Enhancing News Article Analysis: Integrating Topic Modeling and Summarization Techniques

Vittorio Haardt, Luca Porcelli, Fabio Salerno

Abstract

This project investigates the application of text mining techniques, focusing on Topic Modeling and Summarization, to enhance text handling systems. Topic modeling study evaluates traditional Latent Dirichlet Allocation and deep network-based BERT models for Topic Modeling. Additionally, summarization study compares extractive and abstractive summarization strategies. The goal is to contribute to a cohesive text handling system, particularly for articles and news texts. The conclusion discusses results and proposes future implementations.
Keywords: CNN / Daily Mail News; Topic Modeling; Summarization

Contents

1. Introduction

In text mining, the intersection of topic modeling and text summarization presents a could contribute for the development of cohesive text handling systems. This project delves into these two techniques, each aimed to distinct objectives, with the ultimate aim of contributing to understanding text processing strategies.

The first part, after the data intrdouction, of this project explores Topic Modeling, an unsupervised machine learning technique for identifying word and phrase patterns within a set of documents. Unlike traditional Text Clustering, which focuses on document grouping based on similarity measures, Topic Modeling clusters words into topic-specific groups. This process assigns a probability of occurrence to each word within a cluster, enabling the characterization of documents with multiple associated topics.

Concurrently, the second section delves into text summarization, a process involving the distillation of essential information from sources to create concise versions of the original text. The primary goal is to generate abridged texts containing information of higher importance or relevance to the end user.

This project aim to apply and study various techniques within both branches of text mining to evaluate their efficacy. Considering the field of the used data, this analysis could be imagined for a more cohesive text handling system designed to manage articles and news texts. For instance, topic modeling may be employed for a content tagging system, while summarization techniques could offer users insightful previews of articles.

Structured into two distinct parts, the project first focuses on Topic Modeling, delineating a strategy for obtaining an optimal Latent Dirichlet Allocation model, a benchmark for traditional topic modeling. Subsequently, this model is contrasted with a BERT topic model that incorporate deep neural networks.

The second part of the project centers on Summarization techniques, initially employing an extractive strategy, where summaries are crafted from source document phrases or sentences. This method is then juxtaposed against a fine-tuned abstractive summarization strategy, wherein the model generates its own coherent phrases and sentences.

Finally, the conclusion section synthesizes the findings and outlines potential avenues for future research and implementation. This academic endeavor seeks to contribute to the ongoing discourse on text mining, offering insights into the intricate interplay between topic modeling and text summarization techniques.

2. Data

The data utilized for this project is sourced from the CNN-DailyMail News dataset [14]. The CNN/DailyMail Dataset is an English-language collection that comprises just over 300,000 unique news articles written by journalists from CNN and the Daily Mail.

This dataset is organized into Train, Validation, and Test subsets. Each data point includes both the article (a string containing the body of the news article) and the highlights (a string containing the article's key points), which serve as the ground truth for the summarization task. The text also includes an ID, although it was not utilized for any of the tasks.

Due to the substantial volume of the data, we opted for a random sampling approach, selecting 20,000 observations exclusively from the original training set. This consistent sample is maintained throughout the entirety of the project. As needed, the new set of data is further subdivided within the project.

3. Topic Modeling

A. Introduction

In this section, we present the methodology employed for conducting topic modeling on the previously discussed subset. The chosen method for this task is the Latent Dirichlet Allocation (LDA) topic modeling strategy [2]. Our objective was to achieve best possible topic identification, utilizing an approach that maximizes performance specifically within our set of texts.

The purpose of this section is to demonstrate the process of attaining the most effective LDA topic modeling, which is subsequently compared with a BERT-based approach on the same subset. This serves as a benchmark for performance evaluation.

The evaluation of the models will be executed using "perplexity" and "coherence" scores. Precisely, the perplexity metric will be used to optimize the hyperparameters of the LDA model - a strategy that is discussed in detail in the experiment section that follows. Coherence, specifically "u_mass" variance, will be later employed to tune the number of topics for the document. The resultant best-performing LDA model will then be compared with the BERT model, using both choerence and qualitative human evaluation.

B. Pre-procesing

To thoroughly asses this task, it is mandatory to implement a pre-processing strategy that facilitates the use of text suitable for the LDA model. The initial and apparent step involves the removal of punctuation, retaining only alphanumeric characters. Subsequently, the text is converted to lowercase, constituting another fundamental pre-processing step. Delving into more specific pre-processing, stop words were eliminated using a list of English stop words from the `nltk` Python library [7]. Additionally, the '`simple_preprocessing`' function from the Gensim library [12] was employed to tokenize the text and eliminate accented characters.

Following these foundational pre-processing steps, n-grams were applied to the text. Specifically, three strategies were employed: retaining unigrams, incorporating bigrams, and incorporating trigrams. The detailed discussion of the rationale behind these three strategies is deferred to the subsequent section. The main objective was to treat this as a hyperparameter and identify the n-gram strategy that optimizes LDA performance. For bigrams and trigrams, a co-occurrence threshold of 100 was set; if words co-occurred more than 100 times, they were amalgamated into an n-gram. This threshold was chosen to get a balance considering the volume of 20,000 articles within the selected subset.

