WikiBot: A Chatbot for Summarizing and Recommending Wikipedia Articles

Group 30:
Ayush Patel, Shivansh Verma, Spandan Maaheswari
Khoury College of Computer Science, Northeastern University, Boston, MA, 02115

Abstract

WikiBot is a chatbot designed to provide quick access to relevant information from Wikipedia. Using web scraping techniques, the chatbot retrieves the most relevant articles and applies natural language processing techniques to generate a concise summary for the user. In addition, WikiBot recommends related articles based on the user's input using a recommendation algorithm that employs cosine distance and content-based collaborative filtering. The algorithm uses Laplace smoothing to avoid zero probabilities. The chatbot's ability to retrieve and summarize vast amounts of information in real-time makes it an invaluable tool for researchers, students, or anyone in need of quick access to information. The recommendations are generated using unsupervised machine learning algorithms such as K-Nearest Neighbours to generate clusters for recommendation. Overall, WikiBot is a powerful tool for anyone looking to access and summarize information from Wikipedia quickly and efficiently.

Introduction

In today's world, the ability to access and summarize information quickly is crucial for personal and professional growth. With the abundance of information available at our fingertips, the challenge is not so much finding information, but rather finding the relevant information and digesting it in a timely manner. For researchers, students, and professionals alike, the ability to access and summarize information quickly and efficiently is crucial to success. This is where WikiBot comes in, a chatbot designed to provide quick access to relevant information from Wikipedia.

WikiBot uses web scraping techniques to retrieve the most relevant articles and applies natural language processing techniques to generate a concise summary for the user. It can even recommend related articles based on the user's input using a recommendation algorithm that employs cosine distance and content-based collaborative filtering. With the ability to retrieve and summarize vast amounts of information in real-time, WikiBot is an invaluable tool for those who need to access information quickly and efficiently. It offers a solution to the challenge of information overload, allowing users to focus on the most relevant information and make informed decisions.

The report contains the technical aspects of WikiBot, including the algorithms used and their rationale, as well as the potential applications of this innovative tool. Additionally, the report will discuss the benefits and limitations of WikiBot, as well as potential future developments in the field of chatbots and information retrieval. By providing a detailed analysis of WikiBot's functionality and potential, this report aims to highlight the importance of efficient information management and demonstrate the potential of chatbots as a tool for achieving this goal.

Brief Literature Survey

Title of the Paper	**Key Findings**	**Gaps Identified**
Dialogue Generation Using Wikipedia as Knowledge Base" by Peng et al. (2018)	The proposed approach in the paper showed superior performance to existing state-of-the art dialogue generation models across various evaluation metrics. It was able to generate more informative and diverse responses by incorporating external knowledge from Wikipedia and using a latent variable model. The model was versatile enough to handle various types of dialogue tasks, including chit-chat, question answering, and knowledge grounded dialogue.	The model's performance is affected by the quality and bias of the available Wikipedia articles and may generate factually correct but socially inappropriate responses. The model's training and reliance on external knowledge sources require a large amount of data and computational resources, which may limit its applicability and raise ethical and privacy concerns.
Exploring Content Models for Multi-Document Summarization	LexRank outperformed other popular summarization methods, such as TextRank and Latent Semantic Analysis (LSA), on several benchmark datasets. LexRank is effective in capturing the most important and representative sentences in a document and can generate summaries that are both concise and informative.	LexRank is computationally intensive and may not scale well to large datasets. LexRank requires a pre-defined threshold for edge weights, which can be a challenge to set optimally for different types of documents and summarization tasks.
BART: Denoising Sequence-to-	BART outperforms previous state-of-the-art models on a range of	BART is a computationally expensive model that requires a

Sequence Pre-training for Natural Language Generation, Translation, and Comprehension (2019)	natural language generation, translation, and comprehension tasks. BART's denoising autoencoder architecture allows for a range of input and output types, making it a versatile pre-training method for a variety of natural language processing tasks.	large amount of data and computing resources to train effectively. BART's use of a denoising autoencoder may limit its ability to generate diverse outputs, as it encourages the model to reconstruct input sequences rather than generating novel variations.

Dataset and Preprocessing

The Wikipedia Vital Level 4 dataset is a curated collection of over 10,000 articles that cover essential topics in human knowledge. These articles meet certain criteria, such as being notable, widely recognized, and having a significant impact on human knowledge. By leveraging BeautifulSoup4 we crawled the Vital Level 4 articles homepage from https://en.wikipedia.org/wiki/Wikipedia:Vital_articles/Level/4. Then we found all the title headers for all the articles in each of the 11 subsections by identifying correct articles titles using complex BS4 logic and simultaneously removing all the pages that contained noise and improper articles. Finally, after getting all the article title headers from the webpage, we would then feed the article titles to the Python Wikipedia-API to fetch the articles and the summary. Since the API endpoints are rate-limited we have implemented appropriate timeouts in the API calls to avoid requests getting blocked by the endpoint. We are using this API as it is already load balanced for research purposes.

After fetching the content from Wikipedia-API we are then storing the articles and the corresponding summaries in separate folders for each of the categories in the textual format with all the formatting present in the actual articles. This helps us to bifurcate articles and summaries based on the categories. We are then reading all articles and picking up their summaries and labeling their categories based on their folder name (category name) and loading it into a Pandas Data Frame for further text cleaning. By leveraging the pandas framework, we were able to achieve faster analysis and cleaning of text as Pandas perform several under the hood optimization for column level transformations.

The cleaning process involves removing regular expressions, contractions, punctuation, and non-English words, converting all words to lowercase, replacing slang words with their formal equivalent, and performing stemming and lemmatization if desired. Stop words are also removed from the text. The final output is a new column containing the cleaned text data, which is ready for further analysis. Any missing values or empty rows are also removed in the process.

Exploratory Data Analysis (EDA)

These are some pre-analyses before generating summaries and recommending the articles to the users to help us in the process of feature selection, parameter tuning and spotting anomalies.

