IJRAR.ORG E-ISSN: 2348-1269, P-ISSN: 2349-5138

INTERNATIONAL JOURNAL OF RESEARCH AND ANALYTICAL REVIEWS (IJRAR) | IJRAR.ORG

An International Open Access, Peer-reviewed, Refereed Journal

TEXT SUMMARIZATION WITH PYTHON

[1]Mr.Satya Mohan Chowdary G, [2]Ms.S.M.M.Valli, [3]Ms.P.Lakshmi Bhavana, [4]Mr.A.Shyam Vineel, [5]Mr.A.Lakshmi Sriya,

[1]Assistant Professor, [2]B.Tech. III Year Student, [3]B.Tech. III Year Student, [4]B.Tech. III Year Student,
[5]B.Tech. III Year Student,
[1]IT Department,
[1]Pragati Engineering College(A), Surampalem, A.P, India

Abstract : The process of creating a summary from a huge amount of information while maintaining the actual information context is called a text summary.Summarization structures regularly have extra proof they could make use of for you to specify the maximum critical subjects of document(s). For example, while summarizing blogs, there are discussions or remarks coming after the weblog publish which are properly re-assets of facts to decide which elements of the weblog are crucial and interesting.The main idea behind the automatic text summarization is to find a short subset of the most important information in the whole and make it visible to humans.

Keywords– Huge amount, weblog, text, whole, visible, summary, summarizing, summarization, subset, subjects.

I. INTRODUCTION

Text Summary Python helps you summarize and summarize user feedback text. This can be done using algorithms that help reduce the body of the text while preserving its original meaning, or by providing insight into the original text.

There is a huge amount of textual material, which is growing every day.

Think of the Internet, which consists of websites, news articles, status updates, blogs, and more. The data is unstructured and the best thing you can do to navigate the data is to use search to skim the results. Much of this textual data needs to be reduced to short, focused summaries that capture important details. This allows you to navigate more effectively and make sure that the larger document contains the information you are looking for.

A.TYPES OF TEXT SUMMARIZATION

An executive summary is a small section of text that covers important points and accurately reflects the meaning of the original document. Text summarization is a way to summarize a document in a few sentences. This can be done in two ways:

1. Abstractive Text Summarization
2. Extractive Text Summarization

1. Abstractive Text Summarization:

Abstract methods select words based on their understanding of meaning. Even those words didn't show up in the source document. We aim to create important materials in new ways. They use advanced natural language techniques to interpret and explore the text and create new short texts that convey important information from the original text. This can be associated with how people read text articles and blog posts and summarize them in their own words.

Enter Document → Understand Context → Semantics → Create your own summary.

2. Extractive Text Summarization:

The extraction method attempts to summarize the article by selecting a subset of words that contain the gist. This approach weights important parts of the sentence and uses the same to form a summary. Various algorithms and techniques are used to define sentence weights and further rank them based on their mutual importance and similarity.

Select Input Document ➝ Sentence Similarity ➝ Weighted Sentence ➝ Highly Ranking Sentence.

B. The need for text summarization

With the current explosive growth of data circulating in digital space, mostly unstructured text data, we need to develop automated text summarization tools that allow people to easily extract insights from it. Today, you can quickly access vast amounts of information. However, most of this information is verbose and irrelevant and may not convey its intended meaning. For example, if you're looking for specific information from an online news article, you'll need to spend a lot of time sifting through that content and organizing what you don't need before you get the information you need. Therefore, it is becoming increasingly important to use automatic text summarization that can extract useful information that omits irrelevant data and irrelevant data. Implementing summaries improves the readability of documents, reduces the time it takes to retrieve information, and allows you to fit more information in a particular area.

II. LITERATURE REVIEW

1. A hybrid approach to extract key-phrase robotically for multi-record text summarization in e-newspaper articles. This approach has extracted spherical 96.23% of semantically now no longer unusual place sentences maximum of the articles.
2. An approach for record summarization using Agglomerative hierarchical clustering, K technique clustering, DBSCAN clustering and text analytic techniques to reduce the facts redundancy.
3. The implementation of Text Summarization the use of Unsupervised Text Rank Algorithm and changing the summarized textual content into Audio File Using Google Text To Speech API.Large chunks of Text/ internet web page URL hyperlink is given as enter and Summary is generated and convert the Summary into Audio report Using GTTS API.
4. The effective diversity – based totally definitely approach combined with K-suggest Clustering set of regulations to generating summary of the record. The clustering set of regulations is used as supporting element with the approach for finding the most first-rate mind with inside the text.