Regardless of the chosen n-gram strategy, lemmatization was applied to retain only nouns, adjectives, verbs, and adverbs from the original text. This was intended to assist the LDA model in identifying tokens that most effectively characterize distinct topics.

The pre-processing strategy played a vital role in the creation of corpuses and dictionaries, essential components for the upcoming section, in which LDA models will be applied to derive the most effective model. From the dictionary, words that appear with remarkable frequency (in over 80% of the articles) and those that were to rare (below 15 articles) were excised. This serves to eliminate noise, thereby enhancing our analysis.

Prior to delving into the models, the corpus underwent division into training, validation, and test sets (relativetely in 63%, 27% and 10%) . This division was imperative because perplexity capture how "surprised" a model is by new, unseen data. Therefore, this separation was crucial for obtaining a reliable estimation of perplexity as a metric.

C. Experiment

C.1. Introduction As previously mentioned, the experiment focuses on tuning hyperparameters to maximize LDA performance, divided into two distinct parts for clarity. In the initial phase, the objective was to optimize the alpha and beta parameters for each of the three corpuses and dictionaries obtained earlier (one with unigrams, one with bigrams, and one with trigrams). Subsequently, the three best models were compared using perplexity as a metric to identify the overall best-performing model. This process led to the determination of the optimal set of hyperparameters, including the most effective n-gram strategy, alpha, and beta values.

The second and final part of the development involved determining the optimal number of topics by maximizing a coherence score on the LDA model with the previously discovered hyperparameters. This step aimed to refine the model further, ensuring that the chosen hyperparameters were complemented by an appropriate number of topics, thus enhancing the overall performance of the LDA model.

C.2. Best N-grams Strategy with Alpha and Beta Optimization The optimization of alpha and beta parameters was conducted separately for each of the three n-gram strategies, and subsequently, the three best models were compared. Alpha, representing document-topic density, is characterized by higher values indicating that documents are composed of more topics, while lower values suggest that documents contain fewer topics. On the other hand, beta, representing topic-word density, is associated with higher values implying that topics consist of a larger number of words in the corpus, whereas lower values indicate that topics are composed of fewer words.

As previously anticipated, the comparison was performed using the perplexity metric. The models were trained on the training corpus, and the metric was computed using the validation crpus. Given that perplexity is monotonically decreasing concerning the number of topics, the evaluation of the best model was conducted by comparing perplexity trends for various steps and different configurations of alpha and beta. Specifically, experiments were carried out for 5, 10, 20, 50, and 100 topics. The alpha and beta grid of values was derived by combining the following ranges:

1. Alpha: [0.01, 0.31, 0.61, 0.91, "symmetric", "asymmetric"]
2. Beta: [0.01, 0.31, 0.61, 0.91, "symmetric"]

Where "symmetric" uses a fixed symmetric prior of $1/n\_topics$, and "asymmetric" uses a fixed normalized asymmetric prior of $1/(topic\_index + \sqrt{(num\_topics)})$.

This grid search allowed for an exploration of the parameter space, enabling the identification of the most optimal configuration for alpha and beta values for each n-gram strategy.

In the initial exploration, we visualized the perplexity trends for the unigram strategy with the aim of identifying a hyperparameter combination exhibiting a lower perplexity, indicative of superior performance. The visual inspection aim to select the set of parameters with a line consistently below the others. To facilitate a quantitative comparison, the selection of the best parameter set was executed by comparing the sum of perplexity values across different topic points. Following this approach, we

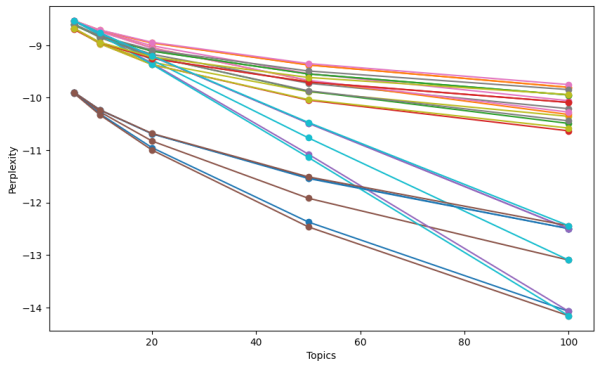

Figure 1 Unigram perplexity trends for different set of alpha and beta.

determined (Figure 1) that the optimal unigram model featured parameters alpha: 0.91 and beta: 0.01.