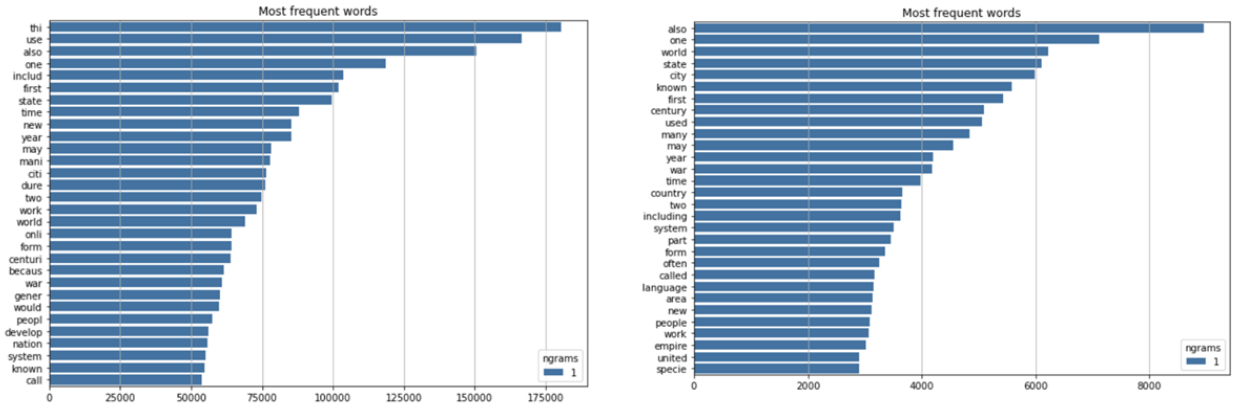


Figure 1: Top words in articles and summaries for the unigram (ngram = 1)

	min	mean	max
char_count	421.000000	19482.671296	126076.00000
word_count	60.000000	3371.737236	21625.00000
sentence_count	1.000000	1.000000	1.00000
avg_word_length	4.719607	5.786694	8.22809
avg_sentence_lenght	60.000000	3371.737236	21625.00000

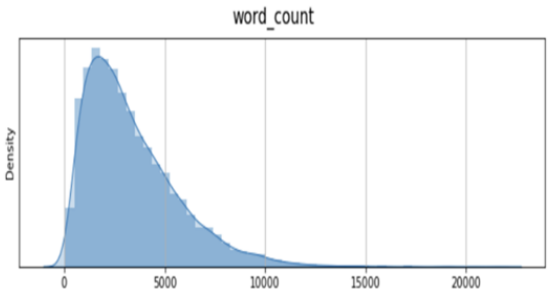


Figure 2: Statistics for word, character and sentence count

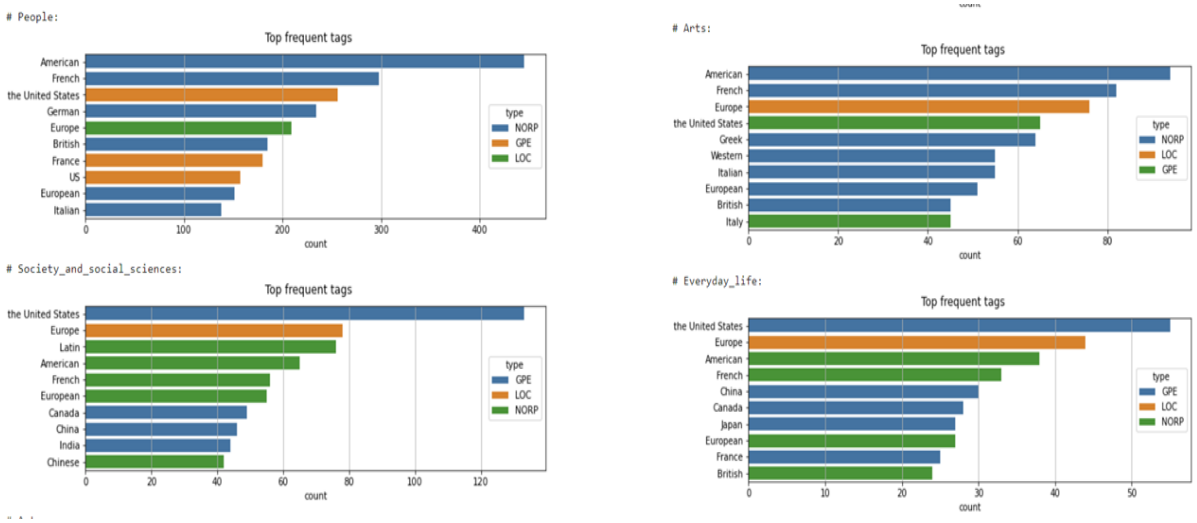


Figure 3: List of topmost tags of entities in different category of articles

Most frequent tags that occur in the category of articles after NER tagging, thus gives us the indication of what type of entities are most talked about in the articles pertaining to that category and for us to better understand the nature of the texts and entities mentioned in the documents.

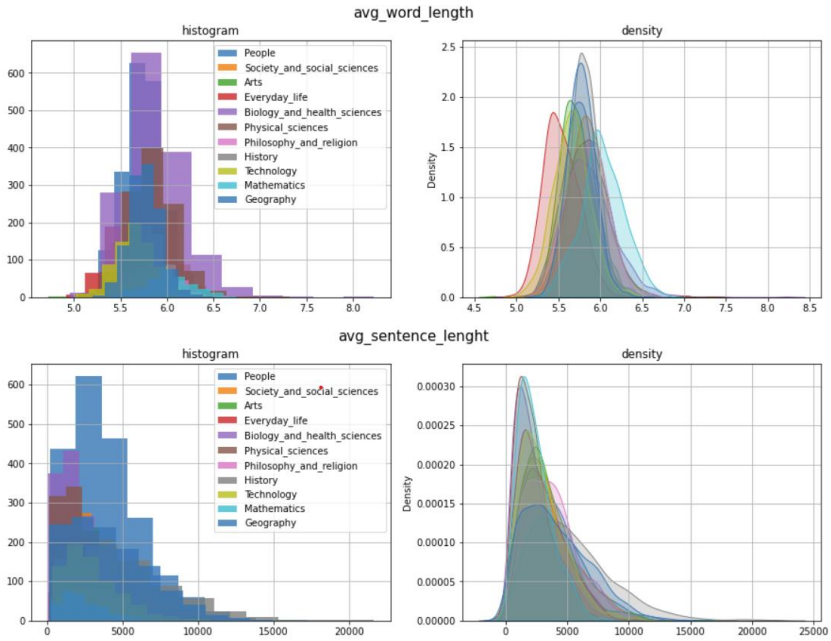


Figure 4: Density distribution for words and characters for all categories

This density distribution helps us to understand the distribution of words and characters in each category. This helps us understand the density of words per category as some category will contain more textual content than others and some would contain less textual content.