We construct a textual content summarizer in which the Enter is a protracted collection of words (in a textual content body), and the output is a quick summary (that is a chain as well). So, we will version this as a Many-to-Many Seq2Seq problem.

#METHODS USED

1. SEQ2SEQ MODELING

The use of the Sequence to Sequence (Seq2Seq) model for summarizing neural abstract text has become increasingly popular in recent years. Many interesting strategies have been introduced to improve the Seq2seq model, overcoming various issues such as interpretability, comprehension, and human readability, and enabling us to create high quality summaries. Most of these techniques fall into one of three categories: network structure, parameter inference, or decode / generate. Another aspect to consider is that the efficiency and concurrency of model training is generally (but exclusive) for dealing with complex language issues such as machine translation, question answering, chatbot creation, and text summarization. Is used). The Seq2Seq model takes a stream of statements as input and produces another stream of statements as output.

The two main strategies used in Seq2Seq modeling are encoders and decoders:
Encoder Model: The encoder model is used to encode or modify the input phrase and provide feedback at each step. When using LSTM layers, this feedback can be internal.
Decoder Model: The decoder model is used to decode or predict the target text word by word. The decoder's input data accepts the target sentence as input and infers the next word. The next word is sent to the prediction layer below.

2. LSTM APPROACH

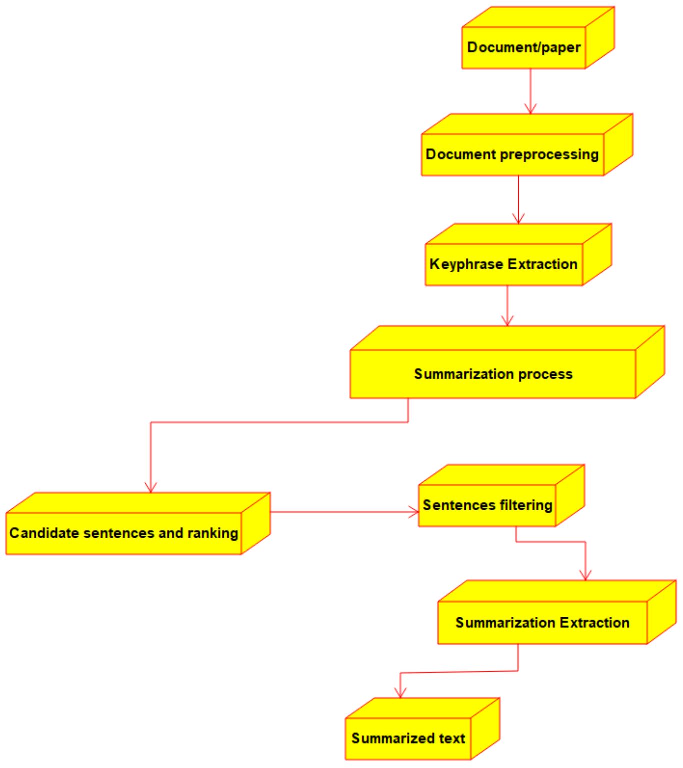
[1] Use Stacked LSTMs with 3 layers of LSTMs stacked.
[2] The first LSTM layer has an encoder input, so it creates a continuous sequence of LSTM layers.
[3] The LSTM layer captures all contextual information present in the input sequence.
[4] Returns the hidden state and the output of the state. H. Hidden state and cell state after execution of each LSTM layer.

3. Attention Layer

The hidden states of every detail with inside the enter series are generated through the Attention Layer Encoder. Between the preceding decoder hidden nation and every of the encoder`s hidden states, Alignment Scores are determined. The ultimate encoder hidden nation may be applied because the decoder`s preliminary hidden nation. Each encoder's hidden nation alignment rankings are pooled and expressed in a unmarried vector, that's then softmaxed. The context vector is fashioned through multiplying the encoder hidden states and their respective alignment rankings. The context vector is given into the Decoder after being concatenated with the previous decoder output.