The same methodology was applied to ascertain the best set of alpha and beta for the bigram strategy. This analysis led to

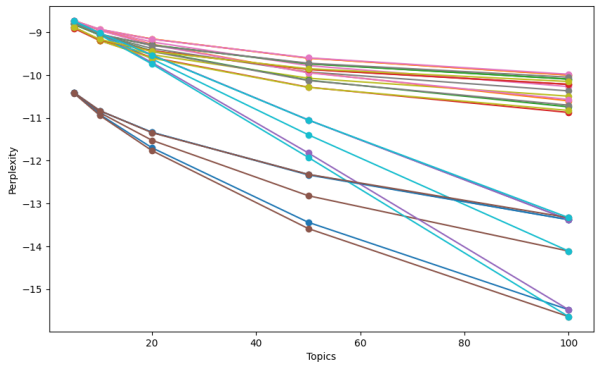

Figure 2 Bigram perplexity trends for different set of alpha and beta.

the identification of an LDA model with alpha: 0.91 and beta: 0.01 as the most effective for the bigram strategy (Figure 2).

Similarly, the trigram strategy underwent the same scrutiny. Ultimately, the trigram analysis resulted in an LDA model with alpha: 0.91 and beta: 0.01 being recognized as the most optimal (Figure 3).

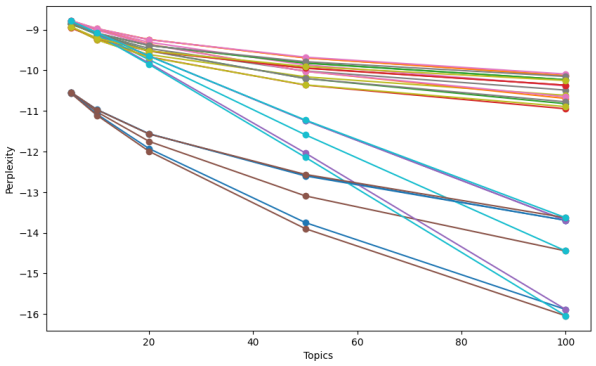

Figure 3 Trigram perplexity trends for different set of alpha and beta.

We can discern that, regardless of the chosen n-gram strategy, the optimal set of hyperparameters typically involves a higher alpha and a smaller beta. This observation implies that documents exhibit a tendency to encompass a greater diversity of topics, each consisting of a more concise selection of words.

The comparison of perplexity trends across various alpha and beta configurations for each n-gram strategy allowed the selection of the best-performing models. Now, it is time to compare the optimal LDA models chosen from the three strategies. The underlying logic for this comparison remains the same as the earlier approach.

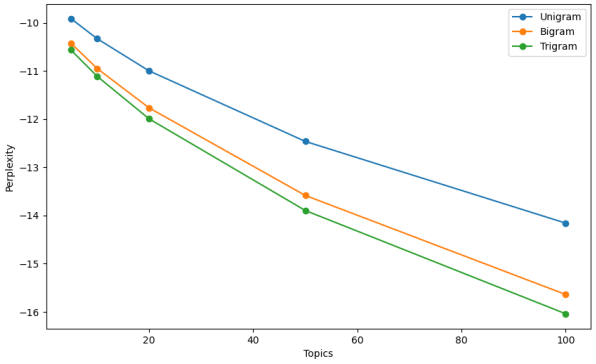

Figure 4 Perplexity comparison for three n-grams stratrgy.

From this analysis, we determined that the LDA model utilizing the corpus and dictionary with trigrams, along with alpha: 0.91 and beta: 0.01, outperformed the counterparts from the unigram and bigram strategies (Figure 4). Notably, there is an higher difference between unigram and bigram strategies, then the difference between bigram and trigram strategies. This observation suggests that text without n-grams may struggle to capture contextual information from the articles.

C.3. Optimal Number of Topics After obtaining the best possible hyperparameters for the LDA model, the next step involves tuning the optimal number of topics. As previously mentioned, the number of topics is selected by optimizing a coherence score, specifically the u_mass variant [13]. The selection of this particular coherence metric was motivated by its lower computational time requirements compared to the classic c_v and its robustness.

The optimal number of topics is expected to be relatively high

due to the substantial size of the dataset and the inherent diversity of articles within it. To explore a wide range of possibilities and ensure that the maximum value is identified, we calculated the coherence metric for a range of topics' number steps. This involved starting from 5 and incrementing by 5 until 100, then by 100 until reaching 1000. This approach aims to provide an overview and ascertain that the maximum coherence value identified is a global optimum.

The results of this analysis are depicted in the following figure. This procedure led to the determination that the optimal number of topics for the LDA model, with the previously selected hyperparameters, is 200.

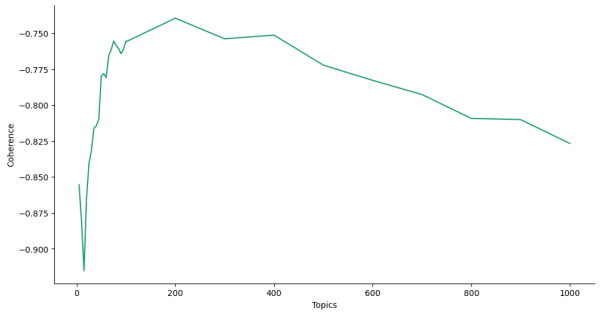

Figure 5 Perplexity coparison for three n-grams stratrgy using u_mass.