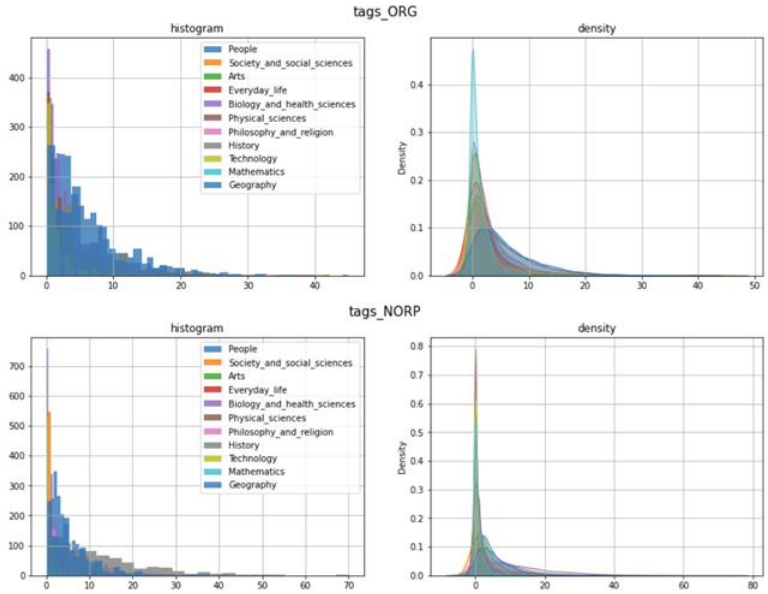
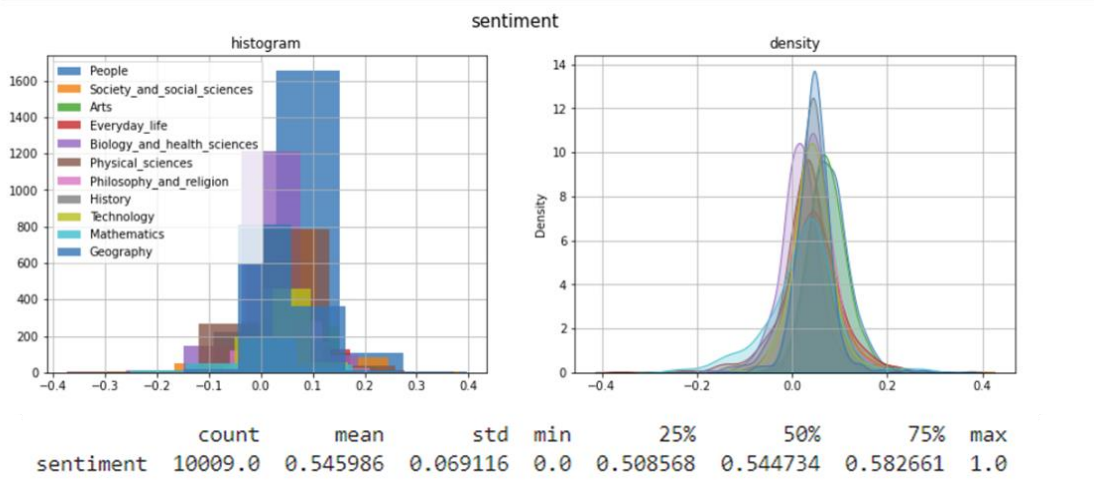


Figure 5: Density distribution of Organization and Nationalities or religious or political groups for all categories



	count	mean	std	min	25%	50%	75%	max
sentiment	10009.0	0.545986	0.069116	0.0	0.508568	0.544734	0.582661	1.0

Figure 6: Density Distribution of sentiment of articles. Sentiment ranges from -1 to 1

Sentiment analysis is the process of using natural language processing, text analysis and computational linguistics to identify and extract subjective information from a text corpus. It involves analyzing the emotional tone and attitude expressed in a piece of text, and determining whether the sentiment is positive, negative or neutral. Sentiment analysis can be performed using various techniques such as lexicon-based methods, machine learning-based methods, and hybrid methods. These methods involve using algorithms to analyze the text data and assign a sentiment score to each piece of text. Here we have used VADER, which is a lexicon-based method that uses a pre-built sentiment lexicon to determine the sentiment of a text, while TextBlob is a machine learning-based method that uses a trained model to classify the sentiment of a text. As you can see from the density distribution, the sentiment of the articles are mainly ‘positive’.

```
features selection: from 100,000 to 9,739

# Arts:
  . selected features: 739
  . top features: art, film, music, paint, artist, instrument, genr, danc, novel, style

# Biology_and_health_sciences:
  . selected features: 1515
  . top features: cell, genu, plant, speci, genus, protein, tissu, flower, infect, diseas

# Everyday_life:
  . selected features: 629
  . top features: ball, game, player, sport, olymp, ski, sauc, bowl, tournament, sexual

# Geography:
  . selected features: 1098
  . top features: citi, island, lake, river, sea, airport, park, district, mountain, strait

# History:
  . selected features: 862
  . top features: empir, war, armi, dynasti, ottoman, kingdom, defeat, byzantin, troop, alli

# Mathematics:
  . selected features: 1331
  . top features: algebra, algorithm, axiom, calculu, complex number, displaystyl, displaystyl displaystyl, displaystyl mathbb, equat, euclidean

# People:
  . selected features: 394
  . top features: career, father, work, award, marri, son, film, wrote, daughter, friend

# Philosophy_and_religion:
  . selected features: 884
  . top features: belief, church, deiti, god, myth, religion, philosophi, worship, goddess, judaism

# Physical_sciences:
  . selected features: 1435
  . top features: atom, energi, galaxi, hydrogen, magnet, orbit, oxid, particl, star, earth

# Society_and_social_sciences:
  . selected features: 582
  . top features: conson, languag, vowel, verb, social, dialect, noun, law, phonem, linguist
```

Figure 7: Top features for each category and their count

We performed feature selection using chi squared test and set the p value as 0.95 to find out the most import features in each of the categories. This helped us understand the number features we require to build our recommendation system and to understand the prevalent features which can represent each of the categories.

Name Entity Recognition

Named Entity Recognition involves identifying and categorizing named entities in text into predefined categories such as person names, organizations, locations, medical codes, etc. NER has various applications in different fields such as information extraction, sentiment analysis, question answering, and summarization. For example, in our project, we used NER to highlight important entities in Wikipedia articles which could be helpful in generating summaries.