III. SYSTEM ARCHITECTURE

A new method to supply a précis of a unique textual content investigated on this paper. The device develops many processes to solve this problem that gave an excessive result. The version includes 4 ranges. The preprocessing ranges convert the unstructured textual content into structured. In first degree, the device eliminates the forestall words, pars the textual content and assigns a POS (tag) for every phrase inside the textual content and saves the about a table. The 2d ranges to extract the critical key phrases inside the textual content via imposing a brand-new set of rules through rating the candidate words. The device makes use of the extracted keywords / key phrases to pick the critical sentence. Each sentence ranked relying on many functions together with the lifestyles of the keywords / rephrased in it, the relation between the sentences and the identified via means of the use of a similarity dimension and different functions. The third degree of the proposed device is to extract the sentences with the best rank. The fourth degree is the filtering degree. This degree decreased the quantity of the candidate sentences inside the precis in an effort to produce a qualitative precis.



IV. PERFORMANCE AND ANALYSIS

A. IMPLEMENTION

1.The steps involved in the implementation are:

1.Consumer Import the libraries
2.Parse the dataset file
3.Preprocessing
4.Splitting the records
5.Text vectorization
6.Build the model
7.Train the model
8.Inference model

1. Consumer Import the libraries:

First, we will make a textual content summarizer.py document and import all the libraries.

2.Parse the dataset file

We'll visit the 'Reviews.csv' dataset document and extract all the enter and goal texts. For the education and trying out portion, we're going to use the primary 1,00,000 rows of our dataset. It may be adjusted to fit your needs. The 'Text' column, that's the assessment column, can be our enter, and the 'Summary' column can be our aim. Duplicate facts and NA values may also be eliminated from our facts set.

3.Preprocessing

Real-global texts are imperfect, and sending them to the version at once will bring about inaccuracies. As a result, we easy up all of our texts and flip them right into a layout that can be used for prediction tasks. So, first and foremost, we will installation all the variables and methods.

Because a number of our texts are in html layout and incorporate html tags, we will use the 'Beautiful Soup library' to parse the textual content and cast off all the html tags. After that, we flip our texts into phrases via way of means of tokenizing them. Also, ensure that the subsequent situations are met:

1) Consists of integers
2) They have fewer than 3 characters or
3) They're in a prevent phrase

If one of the above situations is met, the phrase can be eliminated from the listing of enter or goal phrases.

In our enter or goal texts, we additionally have contraction phrases, which might be phrases blended and abbreviated via way of means of an apostrophe or via way of means of eliminating letters, for example, 'haven't' is shortened for 'have not'. The 'contractions.pkl' file, which includes a dictionary with keys as abbreviated phrases and values as enlarged phrases, has been used to increase positive kinds of phrases. We need to additionally stem all the enter phrases returned to their base phrases.

Stemming: The exercise of decreasing phrases to their underlying phrases is called stemming.

We filtered reproduction phrases and taken care of them as a result after cleansing the texts.

4.Splitting the records

We have break up the dataset information into education and trying out units. We have break up it in 80:20 ratio in which 80% file is for education units and 20% for trying out units.

5.Text vectorization

We have transformed our phrases into integer collection the usage of vectorization technique.

For example,

L = [‘what doing’, ‘how are you’, ’good’]

D = {‘what’ : 1 , ‘doing’ :2 , ‘how’ : 3 , ‘are’ : 4 , ‘you’ :5 , ‘good’ : 6 }

So, we've in shape our facts, now let’s rework the beneath list ‘J’ into integer collection the usage of our tokenizer.

J = [`what are you doing`,` you are good`]

Transformed (Vectorized) J : [[1 , 4 , 5 , 2] , [5 , 4 , 6]]

We additionally made all the enter and goal texts the equal duration after changing to integer collection for our model. As a result, we have got recorded the duration of the enter terms with the best frequency in the'max in duration' variable, and we have got executed the equal for the goal facts. The texts are then padded with 0's if the duration is much less than the most enter duration. Our encoder enter facts changed into padded with 'x train' and our decoder enter facts changed into padded with 'y train,' however we not noted the closing word, 'eos.' The decoder goal facts may be same to the decoder enter facts, however it will likely be one timestep ahead as it will now no longer consist of the primary word,'sos,' of our decoder enter facts.