This model obtained a c_v score of 0.26, indicating a moderate level of coherence. Despite achieving the best model, the performance remains somewhat modest, likely owing to the inherent complexity of the task. Figure 6 presents the intertopic distance map. This provide insights into the relationships and similarities between different topics. The inter-topic distance map illustrates the distances or similarities between topics in a two-dimensional space. Short distances imply strong relationships or shared terms, whereas longer distances indicate less similarity. Similarity may suggest that those topics frequently co-occur in documents.

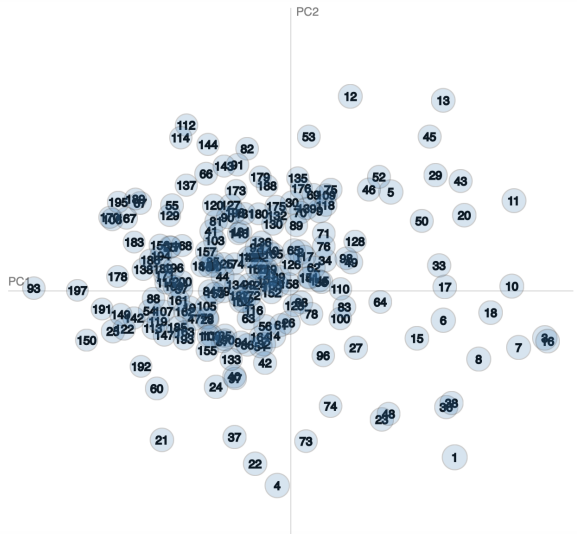

Figure 6 LDA intertopic distance map.

It is evident how most of the topics are grouped together. This might suggest that they not rightly specified.

We will now proceed to compare the results with those obtained by the BERT topic model. We expected to have better results form it.

D. Comparison

D.1. BERT model After assessing our top-performing LDA model, it's time to compare it with the BERT model.

BERT, a robust NLP model, has significantly advanced machine learning and language understanding. Developed by Google in 2018, BERT [3] is a transformer-based model known for its ability to capture contextual relationships within a given text. Its bidirectional approach considers both left and right surrounding words, allowing it to understand the complete context of a word. Due to BERT's success, researchers and practitioners continue to explore and enhance its capabilities for various NLP tasks, such as BerTopic for topic modeling [4]. This method offers several advantages over traditional techniques like LDA, including improved thematic precision, effective handling of large text corpora, capturing semantic relationships among words, precise control over the number of extracted topics, and easy optimization for specific text collections. In the case of this model, preprocessing is not required, and the input consists of straightforward articles. The BERT model used was in its default configuration [3]. BERT select 259 topic form the document, the Figure 7 show the intertopic distance map.

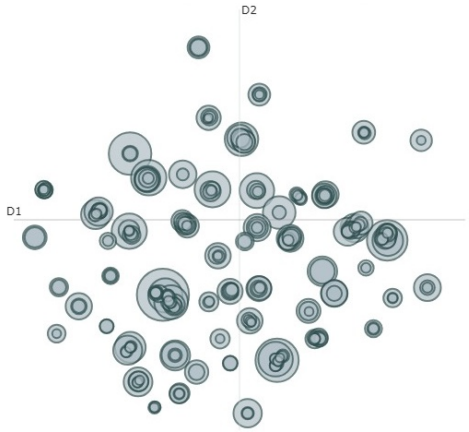

Figure 7 BERT intertopic distance map.

From this, it is evident that the topics are generally sparse, with some showing a tendency to co-occur, as indicated by the overlapping points. Additionally, I examined the top words for each topic, enabling a qualitative understanding of the quality of the topic detection.

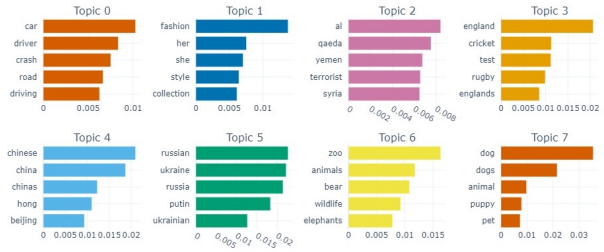

Figure 8 BERT topic word scores.

It is indeed clear how, for the topic in Figure 8, the words asso-

ciated with the topic are sensible and make it easy to understand the subject to which they refer.

D.2. Final Comparison To conclude this segment of our topic modeling exploration, we compare the derived LDA with BERT. As anticipated, we expect superior performance from the BERT model.

The comparison is twofold: firstly, we assess the coherence score of both models, focusing on the c_v value. Secondly, we provide a qualitative overview of select topics, examining how well-defined keywords characterize them. The comparative performance is presented in Table 1.

Table 1 Choerence score for the topic modeling models.