Figure 8: NER tagging for a sample summary.

NER Architecture

The architecture of NER involves several components, including tokenization, part-of-speech tagging, and sequence labeling. The first step in NER is tokenization, where the text is split into individual tokens or words. The next step is part-of-speech tagging, where each token is labeled with its corresponding part-of-speech (e.g., noun, verb, adjective).

The final step in NER is sequence labeling, where the model assigns a label to each token in the text sequence indicating whether it belongs to a named entity or not. Popular sequence labeling algorithms for NER include Hidden Markov Models, Conditional Random Fields, and Bidirectional LSTMs with CRF.

During training, the NER model is trained on a labeled dataset where each token is annotated with its corresponding named entity type. The model then learns to predict the named entity label for

each token in a new sequence of text. Overall, the NER architecture allows for the efficient identification and classification of named entities in text, which can be used in various downstream applications such as sentiment analysis and text summarization.

Text Classification

Accuracy: 0.62
Auc: 0.96
Detail:

	precision	recall	f1-score	support
Arts	1.00	0.05	0.10	134
Biology_and_health_sciences	0.77	0.92	0.84	296
Everyday_life	1.00	0.04	0.08	95
Geography	0.67	0.96	0.79	241
History	0.70	0.19	0.30	137
Mathematics	1.00	0.25	0.40	60
People	0.44	0.93	0.60	399
Philosophy_and_religion	0.86	0.07	0.13	87
Physical_sciences	0.75	0.84	0.79	220
Society_and_social_sciences	0.82	0.32	0.46	185
Technology	0.82	0.39	0.53	148
accuracy			0.62	2002
macro avg	0.80	0.45	0.46	2002
weighted avg	0.73	0.62	0.56	2002

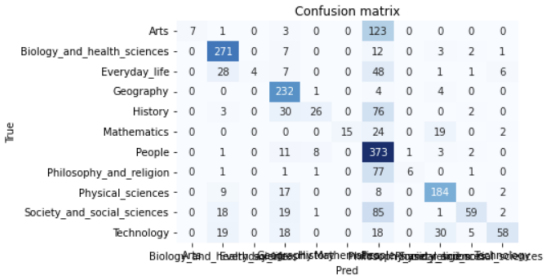


Figure 9: Classification report and Classification Report for the classification

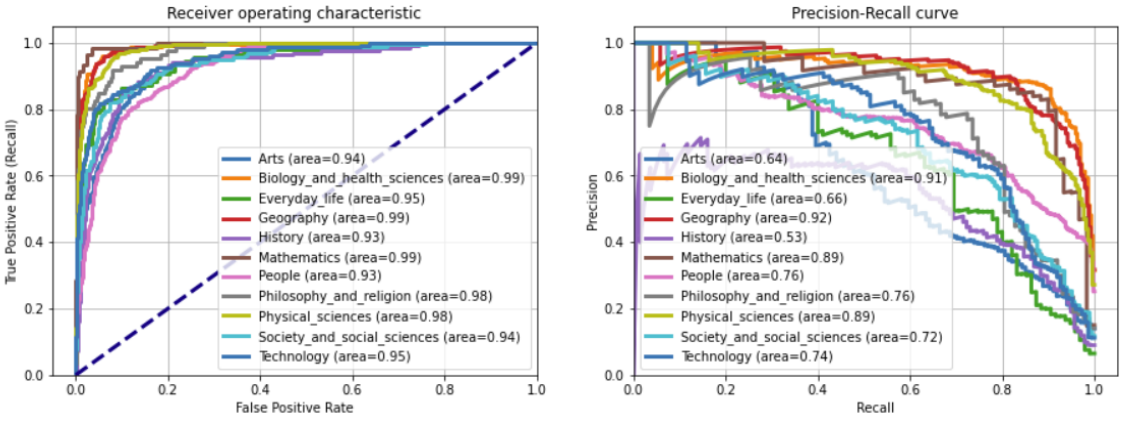


Figure 10: Receiver Operating Characteristic (ROC) and Precision-Recall Curve

Figure 11: Predicting the class of an article and highlighting the words that give indication of the class belonging to that category.

Multinomial Naive Bayes is a probabilistic machine learning algorithm that is commonly used for text classification tasks such as sentiment analysis, spam filtering, and topic modeling. It is based on Bayes' theorem, which states that the probability of a hypothesis (in this case, a document belonging to a particular class) can be calculated by combining prior knowledge (the probability of the class occurring in the training data) with new evidence (the features or words in the document).

In Multinomial Naive Bayes, the model assumes that the features or words in the document are independent of each other, and that the frequency of each word follows a multinomial distribution. The algorithm first learns the probabilities of each feature or word occurring in each class based on the training data. It then uses these probabilities to calculate the likelihood of a new document belonging to each class and combines this with the prior probability of each class to make a prediction. The class with the highest probability is assigned as the predicted class for the document.

Multinomial Naive Bayes is a fast and efficient algorithm that can handle large amounts of text data with high dimensionality. However, its assumption of independence between features may not hold true for all text classification tasks, and it may not perform well on imbalanced datasets or when there are overlapping features between classes. You can see from the above diagram that the predicted and the actual category are same, and the probability is 0.86.

Multi-Class Classification

The model describes the process of multi-class text classification using Doc2Vec and logistic regression algorithms. The Doc2Vec model was used to convert the text data into fixed-length feature vectors, which were then used as input to a logistic regression classifier. Doc2Vec is an unsupervised algorithm that can generate fixed-length feature vectors for a given text document by considering the context of words within the document, allowing it to capture the semantic meaning of the text. The results showed that the proposed approach achieved an accuracy of over 75%, demonstrating the effectiveness of Doc2Vec and logistic regression for multi-class text classification tasks.

Doc2Vec Architecture

Doc2Vec is a neural network-based algorithm that learns a dense, fixed-length vector representation of a document from its words. The architecture of Doc2Vec is an extension of Word2Vec, a popular algorithm for learning word embeddings.

The Doc2Vec architecture involves two main components: a paragraph vector and a word vector. The paragraph vector, also known as the document vector or the document embedding, is a fixed-length vector that represents the entire document. The word vector, also known as the word embedding, is a fixed-length vector that represents each word in the document.