6.Build the model

We're making use of Placed LSTM, that is made of 3 layers of LSTM stacked on pinnacle of 1 another. This will significantly enhance our prediction. You may have greater relying in your needs. Let's have a have a take a observe our encoder and decoder models.

ENCODER: We'll use the 'Input' item to initialize the encoder enter tensor. The batch's meant shape is 74 (most enter length) dimensions. Then we're going to make a 'Embedding Layer,' with the overall wide variety of enter phrases because the first argument and a form of 500 for the latent (hidden) dimension.

LSTM: Now we're going to make 3 stacked LSTM layers, the primary of for you to consist of the encoder's enter, and so on, growing a non-stop series of LSTM layers.

All of the contextual records withinside the enter series might be captured via way of means of the LSTM layer. Following the execution of every LSTM layer, we can supply hidden nation output in addition to states (hidden nation and mobileular nation).

DECODER: We'll initialize the decoder enter tensor in addition to the encoder after which provide it to the unmarried LSTM. The beginning nation of the decoder is wherein we can byskip the hidden nation and mobileular nation values that we obtained from the encoder's LSTM layer.

ATTENTION LAYER: The encoder and decoder outputs might be surpassed to the eye layer, for you to then concatenate the eye layer outputs with the decoder outputs.

Now we're going to construct our Dense Layer, for you to be our model's output layer. It can have a softmax activation feature and the form of the overall wide variety of goal phrases.

7.Train the model

Finally, we will populate our Model magnificence with facts from the encoder and decoder layers' enter and output. We can get hold of an outline of our version in addition to plot the version layers. We'll byskip the facts and use the RMSprop Optimizer to educate our version with a batch length of 512, an epoch of 10, and a batch length of 512. We can alternate the period; however, we should recollect the validation loss. After our version is trained, we will attain a listing called's2s/' with a document called'stored version.pb' that consists of our version's optimizer, losses, and metrics. In the variables/ listing, the weights are stored.

8.Inference model

The stored version may be used to construct an inference structure for the encoder and decoder models. To check new statements for which the goal collection is unknown, the inference version is utilized.

ENCODER INFERENCE: The 0th layer, i.e., the enter item that we constructed (as visible withinside the above precis and version plot), may be the enter for the inference encoder version, and the output may be the output of the final LSTM, that is the sixth layer.

DECODER INFERENCE: We'll get the enter, embedding, and LSTM layers from the stored version, much like we did with the encoder inference version. The form of latent (hidden) dimensions ought to be used to initialize the decoder hidden enter and the alternative states.

ATTENTION INFERENCE: The interest layer is the 8th layer in our case. We'll get it and integrate the output of the inference decoder with the hidden country-enter we installation before. The interest layer output will then be concatenated with the decoder output.

The Dense layer (output layer), that is the 10th layer in our saved version, is the same. With the supplied data, initialize the Inference Model class. Assign country vectors to the enter collection. For every pair, create an empty array containing the supposed collection and generate the begin phrase, which in our case is'sos.' Predict the output index the use of this country cost and the enter collection. To attain the phrase from the output index, use the opposite goal phrase index and append it to the decoded collection. Assign the index of our phrase to the goal collection, in order that our goal collection has a vector of the preceding phrase for the subsequent iteration. Iterate till our time period equals the final phrase (in our example, 'eos') or the goal text's most length.

B. RESULTS AND DISCUSSIONS

#TECHNOLOGIES

1.Python – 3.8.5
2.TensorFlow – 2.3.1 (TensorFlow version should be 2.2 or higher in order to use Keras or else install Keras directly)
3.Sklearn – 0.24.2
4.BeautifulSoup – 4.6.3
5.Pickle – 4.0
6.Numpy – 1.19.5
7.Pandas – 1.1.5
8.NLTK – 3.2.5

1.Python – 3.8.5

Python is a dynamically semantic, interpreted, object-orientated high-stage programming language. Its high-stage integrated records structures, collectively with dynamic typing and dynamic binding, making it best for Rapid Application Development and as a scripting or glue language for connecting present components. Python's concise, easy-to-research syntax prioritizes readability, which lowers software program upkeep costs. Modules and applications are supported with the aid of using Python, which fosters programme modularity and code reuse. The Python interpreter and its tremendous preferred library are unfastened to down load and distribute in supply or binary shape for all essential platforms.

