Text Mining and Search

Project Report

WikiHow Articles:

A Study on Topic Modeling and Text Summarization

Authors:
Tariq Baghrous - 904027 - t.baghrous@campus.unimib.it
Roberto Ferrari - 852220 - r.ferrari22@campus.unimib.it
Luca Iarocci - 894066 - l.iarocci@campus.unimib.it

Master's Degree in Data Science
Università degli Studi di Milano-Bicocca
2023/2024 Academic Year

Contents

1 Introduction

The exponential proliferation of textual data in the digital era has presented both challenges and opportunities for extracting meaningful insights and knowledge. This project is dedicated to a comprehensive text mining procedure aimed at extracting valuable information from a vast collection of articles.

In this paper, we present WikiHow, a dataset of more than 230,000 articles and summary pairs extracted and constructed from an online knowledge base written by different human authors.

The datasets provided underwent a preparation procedure, ensuring their uniformity and efficacy in alignment with the project's goals. Indeed, the text mining tasks we employed require a specific type of data modeling to utilize them effectively and generate accurate results.

Following the data pre-processing, the project shifted its focus towards applying the following text mining tasks: Topic Modeling and Text Summarization. Each task required a specific Text Representation in formats conducive to computational analysis.

Lastly, We utilize a spectrum of evaluation metrics to assess the performance of our models and ensure the relevance and interpretability of the extracted insights.

1.1 Datasets description

The WikiHow knowledge base contains online articles describing a procedural task about various topics with multiple methods or steps and new articles are added to it regularly. Each article consists of a title starting with "How to" and a short description of the article. There are two types of articles: the first type describes single-method tasks in different steps, while the second type represents multiple steps of different methods for a task. Each step description starts with a bold line summarizing that step and it's followed by a more detailed explanation.

Each article consists of multiple paragraphs and each paragraph starts with a sentence summarizing it. By merging the paragraphs to form the article and the paragraph outlines to form the summary, the resulting version of the dataset contains more than 200,000 long-sequence pairs.

There are two separate data files containing the articles and their summaries adopted for the two different text mining tasks: *wikihowAll.csv* and *wikihowSep.csv*.

1.1.1 wikihowAll.csv

The *wikihowAll.csv* file consists of the concatenation of all paragraphs as articles and the bold lines as reference summaries.

This dataset was implemented for the Topic Modeling task, and it has the following attributes:

- **Title:** the title of the article as it appears on the WikiHow knowledge base
- **Headline:** the concatenation of all the bold lines (the summary sentences) of all the paragraphs to serve as the reference summary
- **Text:** the concatenation of all paragraphs (except the bold lines) to generate the article to be summarized

1.1.2 wikihowSep.csv

The *wikihowSep.csv* file contains separate paragraphs as the articles and the bold lines corresponding to each paragraph as the reference summary.

This dataset was implemented for the Text Summarization task, and it has the following attributes:

- **Title:** the title of the article as it appears on the WikiHow knowledge base
- **Overview:** the introduction section of the WikiHow articles represented before the paragraphs corresponding to procedures
- **Headline:** the bold line (the summary sentence) of the paragraph to serve as the reference summary
- **Text:** the paragraph (except the bold line) to generate the article to be summarized

2 Text Pre-processing

2.1 wikihowAll.csv

2.1.1 Normalization

The collected dataset underwent a meticulous pre-processing phase to cleanse and normalize the text. The process consistend in the following phases, in order:

- **Feature selection:** extraction of the relevant columns 'headline' and 'text'
- **Completeness:** Null values check
- **Case folding:** converted the text in lower case
- **Consistency:** removed duplicated redords, reducing the dataframe from 215365 to 214458 rows
- **URLs and links:** removed URLs that start with "http://" or "https://" or "www." and/or end with ".com"
- **HTML tags:** removed HTML tags using the *BeautifulSoup* library
- **Contractions:** expanded contractions in text data to their full forms
- **Numbers:** removed numerical digits
- **Punctuation and symbols:** removed special characters and punctuation except for the periods and \n symbols
- **White-spaces:** removed extra white spaces

2.1.2 Tokenization

The next phase consisted in tokenize the 'text' and 'headline' into sentences using the *sent_tokenize* function from the NLTK library. We created two new columns in the DataFrame called 'headline_sentences' and 'text_sentences', containing the created tokens. In addition, these two attributes will be the main subject of the text summarization task.

2.1.3 Stop words removal

Then, we enhanced the quality of textual data by eliminating common stop words, which are words that often occur frequently but contribute little to the semantic meaning of the text. This was possible through the NTLK library, allowing us to tokenize and filter words.