The architecture involves two main approaches for learning paragraph vectors: Distributed Memory Model of Paragraph Vectors (PV-DM) and Distributed Bag of Words Model of Paragraph Vectors (PV-DBOW). In the PV-DM approach, the algorithm predicts the next word in a sentence given a context of words and the paragraph vector, while in the PV-DBOW approach, the paragraph vector is predicted given a context of words.

During training, the Doc2Vec algorithm learns to optimize the objective function that maximizes the probability of the context given the target word or the probability of the target paragraph given the context words. The learned vectors can then be used to represent documents in downstream tasks such as classification, clustering, and information retrieval.

Overall, the Doc2Vec architecture allows for the efficient learning of document embeddings, which can be used to represent and compare documents in a high-dimensional vector space. This can be a powerful tool for text analysis tasks, especially when combined with other NLP techniques such as topic modeling and named entity recognition.

Summarization

In natural language processing, there are two types of summarizations:

- **Abstractive text summarization**: The abstractive approach involves summarization based on deep learning. So, it uses new phrases and terms, different from the actual document, while keeping the points the same, just like how we summarize. This results in a natural, grammatically accurate summary. In this approach we used Large-BART model.

- **Extractive summarization**: The extractive approach involves picking up the most important phrases and lines from the documents. It then combines all the important lines to create the summary. Every line and word of the summary belongs to the original document which is summarized. Extractive summaries can often be inconsistent grammatically, as sentences are stitched together haphazardly. In this approach we used 3 models which are TextRank, Latent Dirichlet Allocation (LDA) and Non-negative Matrix-Factorization (NMF).

BART

BART stands for bidirectional autoregressive transformer, a reference to its neural network architecture. BART proposes an architecture and pre-training strategy that makes it useful as a sequence-to-sequence model (seq-2-seq model) for any NLP task, like summarization, machine translation, categorizing input text sentences, or question answering under real-world conditions. In this article, we'll focus on its summarization capabilities.

How does BART summarize text?

BART (Bidirectional and Auto-Regressive Transformer) is a pre-trained sequence-to-sequence (seq2seq) model that is designed to perform a variety of natural language processing tasks, including text summarization.

To summarize text, BART uses an encoder-decoder architecture that combines bidirectional and unidirectional transformer layers. The encoder processes the input text and generates a sequence of hidden states that represent the input sequence's contextual information. The decoder then takes these hidden states as input and generates a summary by predicting the next token in the summary sequence at each step.

During training, BART is trained to minimize the difference between the predicted summary and the ground truth summary using a reconstruction loss, which measures the difference between the predicted and actual summaries in terms of their token distributions.
When summarizing text, BART takes the input text as a sequence of tokens, encodes it using the transformer layers in the encoder, and then generates a summary using the transformer layers in the decoder. The decoder is trained to predict the next token in the summary sequence given the previous tokens generated by the encoder. This process continues until a predefined stop token is generated, or a maximum summary length is reached.

Overall, BART's ability to generate high-quality summaries stems from its ability to encode and decode text sequences using a combination of bidirectional and unidirectional transformer layers, as well as its training procedure, which uses a reconstruction loss to encourage the model to generate summaries that are similar to human-generated summaries.

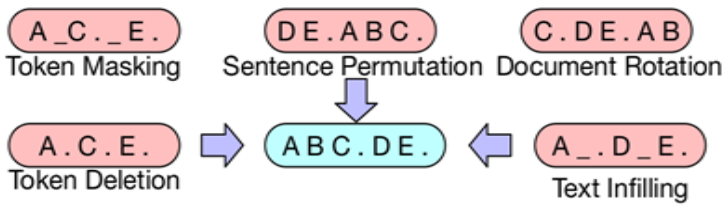


Figure 12: Transformations for noising the input.

BART-Architecture

The Large-BART architecture is significantly larger than its predecessor, with 406 million parameters, making it one of the largest pre-trained language models available. It was trained on a massive dataset of 160GB of text, including books, articles, and web pages.

Large-BART has several improvements over BART, including improved training procedures and architecture modifications. One notable difference is that it uses a "encoder-decoder-decoder" architecture instead of the "encoder-decoder" architecture used in BART. This additional decoder layer helps to improve the model's performance on tasks that require generating longer sequences of text.

Large-BART has achieved state-of-the-art results on a range of natural language processing (NLP) tasks, including machine translation, summarization, and question answering. Its large size and superior performance make it a valuable tool for researchers and practitioners in the field of NLP.

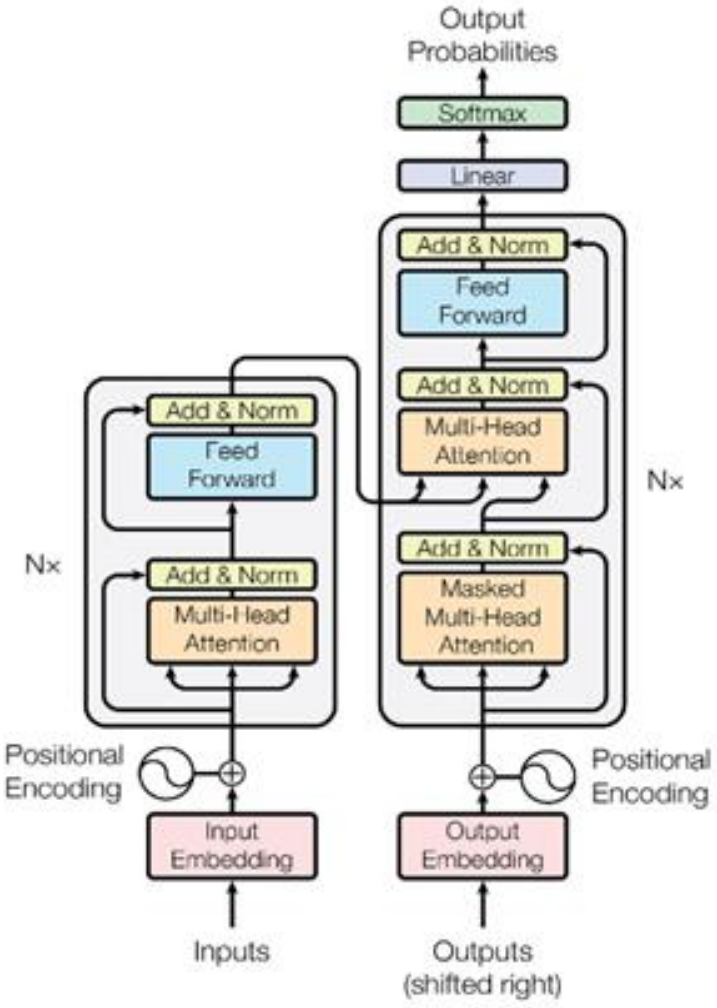


Figure 13: BART Architecture

TextRank

Like all Extractive Summarization methods TextRank takes parts of the text (sentences, phrases, paragraphs, etc.). Therefore, identifying right sentences for summarization is a vital process for summarization. TextRank algorithm is based on PageRank algorithm which was originally developed by Google to rank pages for their search engines. Inplace of webpages, TextRank uses the sentence and its corresponding similarities as a transition is equated to page transition probability.

