Improvised Clarity-cuts Text Summarization with Text to Graph Prediction

Geeta Patil[1], Aditi chougule[2], Kshama Jadhav[3], Mayuri toralkar[4]

[1]Student, Computer Science Engineering, S G Balekundri Institute of technology Belagavi, Karnataka, India
[2]Student, Computer Science Engineering, S G Balekundri Institute of technology Belagavi, Karnataka, India
[3]Student, Computer Science Engineering, S G Balekundri Institute of technology Belagavi, Karnataka, India
[4]Student, Computer Science Engineering, S G Balekundri Institute of technology Belagavi, Karnataka, India

ABSTRACT

Text summarization is a natural language processing task aimed at reducing a given document or text into a condensed, coherent summary. It entails taking out or generating the most critical information from the source text while preserving its key ideas and meaning. Text summarization can be categorized into two main types: Extractive and Abstractive. Extractive summarization entails choosing and merging key phrases or the sentences from the source text to form an overview. This method relies on finding the most pertinent and informative sentences based on criteria like sentence position, significance, and similarity to other sentences in the text. Unlike abstractive synthesis, which entails creating new sentences, extractive summarization utilizes the existing content without generating new text. This project presents the development and evaluation of Text2Chart, a machine learning model designed for converting textual descriptions to graph charts. In this we compared the performance of four different classification models: Multinomial Naive Bayes, Random Forest classifier, Linear SVM, and Logistic Regression. The dataset employed in training and evaluation consisted of textual descriptions with corresponding chart types. Consequently, Linear SVM was chosen as the algorithm for training the Text2Chart model to predict chart types, including bar chart, line chart, pie chart, and box plot. Abstractive summarization aims to overview that captures the essence of the initial manuscript through natural language generation techniques. Unlike extractive summarization, which selects and merges existing sentences or phrases, abstractive overview involves creating new sentences that may not appear in the source document. This method requires a deeper understanding of the input text and often includes paraphrasing and rephrasing the content produce a coherent and concise summary.

Keyword : - *Automatic text summarization, attention mechanism, Stemming, Transform Model, syntactic parsing, Graph Generation.*

---

1. Introduction

In a number of domains where massive volumes of text need to be processed fast and effectively, including news aggregation, document summarization, and search engine result summarization, text summarization is the process of reducing a document's content into a shorter, more concise version while maintaining its essential points and main ideas. Machine learning, particularly techniques within natural language processing (NLP), has revolutionized text summarization by enabling automatic extraction and generation of summaries. Two main approaches are commonly used in machine learning-based text summarization: extractive and abstractive.

1. Summarization: When a model performs extractive summarization, it finds the most significant sentences or phrases in the source text and uses those to create a summary. This method is predicated on the idea that the most important details are already included in the original text. Sentences are frequently ranked using a variety of factors, including word frequency, sentence position, and semantic similarity, in extractive summarization algorithms.

2. Summarization: Conversely, abstractive summarization creates new sentences that encapsulate the main ideas of the source material. This method is more difficult since it calls for the model to produce summaries that are human-like and comprehend the text's content. Advanced natural language processing models like transformers, which may provide coherent and contextually relevant summaries, are frequently used in abstractive summarizing techniques. Machine learning models for text summarization are typically trained on large datasets of text-summary pairs. These models learn to identify important information and generate summaries based on the patterns and structures present in the training data. Evaluation of these models is often

done using metrics such as ROUGE (Recall-Oriented Understudy for Gisting Evaluation), which measures the overlap between the generated summary and a set of reference summaries.
Overall, machine learning-based text summarization holds great promise for greatly enhancing the effectiveness of processing and comprehending massive amounts of text, with applications ranging from information retrieval and research to journalism. abstractive summary, which pulls out key words or phrases from the source material.

By automating the chart selection process, the overall goal of this project is to make a substantial contribution to the field of data visualization. In addition to saving time, an accurate Text2Chart model will let users make efficient use of their textual data to produce charts that are both visually beautiful and educational. The growing availability of data is driving up demand for effective and intuitive approaches to data interpretation and analysis. Charts and graphs, among other visualizations, are essential for facilitating a more intuitive and clear comprehension of data, making it simpler to extract insights and spot patterns [1]. With the use of machine learning algorithms, Text-to-Chart Conversion provides a potent solution by analyzing textual input, extracting relevant information, and then mapping that information to the proper chart type. By streamlining the visualization generation process, this automated method saves time and minimizes the need for manual labor [3]. Through the utilization of machine learning, Text-to-Chart Conversion provides a more effective and user-friendly method for analyzing and interpreting data. It does away with the requirement that users have sophisticated knowledge of data visualization because the technology converts words into visuals that help users understand the data on a deeper level[1].
In summary, Text-to-Chart Conversion uses machine learning algorithms to automate the transformation of textual data into visually appealing and informative charts. This innovation enhances the efficiency and accessibility of data analysis, allowing users to gain valuable insights quickly and effortlessly.