2.1.4 Stemmatization and Lemmatization

During this phase, four additional columns were incorporated into the dataset: two containing the 'text' and 'headline' subjected to stemmatization, a process that reduces words to their root form using the Porter Stemmer technique; and another containing text subjected to lemmatization, which reduces words to their base form or lemma using WordNet, an English lexical database.

Stemming carries the risk of semantic information loss but offers memory efficiency gains. Given the milder nature of lemmatization and its potential for improved accuracy

and comprehensiveness, only text processed through lemmatization was selected for the further topic modeling analysis.

Following this process, a file named *wikiall_processed.csv* was created to facilitate easier reading for subsequent steps.

2.2 wikihowSep.csv

Unlike the wikihowAll dataset, wikihowSep underwent a simpler pre-processing phase. This was due to the fact that, as it will be discussed in the Text Summarization section in further detail, the summarization task was implemented using the Python library Sumy.

Sumy implements various extractive summarization techniques and is designed to accept plain text samples as input. It internally handles all major pre-processing steps required by the summarization task via built-in functions and objects. For this reason and for the wikihowSep dataset only, we decided to leverage the library's capabilities and use them to perform: text tokenization, stop-words removal, and stemming. Consequently, the only text processing tasks explicitly performed by us on the dataset were mainly related to its normalization and aimed primarily to facilitate the tasks performed internally by Sumy.

Similarly to the wikihowall dataset, a preliminary data pre-processing phase was dedicated to the removal of incomplete or duplicate records and the selection of relevant features. For the summarization task these were: title, text, and headline.

Moving onto text normalization, the aim of this phase was to facilitate the machine interpretability of text samples without hindering their interpretability by humans. This approach was dictated by the extractive nature of the algorithms used, which work by ranking and selecting the most informative sentences in a text and by piecing them together to create a summary. Hence, the starting text must be itself understandable to produce understandable summaries. To achieve this we processed both Headline and Text by:

- removing URLs and links.
- removing unwanted or repeated stop characters.
- removing repeated punctuation marks.
- removing sequences of punctuation marks separated only by white spaces.
- enforcing proper spacing before and after punctuation marks.

This set of operations proved especially effective in helping Sumy's built-in tokenizer to correctly segment the text sample and in doing so to improve Sumy's overall quality of the text pre-processing pipeline results.

The results of the pre-processing of wikihowSep were saved into a .csv file named *wikisep_processed.csv* to make them readly available for all subsequent operations.

3 Topic Modeling

Topic modeling is a pivotal technique in text mining and search, enabling the automated discovery of latent topics within large collections of textual data.

By employing probabilistic models and statistical algorithms, topic modeling algorithms analyze the co-occurrence patterns of words across documents to identify clusters of related terms that represent coherent themes or topics. These topics encapsulate the underlying concepts and themes present in the corpus, offering valuable insights into the content and meaning of the text.

In this study, we implement the Latent Semantic Analysis (LSA) and the Latent Dirichlet Allocation (LDA) through respectively "scikit-learn" and "gensim" libraries, using as input the *wikiall_processed.csv* dataset composed of 214458 articles of heterogeneous nature.

To assess the optimal number of topics, we compute the following coherence evaluation measures:

- **UMASS coherence:** metric based on the co-occurrence of terms within documents in the corpus. It can be sensitive to word frequency and corpus size.
- **Cv coherence:** metric based on a sliding window, one-set segmentation of the top words, and an indirect confirmation measure that uses normalized point-wise mutual information (NPMI) and cosine similarity.

Higher values of these metrics mean a greater measure of semantic similarity between the top words in our topic, which indicates better model performance.

3.1 Text Representation

Since both LDA and LSA were applied in this study, two distinct text representation techniques were used: the Bag of Words (BoW) model, implemented using the Gensim library, is a simple yet effective way to represent text data for LDA, which relies on word frequencies to identify latent topics. It disregards the sequential order of words in the text and focuses solely on their frequency of occurrence within the document; on the other hand the TF-IDF matrix was utilized for the application of the LSA model. Both models, though, underwent the same pre-processing steps.

First, bigrams and trigrams individuation was conducted, in order to capture more intricate relationships and structures between words, rather than implementing only single terms, which can cause the loss of potential topics. Bigrams were defined as pairs of words that occurred, at minimum, 300 times in the documents, while trigrams were defined as group of three words that appeared, at minimum, 80 times in all documents: this operation resulted in 12434 bigrams, such as 'physical_activity', 'dental_problem', 'magnetic_field', and 514 trigrams, such as 'fast_food_restaurant' and 'maintaining_eye_contact'. The last step of this process was to apply these structures to the corpus.