Models	c_v
LDA	0.26
BERTopic	0.74

From the table, it is evident that, despite all optimizations for LDA, it remains inferior to the BERT model. As expected, this result is corroborated by the qualitative examination of topic keywords. In BERT, the keywords distinctly define topics, readily interpretable at a glance. For instance, in Figure 8, Topic 5 with words like "russian," "ukraine," and "putin" unambiguously pertains to the Ukraine-Russia war. Similarly, Topic 6 with words like "zoo," "animals," "wildlife," and "elephant" clearly aligns with the zoo/wildlife theme. Conversely, examining LDA words for a topic reveals greater ambiguity, lacking clarity in topic definition. This discrepancy is illustrated in Figure 9.

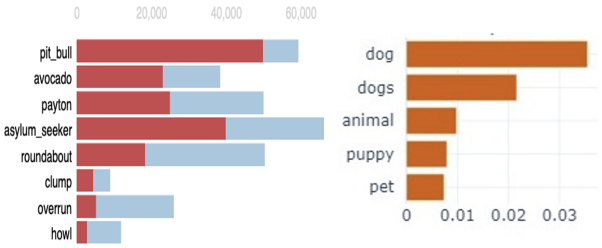

Figure 9 Top words visual comparison, LDA on the left and BERT on the right.

In the image, the topic in BERT is evidently comprehensible, whereas the one produced by LDA is somewhat unclear. Despite employing strategies that optimized the LDA model, its performance falls short of the BERTopic model. This discrepancy may be attributed to BERTopic's deep architecture, which facilitates a better understanding of semantics, context, and nuances within textual data, resulting in superior outcomes compared to LDA models. Another contributing factor could be the diverse and complex nature of the data sourced from different articles. Such intricacies pose challenges for traditional topic modeling strategies like LDA.

In conclusion, despite concerted efforts, the LDA model proves less efficient than the BERT model in grasping natural language processing intricacies and in precisely defining topics.

4. Summarization

A. Introduction

In this section, we present the methodology employed for conducting news articles summarisation on the previously discussed subset. Summarisation is the task of producing a shorter version of a document while preserving its important information. In the context of news articles, summarisation helps provide a quick overview of the main points, allowing readers to grasp the essential information and then deciding if it is worth delving deeper into the full article. There are broadly two different approaches that are used for text summarization: Extractive and Abstractive. In this section we will present and apply both methods:

- Extractive summarisation through **TexRank algorithm**.
- Abstractive summarisation through **T5 Model**.

We will assess and compare these approaches using various evaluation metrics, including ROUGE, BLEU, METEOR, and BERTScore.

B. Extractive summarization with TextRank

Extractive summarization is a text summarization technique that involves identifying (by ranking) the most important sentences from the original text through a score, and extracting only those from the text.

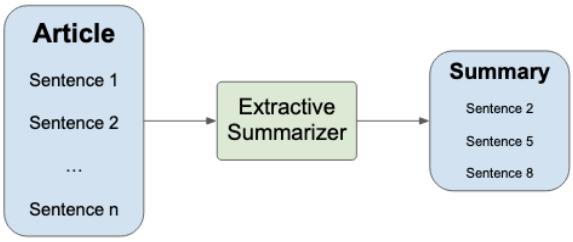

Figure 10 Schematic representation of the extractive summarization task.

Extractive methods aim to condense articles by choosing a subset of words that capture the essential information. This method assigns weights to crucial parts of sentences and utilizes them to construct the summary. Various algorithms and techniques come into play to determine sentence weights, ranking them based on significance and similarity to one another. In this project the summary will be extracted exploiting the TextRank algorithm [8].

TextRank is an algorithm for extractive summarization and keyword extraction in natural language processing. The algorithm is based on the PageRank algorithm used by Google for ranking web pages. TextRank treats each sentence in a document as a node in a graph and assigns weights to the edges between sentences based on their similarity. The algorithm then iteratively updates the importance scores of each sentence until convergence. The extractive summarization consists of four phases:

1. Pre-process the text and split the news article in sentences.
2. Creating an intermediate representation of the input which can capture the key aspects of the text.
3. Scoring the sentences based on that representation.
4. Selecting the summary according to the scored sentences.

B.1. Pre-processing Before applying the TextRank algorithm, it's essential to pre-process the articles to enhance their compatibility with the algorithm. The first step involves dividing each article into sentences. This sentence tokenization is performed using the `sent_tokenize` function from `nltk` Python library. The list of tokenized sentences is then duplicated; the first set is kept as it is, and it will be used once computed the sentence ranking for composing the final summary of the article. The second copy will be used for performing the actual ranking and the sentences need to be further processed. In this second set, every non-alphanumeric character, including punctuation, is removed. Additionally the text is converted to lowercase and stop words were eliminated using a list of English stop words from the `nltk` Python library. Eliminating stop words refines the sentence representation, focusing on words that significantly contribute to the article's meaning.

B.2. Intermediate representation Once the input is pre-processed, the next step is to create an intermediate representation aiming to capture the key aspects of the text. There are several approaches to create an intermediate representation, for this project it was decided to use the cosine-similarity between GLOVE100d [10] vector sentences. Delvin into details, each sentence is represented as a 100-dimensional vector, obtained by averaging the word embeddings of the unique words in the sentence. The similarity matrix is then created by calculating the cosine similarity between these sentence vectors Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space. It is a metric used to determine how similar two vectors are, with a value of 1 indicating identical vectors and a value of 0 indicating completely dissimilar vectors.