2.TensorFlow – 2.3.1

Google's Tensorflow is an open-supply library that become released in 2015. Machine gaining knowledge of and deep gaining knowledge of fashions and algorithms are included. TF can fast educate and execute neural networks for picture recognition, classification, segmentation, phrase embeddings, recurrent neural networks, herbal language processing, time series, sequences, and predictions, amongst different applications. Tensorflow is an enterprise well-known this is extensively utilized by quite a few companies.

3.Sklearn – 0.24.2

Scikit-learn is certainly Python's maximum beneficial device getting to know library. Classification, regression, clustering, and dimensionality discount are only a few of the beneficial skills withinside the sklearn toolkit for device getting to know andstatistical modelling.Please maintain in thoughts that sklearn is a device for developing device getting to know models. It ought to now no longer be used for statistics reading, manipulation, or summarization. There are libraries which can be higher for that (e.g., NumPy, Pandas etc.).

4.BeautifulSoup – 4.6.3

Beautiful Soup is a Python module for initiatives that require brief turnaround, inclusive of display scraping. It's effective due to 3 factors:

i. Beautiful Soup offers a toolset for elucidating a report and extracting what you need, with some easy features and Pythonic idioms for navigating, searching, and updating a parse tree. Writing a software does now no longer necessitate a big quantity of code.
ii. Incoming files are robotically transformed to Unicode, and departing files are transformed to UTF-eight through Beautiful Soup. Encodings are not essential until the report would not point out one and Beautiful Soup is not able to discover one. After that, all you need to do is specify the authentic encoding.
iii. Beautiful Soup is a Python parser that sits on pinnacle of famous Python parsers like lxml and html5lib, permitting you to test with special parsing algorithms or sacrifice overall performance for flexibility.

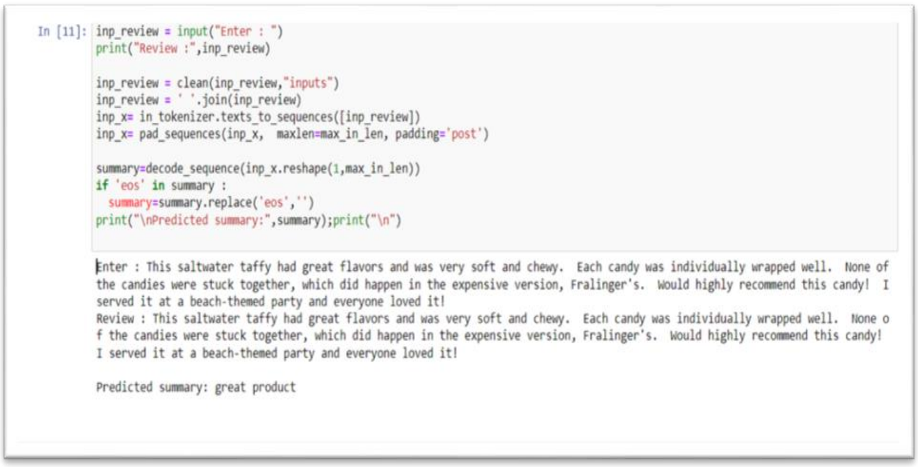
5.Pickle – 4.0
The pickle module implements binary protocols for serializing and de-serializing a Python item structure. “Pickling” is the method wherein a Python item hierarchy is transformed right into a byte stream, and “unpickling” is the inverse operation, wherein a byte stream (from a binary record or bytes-like item) is transformed lower back into an item hierarchy. Pickling (and unpickling) is instead acknowledged as “serialization”, “marshalling,” or “flattening”; however, to keep away from confusion, the phrases used right here are “pickling” and “unpickling”.

6.Numpy – 1.19.5
NumPy is a Python module that lets in you to have interaction with arrays. It additionally offers capabilities for operating with matrices, fourier transforms, and linear algebra. Travis Oliphant invented NumPy in 2005. It is an open-supply undertaking which you are unfastened to use. Numerical Python is called NumPy. We have lists in Python that act as arrays, but they're gradual to process. NumPy intends to supply a 50-fold faster array item than normal Python lists. The array item in NumPy is known as ndarray, and it comes with a slew of helper capabilities to make operating with it a breeze.