Then, a dictionary was constructed from the lemmatized articles. This dictionary serves as a mapping of words to unique numerical identifiers. To ensure the dictionary's relevance and manageability, certain terms were filtered out based on their frequency of occurrence in the corpus. Terms appearing in over 5% of the documents (about 10700) or those showing in less than 0.1% (about 200 documents) were excluded from the dictionary, resulting in 19413 unique words.

Finally, the Bow model and the TF-IDF matrix were applied, concluding this text representation phase for LDA and LSA implementation.

3.2 Latent Semantic Analysis (LSA)

Latent Semantic Analysis (LSA) is a technique used in natural language processing and information retrieval to uncover hidden patterns in a collection of documents. It analyzes relationships between words and documents by creating a term-document matrix, which represents how often each word appears in each document.

By using the Singular Value Decomposition (SVD), LSA reduces the dimensions of this matrix. This helps to identify underlying topics by grouping together words that frequently appear in similar contexts, even if they don't occur together in any specific document. Essentially, it captures the semantic meaning of words and documents.

LSA is particularly good at handling synonyms (different words with similar meanings) and polysemy (a single word with multiple meanings) by considering the context in which words appear. This makes it a powerful tool for Topic Modeling.

Choosing the right number of topics is crucial for LSA's success. Too many dimensions can introduce noise, while too few can oversimplify the data. For this purpose, the model was first tested on a sample (5% of the corpus) within a topic range of 5-30, while computing the coherence metrics, Cv coherence and UMASS coherence, for each number of topics. the results are displayed in Figure 1:

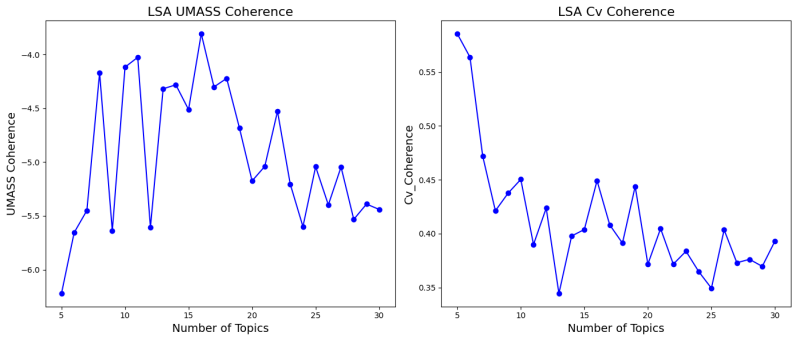


Figure 1: *Comparison of Cv and UMASS Coherence through a range of topics between 5 and 30, taking as input a sample of 5% of the corpus*

Since the highest values of the metrics were found in the lower end of the topics range, an additional iteration was computed, this time over the entire corpus in a range of 7-15 topics, the results of which is displayed in Figure 2.

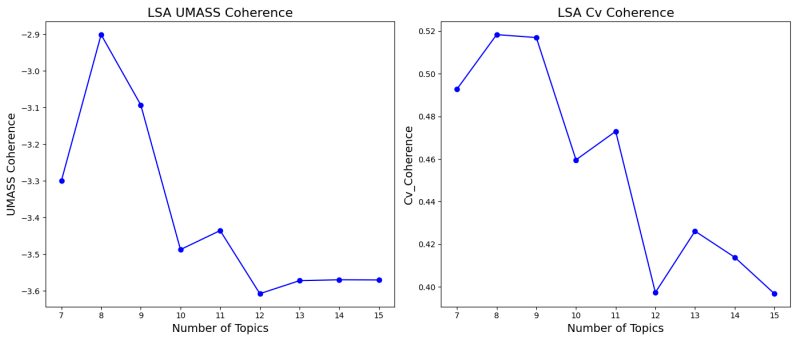


Figure 2: *Comparison of Cv and UMASS Coherence through a range of topics between 7 and 15, taking as input the entire corpus*

The graphics shows that with 8 topics, we get the best Cv score and the best UMASS score, respectively 0.518338 and -2.900979. This makes the choice of 8 topics the most suitable.

To better identify the topics, we display the most relevant words for each one by using word clouds (Figure 3), which provide a simple and intuitive way to visualize the words and the associated weights.