There are various methods available for obtaining vector representations of sentences, including Tf-Idf. However, in this project, the decision was made to employ GLOVE word embeddings. This choice is attributed to the embeddings' effectiveness in capturing both syntactic (grammar and structure) and semantic (meaning) relationships among words.

B.3. PageRank algorithm and summary selection Once the similarity matrix is computed it is then converted into a graph exploiting the `DiGraph()` function from `networkx` Python library; sentences represent the nodes of the graph, and the relationships between them (similarity) as edges. The subsequent phase involves identifying the most important sentences within the articles, this is achieved by implementing the PageRank algorithm on the graph. For this project, the algorithm from the `networkx` Python library was employed. PageRank allocates a score to each node, taking into account the graph's structure and the significance of its connected nodes. Nodes having higher PageRank scores are regarded as more pivotal within the graph. In extractive summarization, these nodes could correspond to sentences that contain the most important information about the article.

After ranking all sentences based on their computed scores, the ultimate summary is generated by choosing the optimal combination of significant sentences. Various approaches exist for sentence selection, and for this project, the "best n" approach was adopted. This method involves selecting the top n most important sentences. Through an examination of the word token distribution in the ground truth, the decision was made to opt for the **top three sentences** to maintain a similar length between the summary and the ground truth.

C. Abstractive summarization with T5

Abstractive summarization is a technique where a model generates a concise and coherent summary of a given text that goes beyond mere extraction and rearrangement of sentences.

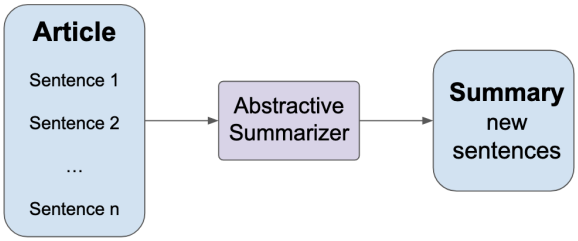

Figure 11 Schematic representation of the abstractive summarization task.

In contrast to extractive summarization, which selects and compiles existing sentences from the source text, abstractive summarization involves a deeper understanding of the content and the ability to generate new, rephrased sentences to convey the essential meaning. This technique entails identifying key pieces, interpreting the context, and recreating them in a new way. Stateofthe-art abstractive summarization models often leverage transformer-based architectures. These models are pretrained on large datasets and finetuned for specific summarization tasks.

T5 [11], or Text-to-Text Transfer Transformer, is a transformer-based neural network architecture developed by Google Research. T5 is a versatile architecture designed to handle various natural language processing tasks such as text summarization. In this project the abstractive summarization task will be performed with the pre-trained T5-small model, through the HuggingFace API [15].

C.1. Inference The summarization task can be clarified through the following schematic representation:

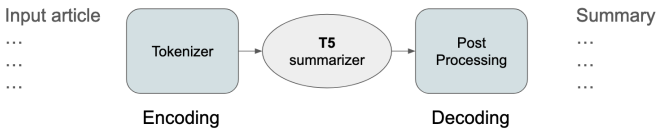

Figure 12 Schematic representation of the inferece task: text summarization with T5.

Initially, the input article must undergo preparation for input into the summarizer. This involves a two-step process. The first step is to add a prefix to the article: "Summarize this article:". This prefix provides explicit instructions to the T5 model, indicating that the task to be performed is text summarization. Following the addition of the prefix, the second step involves translating the text into a format processable by the model. This is handled by the built-in tokenizer from the HuggingFace API, `AutoTokenizer.from_pretrained("t5-small")` by specifying the model of interest. The specific tokenizer used with T5 is the SentencePiece tokenizer [5], which is a data-driven, unsupervised text tokenizer and detokenizer primarily designed for neural network-based text generation tasks.

At this point, the input article is ready to be fed into the T5 model for summarization. The model, once loaded, performs

the summarization task and outputs the summary. The inference is executed using the `model.generate()` method. Parameters such as the minimum and maximum summary length can be set; in this case, summaries are generated within the range of 40-60 tokens based on an examination of the word token distribution in the ground truth.

However, the summary produced by the model is still encoded in numbers and needs to be converted into natural language text. This post-processing step is managed by the `tokenizer.decode()` method. This method not only converts the indices back to tokens but also groups together tokens that were part of the same words to produce a coherent sentence.

To enhance efficiency and manage computational constraints, the described procedure is executed in batches, each consisting of 32 articles. This batching approach allows the model to process multiple articles simultaneously. A comprehensive function has been designed to encompass the entire inference process in batches. It takes a set of articles as input and generates the corresponding summaries as output. In each batch, the 32 articles are encoded, processed by the model, and then decoded to produce the summarizations.

A crucial aspect of the batch process is managing the variation in lengths of multiple sequences. To address this, padding is employed to ensure uniform tensor shapes. Padding involves adding a special word, known as the padding token, to shorter sentences within a batch to match the length of the longest sentence. This uniformity is essential for parallel computation on GPUs, which expect inputs of consistent shapes.