1.1 Problem statement

The goal of text summarization is to reduce lengthy of text into shorter, more manageable summaries, but current techniques often struggle with accuracy. This project aims to develop an advanced system using natural language processing and machine learning to create highquality summaries. Additionally, the project aims to develop a code that utilizes machine learning to automate the conversion of textual data into visualizations such as charts and graphs, enhancing data analysis and interpretation [1]. Several challenges must be addressed to accomplish this:

1. **Understanding the text data**: The model needs to effectively process and extract meaningful information from the textual data provided [2].
2. **Mapping text data to chart types**: The model should determine the most suitable chart type that represents the underlying data. By analyzing the extracted information, it can decide whether a bar chart, pie chart, line chart, or box plot is most appropriate [3].
3. **Pre-processing and cleaning of the data**: The model has to deal with a number of data problems, such as inconsistent data, missing values, and inaccurate entries. To make sure the data is in a format that can be used for visualization, pre-processing procedures must be carried out [2].
4. **Automating the creation of charts**: The model ought to produce excellent, aesthetically pleasing charts and graphs on its own. To successfully express insights and patterns within the data, it must reliably transform the extracted data into visual components, including as axes, labels, colors, and legends [1].
5. **Increasing accuracy and resilience**: The model needs to be able to manage a variety of data inputs, such as text that is both organized and unstructured, and it needs to be resilient in the face of intricate data circumstances. Accuracy and generalization skills may be enhanced by strong feature extraction methods, sophisticated algorithms, and comprehensive training on a variety of datasets. By tackling these challenges, the envisioned machine learning model can facilitate the automated conversion of textual data into visualizations, empowering users to analyze data more easily and derive valuable insights from complex datasets.

1.2 Objectives

The main objectives of our project are as follows:

1. Text Summarization: Utilize machine learning to reduce the size of text while preserving its important information and overall meaning.
2. Enhancing Machine Understanding: Extract the most important features of text to enhance machine understanding and facilitate further analysis or processing.
3. Improving Accessibility: To make information more usable and accessible, clearly and succinctly present the text's main ideas.
4. Create Text2Chart: Text2Chart is a machine learning model that turns textual data into graph charts according to the types of charts that correspond to it (e.g., box plot, pie chart, line chart, and bar chart).
5. Algorithm Evaluation: To ascertain the most accurate method for forecasting chart types, compare and assess the performance of four classification algorithms: Random Forest Classifier, Multinomial Naive Bayes, Linear SVM, and Logistic Regression.
6. Pattern Recognition: To create a relationship between descriptions and certain chart kinds, look for and recognize recurring patterns in textual descriptions.

7. Make Use of Spacy: To enhance charts by extracting significant characteristics from textual data, utilize Spacy, a natural language processing (NLP) model.
8. Graph Chart Generation: Implement functions and techniques to generate graph charts based on predicted chart types.

2. System Design

The process of establishing a system's architecture, parts, modules, interfaces, and data in order to meet predetermined criteria is known as system design. Creating a solution design from the requirements is an essential phase in the software development process.

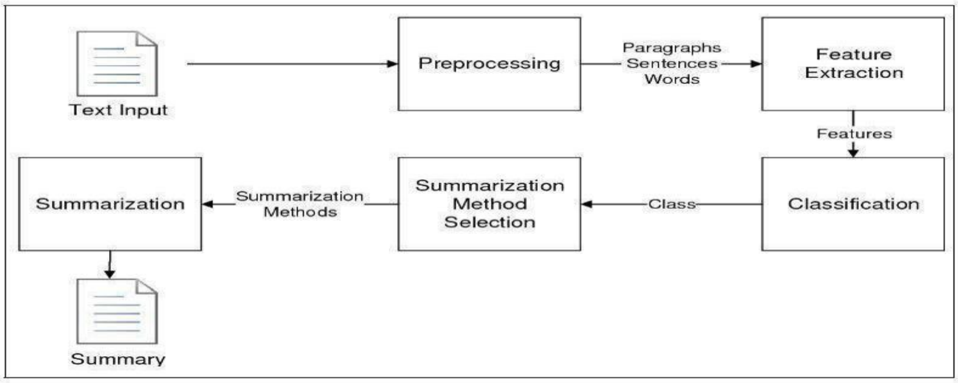


Chart -1: Class and Object diagram

The goal of abstractive text summarization is to provide a brief synopsis that highlights the key concepts from the original text. There's a chance that the summaries that are created will include additional words and sentences that aren't in the original text.
1. Preprocessing - Preprocessing all the texts and converting them into presentable form.
2. Splitting the records - Split the dataset records into training and testing sets. We will be splitting in the 80:20 ratio where 80% record will be for training sets and 20% for testing sets.
3. Text Vectorization - We will convert our word into integer sequence using vectorization technique.
4. Build the model - We are using 3 Layers of LSTM. This will make our prediction much better. We will build it using an encoder and decoder model.
5. Train the model - We will initialize our model class with output and input data from the encoder and decoder layers.
6. Inference Model - The inference model is used to test the new sentences for which the target sequence is not known.
7. Sentence selection - Set a threshold or select a specific number of sentences to include in the summary based on their scores. You can experiment with different thresholds or sentence limits to generate summaries of varying lengths.
8 Summary generation - Arrange the selected sentences in a logical order to form a coherent summary. Optionally, post-process the summary to ensure grammatical correctness and improve readability.

ARCHITECTURE: There is a research paper “Attention Is All You Need”. This paper introduces a novel architecture, namely ‘Transformer’. This new model uses the attention mechanism. Similar to LSTM, we use it to transform one sequence to another by following the encoder-decoder structure.

The transformer model does not need to rely on recurrence or convolution for generating outputs. As mentioned earlier, the transformer model works on two concepts: Encoder and Decoder.