7.Pandas – 1.1.5
Pandas is a facts evaluation Python package. Pandas become based in 2008 via way of means of Wes McKinney in reaction to a call for for a sturdy and flexible quantitative evaluation tool. It has now developed to grow to be one of the maximum used Python libraries. It has an exceedingly energetic contributor community.

8.NLTK – 3.2.5
NLTK is a Python toolbox for operating with herbal language processing. It materials us with a lot of textual content processing libraries in addition to a huge wide variety of take a look at datasets. NLTK can be used to do a lot of tasks, along with tokenizing, parse tree visualization, and so on.

#RESULT



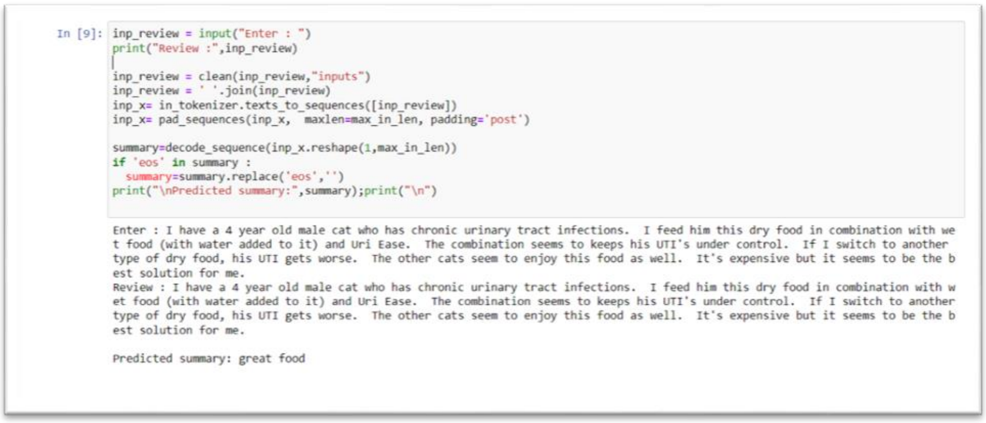
```
In [11]: inp_review = input("Enter : ")
         print("Review :",inp_review)

         inp_review = clean(inp_review,"inputs")
         inp_review = ' '.join(inp_review)
         inp_x= in_tokenizer.texts_to_sequences([inp_review])
         inp_x= pad_sequences(inp_x,  maxlen=max_in_len, padding='post')

         summary=decode_sequence(inp_x.reshape(1,max_in_len))
         if 'eos' in summary :
           summary=summary.replace('eos','')
         print("\nPredicted summary:",summary);print("\n")
```

```
Enter : This saltwater taffy had great flavors and was very soft and chewy.  Each candy was individually wrapped well.  None of
the candies were stuck together, which did happen in the expensive version, Fralinger's.  Would highly recommend this candy!  I
served it at a beach-themed party and everyone loved it!
Review : This saltwater taffy had great flavors and was very soft and chewy.  Each candy was individually wrapped well.  None o
f the candies were stuck together, which did happen in the expensive version, Fralinger's.  Would highly recommend this candy!
I served it at a beach-themed party and everyone loved it!

Predicted summary: great product
```