D. Evaluation

Due to computational constraints—such as limited processing power—we randomly selected a sample of 2000 articles from the complete set for evaluation.

In the field of text summarization, evaluating the performance of summarization models is a complex task that involves assessing various aspects of the generated summaries. Summarization metrics are employed to measure how well a generated summary aligns with a reference summary also called ground truth.

To evaluate the results a combination of multiple metrics are used aiming to provide a more comprehensive and robust evaluation:

- ROUGE [6]
- BLEU [9]
- METEOR [1]
- BERT_score [16]

The results are shown in the tables below. The metrics were calculated by evaluating the entire sample set. To provide deeper insight into the results, we have included the standard deviation, offering a gauge of the variability within the sample set.

The ROUGE scores provided in Table 2 stands for Recall-Oriented Understudy for Gisting Evaluation, compares the overlap of n-grams between the computer-generated summaries and a set of reference summaries. The table presents three variants of ROUGE scores: ROUGE-1 and ROUGE-2, which measure the overlap of 1-gram (individual words) and 2-grams (pairs of consecutive words), respectively, and ROUGE-Lsum, which focuses on the longest common subsequence of sentence units, providing an insight into the coherence and order of the content in the summaries. From the data, the abstractive T5-small model consistently outperforms the extractive TextRank algorithm across all metrics. Most notably, the ROUGE-Lsum score for T5-small exceeds that of TextRank by 7 percentage points, indicating a significant improvement in capturing the sequence of ideas as presented in the original text.

Table 2 ROUGE score for each summarization technique

	ROUGE		
	Rouge1	Rouge2	RougeLsum
Extractive TextRank	26.49($\pm$9.38)	9.03($\pm$7.2)	21.38 $\pm$ 8.25)
Abstractive T5-small	**32.16**($\pm$13.88)	**12.95**($\pm$11.9)	**27.93**($\pm$12.97)

Table 3 illustrates the BLEU scores for the extractive TextRank and abstractive T5-small summarization techniques. Bilingual Evaluation Understudy serves as a metric for assessing the effectiveness of machine translation between languages, yet its applicability extends to the evaluation of automatic text summarization. This metric compares the summarized text with a reference (ground truth), assigning a score based on word overlap between the two. The BLEU score, ranging from 0 to 1, indicates the quality of the summary, with higher scores reflecting better alignment with the reference. The evaluation also includes detailed insights into Unigram and Bigram overlaps, contributing to the overall BLEU score.

The overall BLEU score for the T5-small model is significantly higher than that of TextRank, indicating a more precise match at both the Unigram and Bigram levels. Specifically, the T5-small model exhibits an overall BLEU score that is nearly twice that of TextRank, with a marked improvement of approximately 20 percentage points in Unigram precision and 8 percentage points in Bigram precision.

While BLEU is a widely accepted metric, it has its limitations; it is sensitive to word order and does not always effectively capture semantic similarity. Therefore, the assessment should be viewed in context, taking into account additional metrics for a more comprehensive evaluation.

Table 3 BLEU score for each summarization technique

	BLEU		
	Overall	Unigram	Bigram
Extractive TextRank	4.97($\pm$4.2)	17.47($\pm$7.43)	5.89($\pm$4.77)
Abstractive T5-small	**9.46**($\pm$8.47)	**37.17**($\pm$14.81)	**14.62**($\pm$12.16)

Table 4 presents the METEOR scores for both extractive TextRank and abstractive T5-small summarization techniques. METEOR, or Metric for Evaluation of Translation with Explicit Ordering, extends beyond mere lexical matching to consider grammar and semantic accuracy in translation and text generation tasks. This evaluation metric operates on a 0 to 1 scale, where higher scores denote better alignment with the reference summary.

Interestingly, the extractive TextRank method outperforms the abstractive T5-small model with a METEOR score of 30.35, contrary to the trends observed in other metrics. This could be attributed to the nature of extractive summarization, which inherently preserves the original syntax and semantics by selecting sentences directly from the source text. Consequently, it often maintains grammatical integrity and semantic coherence more closely aligned with the original author's language, which is reflected in the METEOR score. On the other hand, the abstractive method, which constructs new sentences, may not mirror

the reference as closely, potentially leading to lower METEOR scores due to divergences in grammar and choice of semantics.

Table 4 METEOR score for each summarization technique

	METEOR
	Overall
Extractive TextRank	**30.35**($\pm$10.46)
Abstractive T5-small	24.5($\pm$13.04)

Taking into account the BERTScore metric, which distinguishes itself from the previous metrics by harnessing contextual embeddings from pre-trained BERT models, introduces a heightened ability to incorporate semantic and contextual similarity. As illustrated in Table 5, both the extractive TextRank and abstractive T5-small models perform commendably, each surpassing the BERTScore threshold of 75. The T5-small model slightly leads with a score of 78.55, indicating a marginally better semantic and contextual alignment with the reference summaries. Despite the slight differences in F1, Recall, and Precision between the methods, it is clear that both models attain a notable degree of semantic understanding and contextual relevance in their summaries.