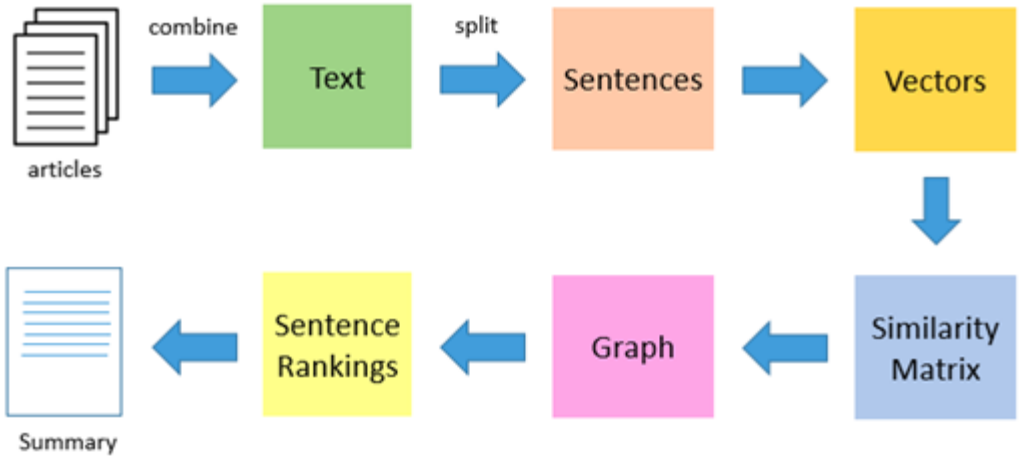


Figure 14: TextRank Implementation

How does TextRank summarize text?

TextRank is a graph-based algorithm that can be used to automatically summarize text. It is based on the PageRank algorithm, which was originally developed by Google to rank web pages based on their importance.

To summarize text using TextRank, the first step is to split the text into individual sentences. Then, a graph is created, where each sentence is represented as a node, and the edges between the nodes represent the similarity between the sentences. The similarity between sentences can be calculated using a variety of measures, such as cosine similarity or Jaccard similarity, which take into account the overlap of words or phrases in the sentences.

Once the graph is constructed, the TextRank algorithm applies an iterative process that calculates a score for each sentence based on the scores of its neighboring sentences in the graph. The scores are initially set to a default value, and then updated iteratively until the scores converge. The scores represent the importance of each sentence in the context of the entire text. Finally, the top-scoring sentences are selected to form the summary. The length of the summary can be adjusted by specifying the desired number of sentences.

TextRank is a simple and effective approach to automatic summarization that does not require pre-training on a large dataset. It can be applied to a variety of text types, including news articles, scientific papers, and legal documents, among others. However, it has some limitations, such as the inability to generate coherent summaries that capture the overall meaning of the text.

Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation (LDA) is a generative statistical model that explains a set of observations through unobserved groups, and each group explains why some parts of the data are similar. The LDA is an example of a topic model. In this, observations (e.g., words) are collected into documents, and each word's presence is attributable to one of the document's topics. Each document will contain a small number of topics. This is a method of topic Modelling. It assumes that documents are generated by a process in which a set of topics are chosen, and then words are generated based on those topics. This means that the topics are latent variables that are inferred from the observed words in the documents.

Non-negative Matrix-Factorization (NMF)

NMF is a form of Topic Modelling — the art of extracting meaningful themes that recur through a corpus of documents. A corpus is composed of a set of topics embedded in its documents. A document is composed of a hierarchy of topics. A topic is composed of a hierarchy of terms. NMF is a matrix factorization technique that decomposes a matrix into two non-negative matrices that represent a set of basis vectors and a set of coefficients. In the context of topic modeling, the vectors represent the topics and the coefficients represent the distribution of topics in each document. It is often used as an alternative to LDA, particularly in cases where the number of topics is small and well-defined.

Article Recommendation

Article recommendation is a type of recommendation system that suggests relevant articles to users based on their interests and preferences. The recommender takes an input article and calculates the cosine similarity between that article and other articles in the same cluster. It then selects the K-nearest neighbors based on the similarity distance and recommends articles that match both the category and the keywords of the input article. While cosine similarity is a widely used metric for article recommendation, it has some limitations. For example, it may not work well for articles with very short or very long content. It also does not consider the temporal aspects of articles, such as the publication date or the popularity of the article. Therefore, it is important to carefully consider the limitations of cosine similarity when designing an article recommendation system. The summarizer creates features from the summary of the document in consideration and then finds similar documents based on their cosine similarity.

We also employed another method of article recommendation that was based on the TF-IDF vectorization of the summaries of articles. We firstly vectorize all summaries and limit the number of features to 1000 to have easier computation as the vocabulary size is large resulting into words that are very infrequent. Thus, after vectorization we take the input of the article from user, we vectorize the summary of the article in consideration. Then using cosine similarity, we are then finding the top K articles with the highest cosine similarity with the current article thus giving us the most similar articles.

```
--------------------------------------------------------------------------------------------
Decoded TF-IDF form of the input article summary is:

['100', '2019', '2020', '2021', '2022', 'act', 'action', 'american', 'attempt', 'back', 'became', 'border', 'born', 'building', 'business', 'campaign', 'change', 'charac
terized', 'charge', 'china', 'climate', 'company', 'congress', 'country', 'court', 'death', 'december', 'degree', 'democratic', 'described', 'despite', 'election', 'envi
ronmental', 'established', 'expanded', 'family', 'father', 'february', 'federal', 'first', 'government', 'health', 'historian', 'history', 'house', 'including', 'individ
ual', 'involved', 'iran', 'january', 'john', 'june', 'later', 'leader', 'legal', 'lost', 'made', 'making', 'many', 'march', 'medium', 'military', 'mostly', 'multiple',
'name', 'national', 'need', 'north', 'november', 'nuclear', 'numerous', 'office', 'official', 'one', 'operation', 'organization', 'paris', 'party', 'policy', 'politica
l', 'politics', 'popular', 'position', 'power', 'president', 'pressure', 'rank', 'real', 'remained', 'result', 'resulting', 'russia', 'scholar', 'school', 'second', 'ser
ies', 'served', 'service', 'several', 'side', 'since', 'six', 'special', 'state', 'television', 'theory', 'three', 'time', 'towards', 'trade', 'treatment', 'united', 'un
iversity', 'wall', 'war', 'widespread']
============================================================================================
Titles of recommended articles:

['Donald Trump', 'Woodrow Wilson', 'Henry Clay', 'Lyndon B. Johnson', 'Aung San Suu Kyi', 'Barack Obama', 'Election', 'Theodore Roosevelt', 'John C. Calhoun', 'Franklin
D. Roosevelt']
============================================================================================
```

