [Accessed: Feb. 15, 2020]

[3] M. Gardner et al., "AllenNLP: A Deep Semantic Natural Language Processing Platform", arXiv:1803.07640 [cs.CL], March 2018.

[4] M. Seo, A. Kembhavi, A. Farhadi, H. Hajishirzi., "Bi-Directional Attention Flow For Machine Comprehension", arXiv:1611.01603v6 [cs.CL], June 2018.

[5] M. Antonio, "An Illustrated Guide to Bi-Directional Attention Flow (BiDAF)", Towards Data Science, Aug 28, 2019. [Online]. Available: https://towardsdatascience.com/the-definitive-guide-to-bi-directional-attention-flow-d0e96e9e666b. [Accessed: March 10, 2020]

[6] U. Kiran, Expose your localhost to web in 50 seconds using Ngrok, Vmoksha, Available: https://vmokshagroup.com/blog/expose-your-localhost-to-web-in-50-seconds-using-ngrok/ [Accessed: Apr 10 2020]