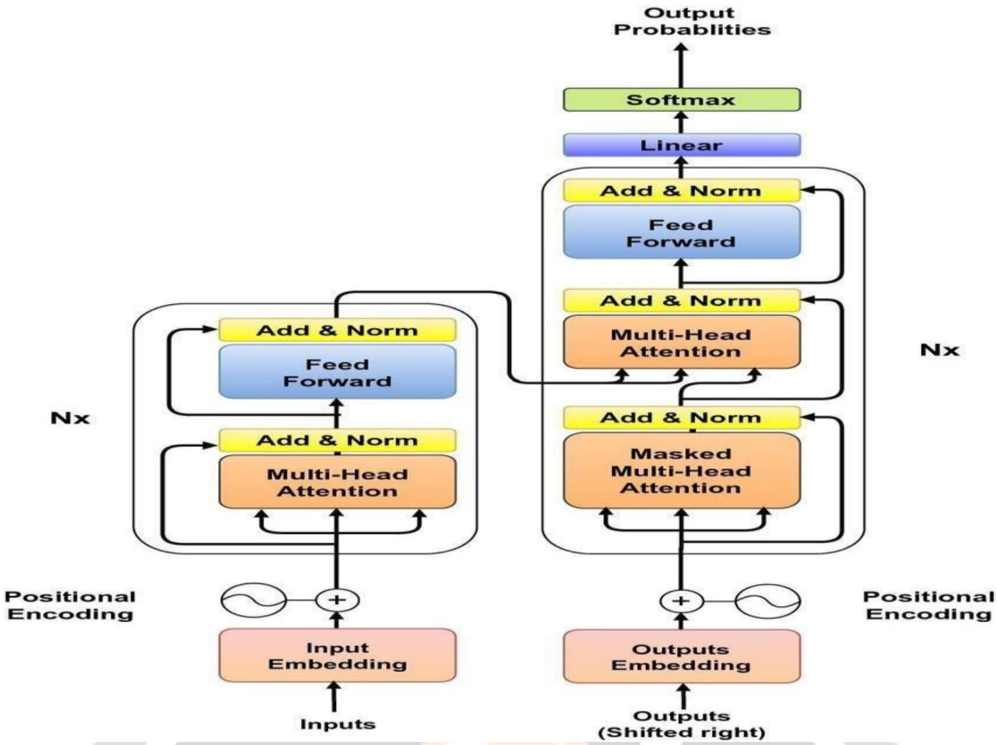


Chart -2: Architecture Diagram

1. Encoder: N = 6 layers, each with two sublayers, make up the encoder, which is on the left side of the architecture:
    - The first sublayer implements a multi-headed self-attention mechanism, where h heads receive linearly projected versions of queries, keys, and values. This results in h outputs in parallel, which are then concatenated and linearly transformed to obtain the final output.
    - The second sublayer is a fully connected feed-forward neural network, consisting of two linear transformations with a ReLU activation function in between.

Every word in the input sequence is subjected to the identical linear alterations by every encoder layer. Both sublayers contain a residual connection after which comes a normalization layer that normalizes the total of the input and output of each sublayer. Each layer has distinct weights and biases.

2. Decoder: The architecture's right-side decoder, or N = 6 layers, is also made up of three sublayers per layer:

• The first sublayer receives the previous output of the decoder stack, augments it with positional information, and implements multiread self-attention over it. Unlike the encoder, the decoder attends only to preceding words in the sequence, achieved through masking in the multi-headed attention mechanism.

• The second sublayer implements the same multi-headed self-attention mechanism as the encoder's first sublayer. However, it receives queries from the previous decoder sublayer and keys/values from the output of the encoder, allowing the decoder to attend to all words in the input sequence.

• The third sublayer is a fully connected feed-forward network, similar to the second sublayer of the encoder.

Both the encoder and decoder play crucial roles in the Transformer architecture, with the encoder processing the input sequence and the decoder generating the output sequence. The attention mechanisms in both components allow for effective modelling of dependencies between words in the input and output sequences, making Transformers highly effective for various natural language processing tasks.1

2.1 Literature Review

The following table shows the literature revies of the previously published papers on text summarization with text to graph prediction.

Table -1: Literature review Table

SL.NO	Author Name	Title	Observations

1	Jingwei cheng , fu zhang , and xuyang guo	A Syntax-Augmented and Headline-Aware Neural Text Summarization Method	They propose several improvements that address critical problems in summarization that are not adequately modeled by the basic Seq2Seq framework.We propose a syntax-augmented encoder and a headline-aware decoder.
2	Jiawen jiang, haiyang zhang , chenxu dai, qingjuan zhao , hao feng.	Enhancements of Attention-Based Bidirectional LSTM for Hybrid Automatic Text Summarization	This paper has put forward enhancements to the structure of the attention-based bi-directional LSTM model (‘Bi-LSTM + Attention’) and the attention-based sequence model (‘Seq2Seq + Attention’) in order to improve the Automatic Text Summarization.It has been proposed to combine the elaborated ‘DA-PN’ model with a coverage mechanism integrating multi-attention
3	Muhammad,yahya saeed1,2,muhammad awais 2 , ramzan talib	Unstructured Text Documents Summarization With Multi-Stage Clustering	The current study’s contribution is a technique to overcome the diversity of unstructured text with reduced cost, as the proposed technique provides a language-structure independent mechanism. First, this technique ignores text-summary & sentence positions. It may cause less reading ease for generated summaries.
4	Ángelhernández-castañeda , rené arnulfo garcía-hernández , and yulia ledeneva	Toward the Automatic Generation of an Objective Function for Extractive Text Summarization	This study proposes the automatic generation of an objective function for the unsupervised text summary task. A combination of a genetic algorithm and genetic programming was performed to build a maximization function that maintains a close correlation with the quality of the summaries.
5	Kaichun Yao, Libo Zhang , Dawei Du , Tiejian Luo, Lili Tao, and Yanjun Wu	Dual Encoding for Abstractive Text Summarization	In this paper, they present a dual encoding model which extends the sequence-to-sequence framework for abstractive text summarization. This model is built on a basic encoder– decoder model with attention mechanism, the PM and the RAM. Different from the encoder–decoder model, the dual encoding model decodes the whole output sequence by stages and produces the partial fixed-length sequence at each stage.
6	Haozhou Li , Qinke Peng , Xu Mou , Ying Wang , Zeyuan Zeng , and Muhammad Fiaz Bashir	Abstractive Financial News Summarization via Transformer-BiLSTM Encoder and Graph Attention-Based Decoder	They propose an enhanced Seq2Seq model TLGA-FNS for financial news summarization. Unlike traditional encoder decoder models, our Transformer-BiLSTM encoder captures both long-range interactions and sequential information in financial news and alleviates the long-term
7	Jesus M. Sanchez-Gomez , Miguel A. Vega-Rodríguez , and Carlos J. Pérez	Automatic Update Summarization by a Multiobjective Number-One-Selection Genetic Approach	It involves the handling of dynamic document collections.The update summarization problem will be addressed in conjunction with sentiment analysis. In this way, the sentiment-oriented update summarization problem will generate sentiment-oriented update summaries, which will take into account both the relevant new information delivered to the user and the sentiment orientation of the sentences included in it.