Figure 15: Article recommendations using Tf-idf vectorizer.

```
Title:  Donald Trump

Recommended articles:

Che Guevara
Harriet Tubman
```

Figure 16: Article Recommendation using KNN.

Results

Model	Rouge-1	Rouge-2	Rouge-l
BART	**0.52**	**0.29**	**0.50**
TextRank	**0.32**	**0.13**	**0.25**
LDA	**0.88**	**0.88**	**0.88**
NMF	**0.48**	**0.38**	**0.45**

Real Summary

Manta rays are large rays belonging to the genus Mobula (formerly its own genus Manta). The larger species, M. birostris, reaches 7 m (23 ft) in width, while the smaller, M. alfredi, reaches 5.5 m (18 ft). Both have triangular pectoral fins, horn-shaped cephalic fins and large, forward-facing mouths. They are classified among the Myliobatiformes (stingrays and relatives) and are placed in the family Myliobatidae (eagle rays). They have the largest brains and brain to body ratio of all fish, and can pass the mirror test. Mantas are found in warm temperate, subtropical and tropical waters. Both species are pelagic; M. birostris migrates across open oceans, singly or in groups, while M. alfredi tends to be resident and coastal. They are filter feeders and eat large quantities of zooplankton, which they gather with their open mouths as they swim. However, research suggests that the majority of their diet (73%) actually comes from mesopelagic sources. Gestation lasts over a year and mantas give birth to live pups. Mantas may visit cleaning stations for the removal of parasites. Like whales, they breach for unknown reasons. Both species are listed as vulnerable by the International Union for Conservation of Nature. Anthropogenic threats include pollution, entanglement in fishing nets, and direct harvesting of their gill rakers for use in Chinese medicine. Their slow reproductive rate exacerbates these threats. They are protected in international waters by the Convention on Migratory Species of Wild Animals, but are more vulnerable closer to shore. Areas where mantas congregate are popular with tourists. Only a few public aquariums are large enough to house them.

Predicted Summary

Anthropogenic threats include pollution, entanglement in fishing nets, and direct harvesting of their gill rakers for use in Chinese medicine. They are protected in international waters by the Convention on Migratory Species of Wild Animals, but are more vulnerable closer to shore. In 2018, an analysis of DNA, and to a lesser degree, morphology, found that Mobula was paraphyletic with respect to the manta rays; that is, the members of the genus Manta are closer related to some members of genus Mobula than them to other Mobula, and they recommended treating Manta as a junior synonym of Mobula.Mantas evolved from bottom-dwelling stingrays, eventually developing more wing-like pectoral fins. The edges of the jaws line up while in devil rays, the lower jaw shifts back when the mouth closes.: 14 Manta rays and devil rays are the only ray species that have evolved into filter feeders. All were eventually treated as synonyms of the single species Manta birostris. The specific name alfredi was first used by Australian zoologist Gerard Krefft, who named the manta after Prince Alfred.A 2009 study analyzed the differences in morphology, including color, meristic variation, spine, dermal denticles (tooth-like scales), and teeth of different populations. Manta rays have broad heads, triangular pectoral fins, and horn-shaped cephalic fins located on both sides of their mouths. The fish, spotted near Lady Elliot Island, is the world's only known pink manta ray. birostris has a caudal spine near its dorsal fin.Mantas move through the water by the wing-like movements of their pectoral fins. The spiracles typical of rays are vestigial and concealed by small flaps of skin, and mantas must keep swimming with their mouths open to keep oxygenated water passing over their gills.: 13 The cephalic fins are usually spiraled but flatten during foraging. The ray adopts a near-stationary position close to the coral surface for several minutes while the cleaner fish feed. In addition, it has been confirmed that reef manta rays form a community with a specific individual and act together.Mantas may be preyed upon by large sharks, orcas and false killer whales. Though they may clean them of parasites,

Figure 17: TextRank Summary Comparison

From our output we can see that TextRank produces a summary based on the sentences based on the original article. Since it is extractive in nature it often fails to produce a natural fluent sounding summary such that of BART which uses an abstractive method.

```
--- Generated Summary ---
masaccio italian decemb 1401 summer 1428 born tommaso ser giovanni simon florentin artist regard first great italian painter quattr
ocento period italian renaiss accord vasari masaccio best painter gener becaus skill imit natur recreat lifelik figur movement well
convinc sen threedimension employ nude foreshorten figur thi seldom done befor name masaccI0 humor version maso short tommasco mean
clumsi messi tom name may creat distinguish princip collabor also call maso came known masolino littledel tom despit brief career p
rofound influenc artist consid start earli italian prenaiss paint work mid late1420 one first use linear perspect paint employ tech
niqu vanish point art first time move away intern gothic style elabor ornament artist like gentil fabriano naturalist mode employ p
erspect chiaroscuro greater realism masaccios art in
```

Figure 18: Bart Summary

The summary generated by BART has a Rogue-1 score of 0.52 which is much higher than TextRank due to the fact being that it is highly fine-tuned to handle large volumes of text. BART can capture both the contextual meaning of words and the global structure of the document, which allows it to generate more accurate and coherent summaries compared to TextRank, which is a graph-based algorithm that ranks sentences based on their similarity to the overall text.