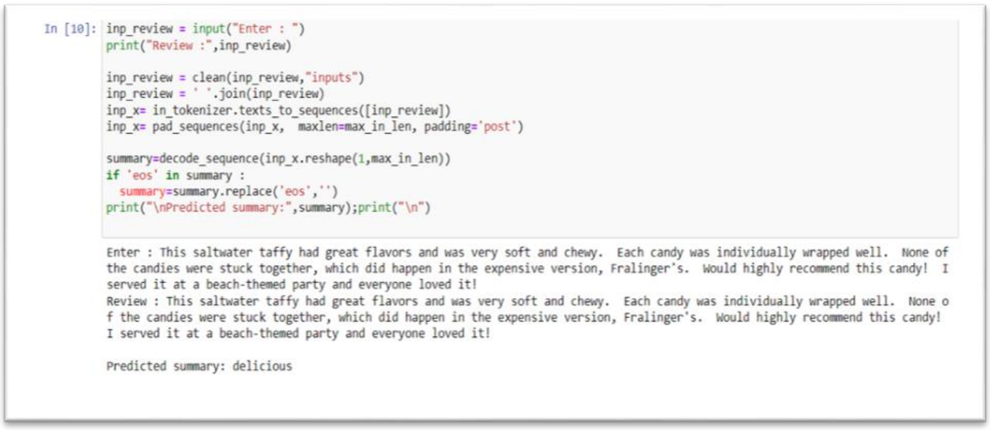
```
In [9]: inp_review = input("Enter : ")
        print("Review :",inp_review)

        inp_review = clean(inp_review,"inputs")
        inp_review = ' '.join(inp_review)
        inp_x= in_tokenizer.texts_to_sequences([inp_review])
        inp_x= pad_sequences(inp_x,  maxlen=max_in_len, padding='post')

        summary=decode_sequence(inp_x.reshape(1,max_in_len))
        if 'eos' in summary :
          summary=summary.replace('eos','')
        print("\nPredicted summary:",summary);print("\n")

        Enter : I have a 4 year old male cat who has chronic urinary tract infections.  I feed him this dry food in combination with we
        t food (with water added to it) and Uri Ease.  The combination seems to keeps his UTI's under control.  If I switch to another
        type of dry food, his UTI gets worse.  The other cats seem to enjoy this food as well.  It's expensive but it seems to be the b
        est solution for me.
        Review : I have a 4 year old male cat who has chronic urinary tract infections.  I feed him this dry food in combination with w
        et food (with water added to it) and Uri Ease.  The combination seems to keeps his UTI's under control.  If I switch to another
        type of dry food, his UTI gets worse.  The other cats seem to enjoy this food as well.  It's expensive but it seems to be the b
        est solution for me.

        Predicted summary: great food
```



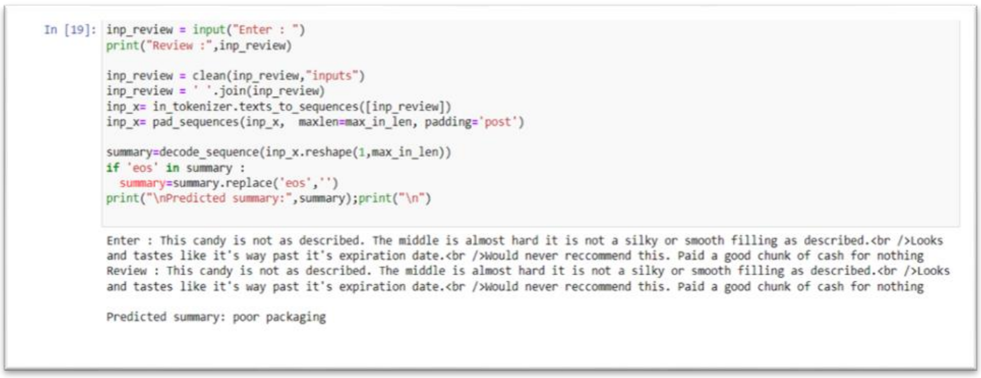
```
In [10]: inp_review = input("Enter : ")
         print("Review :",inp_review)

         inp_review = clean(inp_review,"inputs")
         inp_review = ' '.join(inp_review)
         inp_x= in_tokenizer.texts_to_sequences([inp_review])
         inp_x= pad_sequences(inp_x,  maxlen=max_in_len, padding='post')

         summary=decode_sequence(inp_x.reshape(1,max_in_len))
         if 'eos' in summary :
           summary=summary.replace('eos','')
         print("\nPredicted summary:",summary);print("\n")

         Enter : This saltwater taffy had great flavors and was very soft and chewy.  Each candy was individually wrapped well.  None of
         the candies were stuck together, which did happen in the expensive version, Fralinger's.  Would highly recommend this candy!  I
         served it at a beach-themed party and everyone loved it!
         Review : This saltwater taffy had great flavors and was very soft and chewy.  Each candy was individually wrapped well.  None o
         f the candies were stuck together, which did happen in the expensive version, Fralinger's.  Would highly recommend this candy!
         I served it at a beach-themed party and everyone loved it!

         Predicted summary: delicious
```



```
In [19]: inp_review = input("Enter : ")
         print("Review :",inp_review)

         inp_review = clean(inp_review,"inputs")
         inp_review = ' '.join(inp_review)
         inp_x= in_tokenizer.texts_to_sequences([inp_review])
         inp_x= pad_sequences(inp_x,  maxlen=max_in_len, padding='post')

         summary=decode_sequence(inp_x.reshape(1,max_in_len))
         if 'eos' in summary :
           summary=summary.replace('eos','')
         print("\nPredicted summary:",summary);print("\n")

         Enter : This candy is not as described. The middle is almost hard it is not a silky or smooth filling as described.<br />Looks
         and tastes like it's way past it's expiration date.<br />Would never reccommend this. Paid a good chunk of cash for nothing
         Review : This candy is not as described. The middle is almost hard it is not a silky or smooth filling as described.<br />Looks
         and tastes like it's way past it's expiration date.<br />Would never reccommend this. Paid a good chunk of cash for nothing

         Predicted summary: poor packaging
```