8	Li Huang , Wenyu Chen, Yuguo Liu, Shuai Hou, and Hong Qu , Member, IEEE	Summarization With Self-Aware Context Selecting Mechanism	In this work, they improved the standard attention-based encoder decoder framework with an asynchronous bi directional contextual information encoder and a self-aware context selecting decoder for summarization task.The samples of summaries generated from our model, showing validity and diversity, manifested the potential of the proposed approach.

2.2 Implementation

Text summarization using a transformer model like BERT or GPT involves several key steps.
Here's a detailed outline of the algorithm:

- Preprocessing: Tokenize the input text and add special tokens ([CLS] and [SEP] for BERT) to mark the beginning and end of the text. Convert the tokens to IDs using the model's vocabulary.
- Encoding: Use the pre-trained transformer model to encode the input text. For BERT, this involves feeding the token IDs into the model and obtaining contextual embeddings for each token. For GPT, this involves generating a sequence of embeddings autoregressively.
- Summarization Head: Add a summarization head on top of the transformer model. This can be a linear layer for extractive summarization (selecting important sentences) or a decoder for abstractive summarization (generating a summary).
- Fine-Tuning (Optional): Fine-tune the model on a summarization dataset to improve its performance on the summarization task. This step is particularly important for abstractive summarization.
- Generation: Choose the best-scoring phrases for extractive summarization according to certain standards (e.g., significance score). Utilize the model to provide an abstractive summary depending on the input text's encoding.
- Decoding: Decode the generated summary from token IDs back to text using the model's vocabulary.
- Post-processing: Clean up the generated summary (e.g., remove special tokens, punctuation) to make it more readable.
- Output: The final output is the generated summary of the input text.

This algorithm can be implemented using libraries like Hugging Face's Transformers in Python, which provides easy access to pre-trained transformer models and tools for fine-tuning and inference.

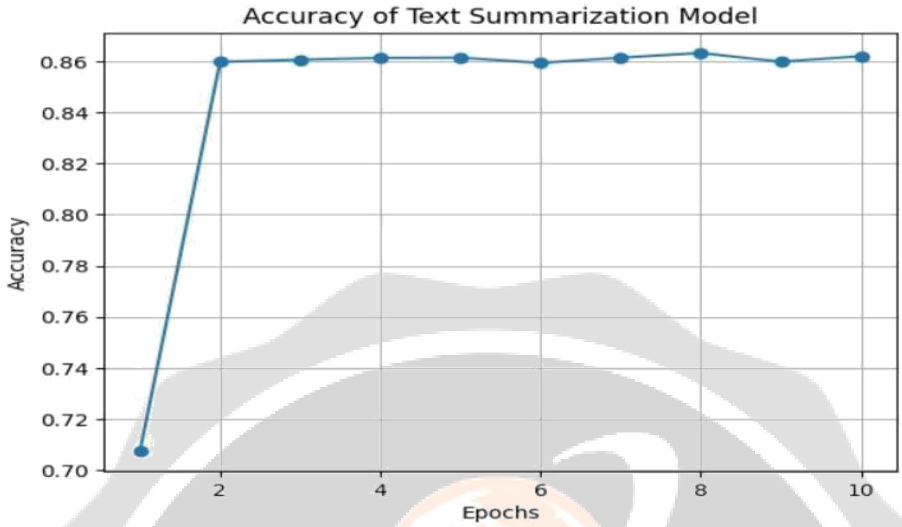
Here are the algorithm steps for text-to-graph prediction:

- Text Preprocessing: Tokenize the input text and perform any necessary cleaning and normalization steps.
- Entity Recognition: Use Named Entity Recognition (NER) to identify entities (e.g., people, organizations, locations) in the text.
- Relation Extraction: Extract relationships between entities mentioned in the text (e.g., "works for," "located in").
- Graph Construction: Build a graph representation where entities are nodes and relationships are edges. Each node is associated with its entity type (e.g., person, organization) and each edge is associated with its relationship type.
- Graph Embedding: Transform the graph into a numerical representation so that a machine learning model may utilize it as input. This might include using methods like graph convolutional networks and node embedding.
- Machine Learning Model: Use a machine learning model (e.g., graph neural network) to predict the graph structure based on the input text and the embedded graph.
- Graph Prediction: The model predicts the graph structure based on the input text and the embedded graph.
- Post-processing: Refine the predicted graph if necessary (e.g., resolving conflicts, adding missing nodes/edges).
- Output: The final output is the predicted graph representing the relationships between entities mentioned in the input text.
- Evaluation: Evaluate the predicted graph against a ground truth graph or other metrics to assess its accuracy.