['Philip Cortelyou Johnson (July 8, 1906 – January 25, 2005) was an American architect best known for his works of modern and postm
odern architecture. Among his best-known designs are his modernist Glass House in New Canaan, Connecticut; the postmodern 550 Madis
on Avenue in New York, designed for AT&T; 190 South La Salle Street in Chicago; the Sculpture Garden of the Museum of Modern Art; a
nd the Pre-Columbian Pavilion at Dumbarton Oaks. In his obituary in 2005, The New York Times wrote that his works "were widely cons
idered among the architectural masterpieces of the 20th century."In 1930, Johnson became the first director of the architecture dep
artment of the Museum of Modern Art in New York. There he arranged for visits by Walter Gropius and Le Corbusier and negotiated the
first American commission for Mies van der Rohe, when he fled Nazi Germany. In 1932, he organized the first exhibition on modern ar
chitecture at the Museum of Modern Art.\nIn 1934, Johnson resigned his position at the museum, and, as the New York Times reported
in his obituary, "took a bizarre and, he later conceded, deeply mistaken detour into right-wing politics, suspending his career to
work on behalf of Gov. Huey P. Long of Louisiana and later the radio priest Father Charles Coughlin, and expressing more than passi
ng admiration for Hitler." In 1941, as the war approached, Johnson abruptly quit Coughlin\'s newspaper and journalism. He was inve
stigated by the FBI, and was eventually cleared for military service. Years later he would refer to these activities as "the stupid
est thing I ever did [which] I never can atone for".\n1978, he was awarded an American Institute of Architects Gold Medal and in 19
79 the first Pritzker Architecture Prize. Today his skyscrapers are prominent features in the skylines of New York, Houston, Chicag
o, Detroit, Minneapolis, Pittsburgh, Atlanta, Madrid, and other cities.']

Figure 19: LDA Summary

LDA is a topic modelling technique that is often used for summarization. In our use case we got the highest rouge score of 0.88 by using the LDA summarizer because LDA is a probabilistic model that is based on statistical principles. It is a generative model that uses statistical inference to estimate the topic distribution in a document and the word distribution in a topic. This statistical foundation allows LDA to capture uncertainty and variability in the data, which can result in more robust and reliable summaries.

In 1947, he curated the first exhibition of modern architecture of the Museum of Modern Art including a model of the glass Farnswor
th House of Mies.In 1949 he began building a new residence, the Glass House in New Canaan, Connecticut, that was completed in 1949.
"In 1930, Johnson became the first director of the architecture department of the Museum of Modern Art in New York.
There he arranged for visits by Walter Gropius and Le Corbusier and negotiated the first American commission for Mies van der Rohe,
when he fled Nazi Germany.
The Man in the Glass House: Philip Johnson, Architect of the Modern Century.
Philip Cortelyou Johnson (July 8, 1906 – January 25, 2005) was an American architect best known for his works of modern and postmod
ern architecture.

Figure 20: NMF Summary

We can see that NMF outperforms TextRank but performs lower than LDA and BART; it is because NMF is a matrix factorization technique that can extract relevant features from the original text and represent them in a lower-dimensional space, allowing for content extraction and abstraction. It is often used as an alternative to LDA, particularly in cases where the number of topics is small and well-defined. In contrast, TextRank is a graph-based algorithm that ranks sentences based on their centrality in a network of interconnected sentences. While TextRank can identify important sentences based on their position in the graph, it may not necessarily capture the underlying content of the text as effectively as NMF, which extracts more meaningful features.

Conclusion

We introduced WikiBot, a useful conversational AI tool that leverages the wealth of knowledge available on Wikipedia to provide users with quick access to relevant information. With its ability to summarize and recommend related articles, WikiBot provides a seamless experience for users seeking information on a wide range of topics. The bot's use of web scraping and NLP techniques to extract key information from articles and generate summaries for users demonstrates the power of AI in simplifying complex tasks.

After our analysis we found that in text summarization TextRank performs lower than abstractive methods; but LDA summarization gave us a higher ROUGE score than all the methods since our text contained very structured sentences which plays into the strength of LDA's topic modelling capabilities. Therefore, we understood that for structured text LDA can be used as a method of summarization.

Moreover, the use of unsupervised machine learning algorithms like clustering and collaborative filtering in generating recommendations adds to the tool's functionality. The potential for WikiBot to improve the way we access and interact with information is vast, and further exploration and development of the tool can lead to exciting advancements in conversational AI.

Future Work

Multi-lingual support: Currently, WikiBot only supports English Wikipedia articles. Adding support for other languages would make the tool accessible to a wider audience and expand its usefulness. Alongside this we can do Neural Machine Translation for articles that don't have multilingual support in other languages.

Fine-tuning summarization algorithms: While WikiBot's current summarization algorithms work well for most articles, there is always room for improvement. Fine-tuning the algorithms with additional training data and exploring new techniques such as GPT-3 models could improve the quality of the generated summaries and provide much more engaging responses for users to interact.

Personalization: Adding personalization features such as user profiles and learning user preferences could improve the accuracy of the recommendation system and make the tool more engaging for users.

Integration with other knowledge sources: While Wikipedia is a vast and comprehensive knowledge source, integrating WikiBot with other sources such as academic journals, news outlets, and other online databases could provide users with even more information and resources.

Ultimately, WikiBot provides a glimpse into the potential for AI and LLMs to revolutionize the way we access and engage with knowledge, and its development is a promising step in this direction. We are also planning to integrate Open-AI's API into our model. By integrating this, users can ask questions based on the summaries generated by the model and can even customize the recommendation system.

References

1. Yamada, I., Tamaki, R., Shindo, H., & Takefuji, Y. (2016). Studio Ousia's Quiz Bowl Question Answering System. arXiv:1603.07042.
2. "Dialogue Generation Using Wikipedia as Knowledge Base" by Peng et al. (2018).
3. "Wikipedia2Vec: An Efficient Toolkit for Learning and Visualizing the Embeddings of Words and Entities from Wikipedia" by Yamada et al. (2018).
4. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension.
5. Exploring Content Models for Multi-Document Summarization.
6. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin (2017). "Attention Is All You Need". arXiv:1706.03762.
7. Angela Fan, David Grangier, and Michael Auli. Controllable abstractive summarization. arXiv preprint arXiv:1711.05217, 2017.
8. Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171– 4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/ v1/N19-1423.
9. Yang Liu and Mirella Lapata. Text summarization with pretrained encoders. arXiv preprint arXiv:1908.08345, 2019.
10. Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. arXiv preprint arXiv:1705.04304, 2017.