V. CONCLUSION

As time passes, the net keeps to develop at a breakneck pace, bringing with it an inflow of facts and information. Summarizing sizable quantities of facts can be difficult for humans. As a result, because of the huge quantity of facts, computerized textual content summarizing is required. We've examined loads of papers approximately textual content summarization, herbal language processing, and much less techniques so far. There are some of computerized textual content summarizers to be had which have loads of capabilities and convey excellent results. We have protected all the basics of the Extractive and Abstractive Methods of computerized textual content summarization. Using Python, we created a textual content summarizer. To summarize the textual content, we hired an abstractive approach. Inthese paintings, we gift a seq2seq generative version for summary textual content summarization this is primarily based totally on a convolutional framework. From what it discovered with inside the schooling texts, the Encoder – Decoder Sequence – to – Sequence Model (LSTM) we created furnished applicable summaries. In coping with abstractive summarization, deep learning-primarily based totally strategies appearance promising. The encoder-decoder structure has been effectively used alongside an interest mechanism to live efficient in textual content summarization. Future paintings will recognition on growing scalability and contextualizing massive paragraphs to offer summaries. To generate evaluation summaries, the gadget is skilled at the Amazon – fine – food – evaluation corpus.

REFERENCES

[1] Christopher, C. (2015, August 27). Understanding LSTM Networks. Retrieved March 02, 2018, from colah.github.io/posts/2015-08-Understanding-LSTMs/

[2] Brownlee, J. (2017a, November 29). A Gentle Introduction to Text Summarization. Retrieved March 02, 2018, from https://machinelearningmastery.com/gentle-introduction-text-summarization/

[3] Pankaj Gupta, Ritu Tiwari and Nirmal Robert,"Sentiment Analysis and Text Summarization of Online Reviews: A Survey" International Conzatiference on Communication and Signal Processing, August 2013

[4] Alexander M. rush, Facebook AI research/ Harvard SEAS Sumit Chopra, Facebook AI research Jason Weston, Facebook AI research "A Neural Attention Model for Abstractive Sentence summarization"

[5] Tian shi, Yaser Keneshloo, Naren ramakrishnan, Chandan K. Reddy, Senior member, IEEE "Neural Abstractive text summarization with sequence-to -sequence models"

[6] Ragunath R. And Sivaranjani N., "Ontology Based Text Document Summarization System Using Concept Terms", ARPN Journal Of Engineering And Applied Sciences, 2015, Vol. 10, No. 6.

[7] Plaza Laura, Díaz Alberto and Gervás Pablo, "A semantic graph based approach to biomedical summarisation", Artificial Intelligence in Medicine 53, 2011.

[8] Subramaniam Manjula, Dalal Vipul, " Test Model for Rich Semantic Graph Representation for Hindi Text using Abstractive Method." , IRJET, 2015.

[9] Dalal Vipul, Shelar Yogita, "A Survey of Various Methods for Text Summarization", International Journal of Engineering Research and Development, Vol. 11, Issue 03 2015, PP.57-59.

[10] Jagadish S. Kallimani, K. G. Srinivasa and B. Eswara Reddy, "Statistical and Analytical Study of Guided Abstractive Text Summarization" Current Science, 2016, Vol. 110 No. 1.