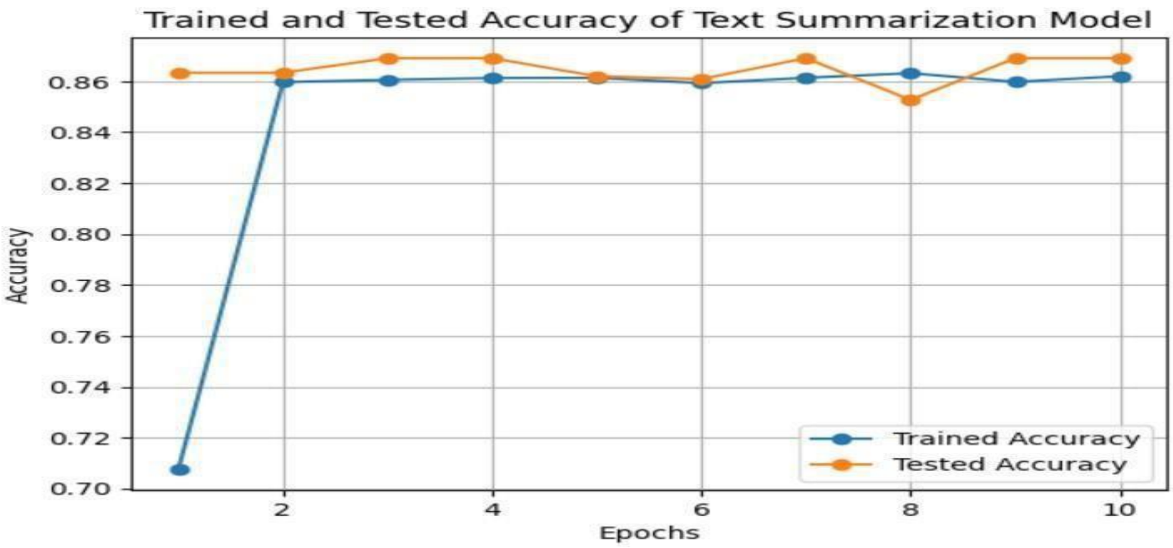
3. Result and Analysis

It appears that you have given an overview or synopsis of your undertaking. You might reword the material in your own terms while keeping the essential details to lessen plagiarism.
This is an updated version:

For our project, we created a Text Summarizer model that generates summaries from reviews by utilizing LSTM and Attention Mechanism. We made advantage of the 100,000-row Amazon food review dataset. The dataset was divided into two groups: training (80%) and testing (20%). We employed the ROUGE (Recall-Oriented Understudy for Gisting Evaluation) criteria to assess the quality of the summaries. In comparison to previous models, our model performed better, demonstrating its efficacy in producing clear and succinct summaries. However, because of its complexity, abstractive summarization generally takes more time and resources.



Chat 3.1:- Accuracy of Text Summarization



Chat 3.2:- Trained And Tested Accuracy

The Text2Chart project aimed to create a machine learning model for converting text data into graph charts using a multiclass classification approach [4]. The project was conducted in a systematic manner, involving stages such as data collection, model development, and graph creation.

For data collection, a diverse dataset was collected from Google, and each text sample was labeled according to its corresponding chart type. This ensured the availability of sufficient examples for each chart type and facilitated effective learning by the model[1]

```
[14] cv_df.groupby('model_name').accuracy.mean()

model_name
LinearSVC                 0.777647
LogisticRegression        0.762353
MultinomialNB             0.758824
RandomForestClassifier    0.738824
Name: accuracy, dtype: float64
```

Chat 3.3 :- Model Comparison

```
[19] print('Test accuracy:', accuracy)

Test accuracy: 0.7823529411764706
```

Chat 3.4: Model's Test Accuracy

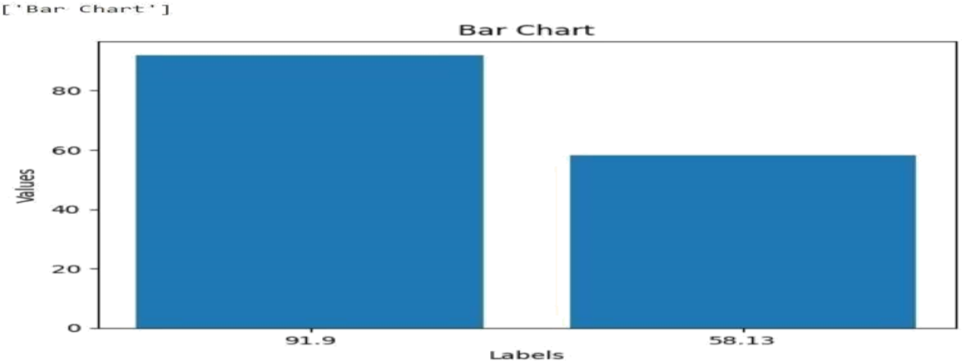
```
# Classification report
print('CLASSIFICATIION REPORT')
print(metrics.classification_report(y_test, y_pred,
                                    target_names= df['Class'].unique()))

CLASSIFICATIION REPORT
              precision    recall  f1-score   support

  Line Chart       0.65      0.42      0.51        26
   Bar Chart       0.82      0.90      0.86       186
   Pie Chart       0.33      0.23      0.27        30
    Box Plot       0.97      1.00      0.99        39

    accuracy                           0.80       281
   macro avg       0.69      0.64      0.66       281
weighted avg       0.78      0.80      0.78       281
```