Table 5 Bert score for each summarization technique

	BERT_score		
	F1	Recall	Precision
Extractive TextRank	77.51	**81.39**	74.05
Abstractive T5-small	**78.55**	77.66	**79.55**

The analysis across four distinct metrics—ROUGE, BLEU, METEOR, and BERTScore—has provided us with a comprehensive picture of the summarization capabilities of both the extractive TextRank and the abstractive T5-small models.

The performance of the T5-small model in BLEU and ROUGE scores indicates its proficiency in generating summaries that are lexically closer to reference summaries, perhaps due to its ability to construct new and relevant sentences. Conversely, the extractive TextRank method exhibited a notable performance in METEOR scoring, suggesting that the preservation of original text structures results in summaries with better grammatical and semantic alignment with source texts.

The BERTScore evaluations revealed that both models possess strong semantic and contextual alignment capabilities, with T5-small showing a slight edge.

5. Conclusion and Further developments

In conclusion, this project extensively explored the realms of topic modeling and text summarization, utilizing the CNN-DailyMail News dataset. With a focus on simplicity and effectiveness, we delved into various techniques within these two branches of text mining.

Our investigation revealed the enhancement of topic modeling with LDA, yet a comparison with the BERT model demonstrated that, at least for this specific topic and dataset, the deep approach proved more efficient. It resulted in better performance and more precise topic specification, especially under visual inspection.

Simultaneously, our project explored text summarization. While the METEOR score metric favored the Extractive method, the preference for it may be attributed to the fact that the abstractive method, which constructs new sentences, might not closely mirror the reference. Despite this, evaluating F1 Recall and Precision indicated that the T5-small model outperformed in BLEU and ROUGE scores, showcasing its proficiency in generating summaries lexically closer to reference summaries.

To conclude this project, we imagined a future development that could integrate the insights gained from both studies. The optimal topic modeling technique and the most effective summarization method could be combined for use in a comprehensive article and news website. The topics identified through topic modeling could be employed in a labeling system, enabling users to apply filters. These filters could be strict, focusing on the most prevalent topic, or shallow, where each article is composed of multiple topics, with the first ones retrieved haveing higher presence of a prticular topic. Considering the challenge users face in selecting articles to read, the summarization could play a crucial role by providing a brief preview of the articles. This would prove valuable as users may not wish to read the entire article to determine their interest, and relying solely on the title might be misleading due to clickbait strategies or overly sensationalized titles.

In summary, this project demonstrated an effective application of text mining analysis through both topic modeling and summarization. The presented system could potentially offer a valuable solution for users seeking a more informed approach to accessing news and articles.

References

[1] Satanjeev Banerjee and Alon Lavie. “METEOR: An automatic metric for MT evaluation with improved correlation with human judgments”. In: *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*. 2005.

[2] David M Blei, Andrew Y Ng, and Michael I Jordan. “Latent dirichlet allocation”. In: *Journal of machine Learning research* (2003).

[3] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).

[4] Maarten Grootendorst. “BERTopic: Neural topic modeling with a class-based TF-IDF procedure”. In: *arXiv preprint arXiv:2203.05794* (2022).

[5] Taku Kudo and John Richardson. “Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing”. In: *arXiv* (2018).

[6] Chin-Yew Lin. “Rouge: A package for automatic evaluation of summaries”. In: *Text summarization branches out*. 2004.

[7] Edward Loper and Steven Bird. “Nltk: The natural language toolkit”. In: *arXiv preprint cs/0205028* (2002).

[8] Rada Mihalcea and Paul Tarau. “Textrank: Bringing order into text”. In: *Proceedings of the 2004 conference on empirical methods in natural language processing*. 2004.

[9] Kishore Papineni et al. “Bleu: a method for automatic evaluation of machine translation”. In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 2002.

[10] Jeffrey Pennington, Richard Socher, and Christopher D Manning. "Glove: Global vectors for word representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 2014.

[11] Colin Raffel, Noam Shazeer, Roberts, et al. "Exploring the limits of transfer learning with a unified text-to-text transformer". In: *The Journal of Machine Learning Research* (2020).

[12] Radim Řehřek, Petr Sojka, et al. "Gensim—statistical semantics in python". In: *Retrieved from genism. org* (2011).

[13] Michael Röder, Andreas Both, and Alexander Hinneburg. "Exploring the space of topic coherence measures". In: *Proceedings of the eighth ACM international conference on Web search and data mining*. 2015.

[14] Abigail See, Peter J Liu, and Christopher D Manning. "Get to the point: Summarization with pointer-generator networks". In: *arXiv* (2017).

[15] Thomas Wolf, Lysandre Debut, et al. "Huggingface's transformers: State-of-the-art natural language processing". In: *arXiv* (2019).

[16] Tianyi Zhang et al. "Bertscore: Evaluating text generation with bert". In: *arXiv* (2019).