Chat 3.5: Classification Report

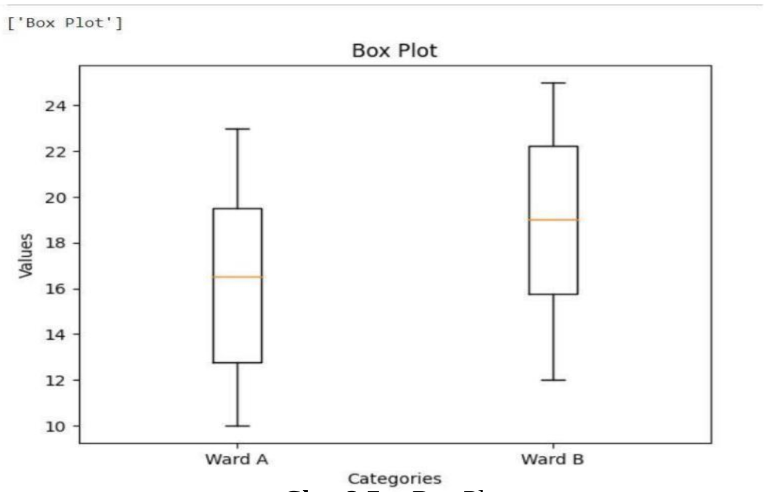
While the project successfully generated graphs for simple and general pattern sentences, there were challenges with complex sentence structures. Further research and improvements were identified for handling complex data and refining the graph generation process.

In summary, the Text2Chart project successfully developed a machine learning model capable of converting text data into graph charts. The model achieved an accuracy rate of 78%, effectively classifying text samples into suitable chart types. While the initial results were promising, future efforts will focus on overcoming challenges related to complex data structures and further refining the overall process

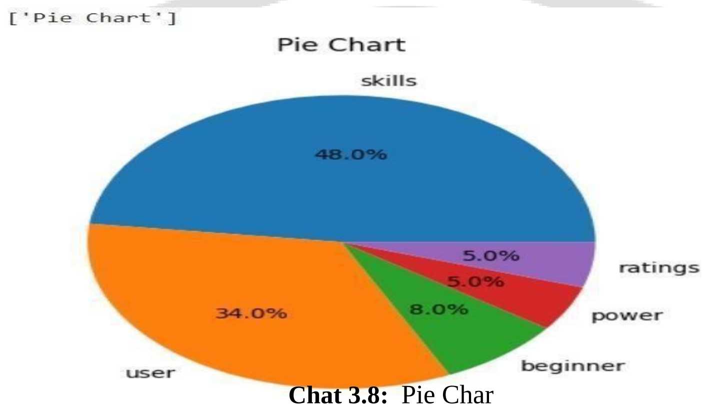
```
['Bar Chart']
```



Chat 3.6:- Bar Char



Chat 3.7 :- Box Plot



Chat 3.8: Pie Char

3.1 Future Scope

Text summarization is a rapidly growing in many fields, and there is enormous potential and future development. Here are some areas of growth of it:

1. Domain-specific summarization: Different industries and fields have their own unique language and terminology, making it challenging to summarize text accurately. There is an opportunity for text summarization systems to be developed that can summarize text for specific industries or domains.
2. Personalized summarization: Individuals have unique preferences and interests, and there is an opportunity for text summarization systems to be developed that can summarize text based on a person's interests and preferences.
3. Summarization of multimedia content: With the explosion of multimedia content, there is a growing need for text summarization systems that can summarize audio, video, and image content.
4. Real-time summarization: As the world becomes more fast-paced, there is a growing need for real-time text summarization systems that can quickly summarize text as it is generated.
5. Graph Generation for Complex Sentences: Currently, the project focuses on generating graphs for simple and general pattern sentences. However, complex sentence structures pose challenges in extracting the necessary data and generating accurate graphs. Future research can delve into developing more advanced algorithms and techniques to handle.
6. Handling Ambiguous Statements: The project may encounter difficulties when dealing with ambiguous statements or sentences that can be interpreted in multiple ways. Further research can explore methods to disambiguate such statements and improve the accuracy of chart type classification and subsequent graph generation.
7. Real-Time Data Visualization: Integrating the Text2Chart model with real-time data sources, such as streaming data or live social media feeds, would enable the model to process and convert textual information into dynamic and up-to-date graph charts. This could be valuable in applications such as real-time analytics, social media monitoring, and financial market analysis.
8. Deployment and Integration: The Text2Chart model can be deployed as a web application or integrated into existing data analysis platforms.

4. CONCLUSIONS

In summary, abstractive text summarizing is a useful method in natural language processing that enables the creation of succinct and logical summaries by assimilating the original text's context and meaning. It provides versatility and the capacity to produce summaries that go beyond straightforward sentence extraction.

Recent advancements in abstractive summarization, particularly with transformer based models, have shown great progress. These models utilize large-scale pretraining and fine-tuning to produce summaries that capture the essence of the source text while maintaining coherence and fluency.

However, challenges remain, particularly in ensuring the factual accuracy of generated summaries and maintaining coherence with the original text's intent. Ongoing research aims to improve summary quality by incorporating domain-specific knowledge and addressing issues with long document summaries and diverse text types.

Despite these challenges, abstractive text summarization has vast potential in applications such as news summarization, document summarization, and content summarization for chatbots and virtual assistants. It enables quick comprehension of lengthy texts, aids in information retrieval, and enhances overall efficiency in processing large volumes of textual data.

Regarding the Text2Chart project, it successfully developed a machine learning model for converting text data into graph charts, achieving an accuracy rate of 78%. The project demonstrated initial success in graph generation, with future work focused on handling

complex sentence structures and refining the overall process. This project contributes to data visualization and lays the groundwork for further advancements in text-to-chart conversion."

5. ACKNOWLEDGEMENT

It is my proud privilege and duty to acknowledge the kind help and guidance received from several people in preparation of this paper. It would not have been possible to prepare this paper in this form without valuable suggestions, cooperation and guidance.

I would like to express my special appreciation and thanks to my Project **Guide Prof. Neha Mysore**, Department of Computer Science and Engineering, SGBIT, Belagavi for excellent guidance, caring and patience. I have benefited greatly from her wealth of knowledge and meticulous editing.

My sincere gratitude to Guide **Prof. Sushant Mangasuli**, Project Coordinator, Department of Computer Science and Engineering, for his valuable suggestions in publishing this paper.

My sincere thanks to **Dr. B.S. Halakarnimath**, Professor and Head, Department of Computer Science and Engineering for his valuable suggestions and guidance.

I wish to record my sincere gratitude to Management, and **Dr. B R Patagundi**, Principal, S.G.Balekundri Institute of Technology for the permission provided to accomplish this opportunity.I am extremely grateful to faculty members of Department of Computer Science and Engineering, friends and family members for their support and encouragement in successful completion of this paper publication.

Place: Belagavi
Date:21/05/2024

Author's name
Ms. Geeta Patil (2BU20CS014)
Ms. Aditi Chougule (2BU21CS400)
Ms. Kshama Jadhav (2BU21CS409)
Ms. Mayuri Toralkar (2BU21CS411)

6. REFERENCES

[1] Shaw, P., Massey, P., Chen, A., Piccinno, F., & Altun, Y. (2019). "Generating logical forms from graph representations of text and entities". arXiv preprint arXiv:1905.08-407. [2] A. Balaji, T. Ramanathan, and V. Sonathi. "Chart-text: A fully automated chart image descriptor." arXiv preprint arXiv:1812.10636, 2018.

[2] Weiwei Cui, Xiaoyu Zhang, Yun Wang, He Huang, Bei Chen, Lei Fang, Haidong Zhang, Jian-Guan Lou, and Dongmei Zhang. "Text-to-viz: Automatic generation of infographics from proportion-related natural language statements". IEEE transactions on visualisation and computer graphics, 26(1):906–916, 2019.

[3] A. Nenkova and K. McKeown, "A survey of text summarization techniques," Mining text data, pp. 43–76, 2012.

[4] S. Alhojely and J. Kalita, "Recent progress on text summarization," in 2020 International Conference on Computational Science and Computational Intelligence (CSCI). IEEE, 2020, pp. 1503–1509.

[5] V. Dehru, P. K. Tiwari, G. Aggarwal, B. Joshi, and P. Kartik, "Text summarization techniques and applications," in IOP Conference Series: Materials Science and Engineering, vol. 1099, no. 1. IOP Publishing, 2021, p. 012042.

[6] E. Lloret, "Text summarization: an overview," Paper supported by the Spanish

Government under the project TEXT-MESS (TIN2006-15265-C06-01), 2008. 34

[7] N. K. Sirohi, M. Bansal, and S. Rajan, "Text summarization approaches using machine learning & lstm," Revista Geintec-Gestao Inovacao E Tecnologias, vol. 11, no. 4, pp. 5010– 5026, 2021.

[8] R. Belkebir and A. Guessoum, ''A supervised approach to arabic text summarization using AdaBoost,'' in New Contributions in Information Systems and Technologies. Cham, Germany: Springer, 2015, pp. 227–236

[9] N. Shafiq, I. Hamid, M. Asif, Q. Nawaz, H. Aljuaid, and H. Ali, "Abstractive text summarization of low-resourced languages using deep learning," PeerJ Computer Science, vol. 9, p. e1176, 2023.

[10] D. Alahmadi, A. Wali, and S. Alzahrani, "Taam: Topic-aware abstractive arabic text summarisation using deep recurrent neural networks," Journal of King Saud UniversityComputer and Information Sciences, vol. 34, no. 6, pp. 2651–2665, 2022.

[12] S.Kolhe et al., "Hybrid model for abstractive text summarization," in Proceedings of the 4th International Conference on Advances in Science & Technology (ICAST2021), 2021.

[13] Y. Jaafar and K. Bouzoubaa, "Towards a new hybrid approach for abstractive summarization," Procedia computer science, vol. 142, pp. 286–293, 2018.

[14] T.-M. Chang and W.-F. Hsiao, "A hybrid approach to automatic text summarization," in 2008 8th IEEE International Conference on Computer and Information Technology. IEEE, 2008, pp. 65–70.

[15] Dae Hyun Kim, Enamul Hoque, and Maneesh Agrawala. "Answering questions about charts and generating visual explanations". In Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems, pages 1–13, 2020.

[16] Qipeng Guo, Zhijing Jin, Xipeng Qiu, Weinan Zhang, David Wipf, and Zheng Zhang. Cyclist."Unsupervised graph-to-text and text-to-graph generation via cycle training ". arXiv preprint arXiv:2006.04702, 2020.

[17] Jason Obeid and Enamul Hoque. "Chart-to-text: Generating natural language descriptions for charts by adapting the transformer model". arXiv preprint arXiv:2010.09142, 2020.

[18] Yang Liu and Mirella Lapata. "Text summarisation with pre-trained encoders ". arXiv preprint arXiv:1908.08345, 2019.

[19] Sutirtha Ghosh, Prasenjit Mukherjee, Baisakhi Chakraborty, and Rezaul Bashar. "Automated generation of er-diagram from a given text in natural language". In 2018 International Conference on Machine Learning and Data Engineering (iCMLDE), pages 91– 96. IEEE, 2018.

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pretraining of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

[21] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. "Attngan: Fine-grained text to image generation with attentional generative adversarial networks ". In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1316–1324, 2018.

[22] R. Paulus, C. Xiong, and R. Socher, ''A deep reinforced model for abstractive summarization,'' 2017, arXiv:1705.04304. [Online]. Available: http://arxiv.org/abs/1705.04304

[23] M. Faruqui, Y. Tsvetkov, P. Rastogi, And C. Dyer, "Problems with evaluating word embeddings using word similarity tasks"., arXiv preprint arXiv:1605.02276, (2016).

[24] Paul Sheridan and Mikael Onsjö. "A hypergeometric test interpretation of a common tf-idf variant ". 2020. arXiv: 2002.11844.

[25] R. E. Prasojo, M. Kacimi, and W. Nutt, "StufIE: Semantic tagging of unlabeled facets using fine-grained information extraction," in Proc. 27th ACM Int. Conf. Inf. Knowl. Manage., 2018, pp. 467–476.

[26] Joeran Beel et al. "Research-paper recommender systems: a literature survey". In: International Journal on Digital Libraries 17.4 (2016), pp. 305–338. ISSN: 1432-5012.

[27] J. Wu, ''Understanding of internal clustering validation measures,'' in Proc. IEEE Int. W. Song, L. C. Choi, S. C. Park, and X. F. Ding, ''Fuzzy evolutionary optimization modeling and its applications to unsupervised categorization and extractive summarization,'' Expert Syst. Appl., vol. 38, no. 8, pp. 9112–9121, Aug. 2011.

[28] R. Xu and D. Wunsch, Clustering, vol. 10. Hoboken, NJ, USA: Wiley, 2008. [29] X. Gu, P. P. Angelov, D. Kangin, and J. C. Principe, ‘‘A new type of distance metric and its use for clustering,’’ Evolving Syst., vol. 8, no. 3, pp. 167–177, Sep. 2017.

[30] G. Giannakopoulos, M. El-Haj, B. Favre, M. Litvak, J. Steinberger, and V. Varma, ‘‘TAC 2011 multiling pilot overview,’’ in Proc. Text Anal. Conf., 2011, pp. 1–22.

[31] P. J. Rousseeuw, ‘‘Silhouettes: A graphical aid to the interpretation and validation of cluster analysis,’’ J. Comput. Appl. Math., vol. 20, pp. 53–65, Nov. 1987

[32] R. Parker, D. Graff, J. Kong, K. Chen, and K. Maeda, ‘‘English gigaword fifth edition, linguistic data consortium,’’ Linguistic Data Consortiume, Philadelphia, PA, USA, Tech. Rep. LDC2011T07, 2011.

[32] E. Rendón, I. Abundez, A. Arizmendi, and E. M. Quiroz, ‘‘Internal versus external cluster validation indexes,’’ Int. J. Comput. Commun., vol. 5, no. 1, pp. 27–34, 2011. [33] Y. Liu, Z. Li, H. Xiong, X. GaoConf. Data Mining, Dec. 2010, pp. 911–916.

[34] A. Templeton and J. Kalita, ‘‘Exploring sentence vector spaces through automatic summarization,’’ in Proc. 17th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA), Dec. 2018, pp. 55–60

[35] A. Huang, ‘‘Similarity measures for text document clustering,’’ in Proc. 6th New Zealand Comput. Sci. Res. Student Conf. (NZCSRSC), Christchurch, New Zealand, 2008, pp. 49–56

[36] Q. Le and T. Mikolov, ‘‘Distributed representations of sentences and documents,’’ in Proc. Int. Conf. Mach. Learn., 2014, pp. 1188–1196.

[37] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, ‘‘Distributed representations of words and phrases and their compositionality,’’ in Proc. Adv. Neural Inf. Process. Syst., 2013, pp. 3111–3119.

[38] D. M. Blei, A. Y. Ng, and M. I. Jordan, ‘‘Latent Dirichlet allocation,’’ J. Mach. Learn. Res., vol. 3, pp. 993–1022, Jan. 2003.

[39] R. Montiel Soto, Y. Ledeneva, R. A. García-Hernández, and R. C. Reyes, ‘‘Comparación de tres modelos de texto para la generación automática de resúmenes,’’ Procesamiento Lenguaje Natural, vol. 43, no. 43, pp. 303–311, Sep. 2009.

[40] R. Montiel Soto, Y. Ledeneva, R. A. García-Hernández, and R. C. Reyes, ‘‘Comparison de tries modelos de texto para la generación automática de resúmenes,’’ Profenamines Language Natural, vol. 43, no. 43, pp. 303–311, Sep. 2009.

[41] R. A. García-Hernández, R. Montiel, Y. Lede Neva, E. Rendón, A. Gulrukh, and R. Cruz, ‘‘Text summarization by sentence extraction using unsupervised learning,’’ in Proc. Mex. Int. Conf. Arif. Intel. Berlin, Germany: Springer, 2008, pp. 133–143