Figure 3: *Word clouds for each topic identified by the LSA model*

By looking at the figures, we could assume the main theme for each topic to be:

1. **Family and Education**: parent, partner, conversation, class, teacher.
2. **Baking/Gardening**: plant, mixture, dough, cake, stir, bowl, stain, soil
3. **Smartphone Usage**: icon, iPhone, device, folder, photo, account, menu
4. **Baking**: dough, cake, mixture, butter, cook, flour
5. **Animal Care**: horse, baby, puppy, rabbit, stain
6. **Household Care**: baby, fabric, paint, nail, puppy
7. **Household Activities**: paint, stain, fabric, nail, stitch
8. **Dog Care**: puppy, crate, breeder, breed, training

3.3 Latent Dirichlet Allocation (LDA)

Latent Dirichlet Allocation model is a generative probabilistic model used to classify text contained in a corpus into a specified number of topics. The model assumes that documents are a collection of topics and topics are a mixture of words. For each document, the model applies a probability distribution over topics, showing which topic is more relevant, and for each topic the model applies another probability distribution, this time, over words, showing the ones that most characterize the topic.

Given the heterogeneous nature of the input data, the hyper parameters 'alpha' and 'eta', which measure the distribution of topics and words, respectively, are both set to 'symmetric' rather than a specified value, meaning that, for every document, all topics have the same probability as priors - i.e. it is assumed that every document has a uniform distribution over topics - and, for every topic, all words have the same probability as priors - i.e. it is assumed that every topic has a uniform distribution over words.

The number of topics, instead, was established iteratively by testing models with different topic ranges and comparing the resulting coherence metrics, Cv coherence and UMASS coherence, which measure the clear separation and interpretability of topics, and the internal coherence of each topic, respectively.

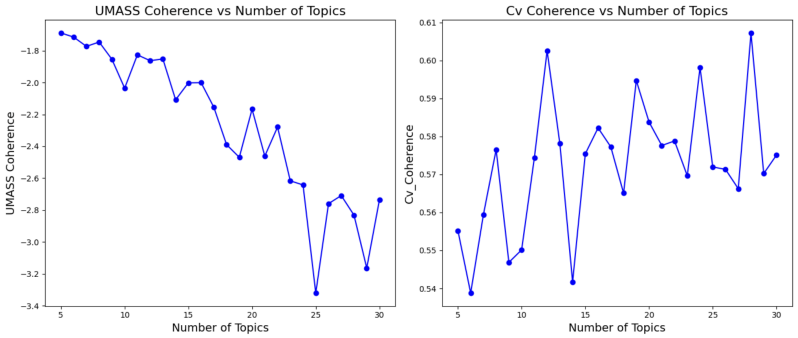


Figure 4: *Comparison of Cv and UMASS Coherence through a higher range of topics, taking as input a sample of 5% of all documents, in a range between 5 and 30.*

Through various tests with samples, 5% of all documents, it has been observed that UMASS coherence index tend to give worse results as the number of topics went above 15, while the Cv coherence index tend to fluctuate and get progressively better as the number of topics increases (figure 4).

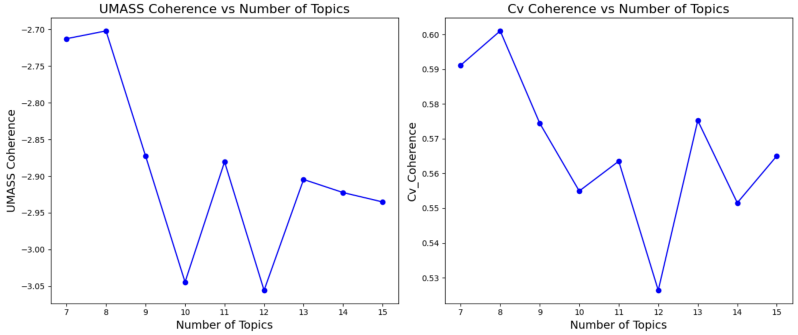


Figure 5: *Comparison of Cv and UMASS Coherence metrics through different number of topics, in a range between 7 and 15.*

Since the model is applied to the entirety of the corpus, and given its various nature, a more fitting range has been established, with the intent to strike a balance between the two coherence measures and the computational resources available. The following plot shows the results obtained (figure 5).

The choice for the optimal number of topics was straightforward, since both metrics registered the best values in correspondence of 8, where a Cv coherence of 0.60 and a UMASS coherence of -2.70 were achieved. Given this fixed number of topics, a second training phase is performed and the results were displayed with WordClouds, one for each topic (figure 6).

Figure 6: *Word clouds for each topic identified by the LDA model*

Also in this case, we could assume the main theme for each topic:

1. **Stitching**: stitch, string, length, needle, hook, loop.
2. **Gardening**: plant, garden, tree, soil, root, ground, battery, cable, install.
3. **Farming/Gardening**: stem, seed, growth, fertilizer, compost, fruit, vegetable, watering.
4. **Music/Art**: chord, drum, card, image, artist, letter, picture.
5. **Music/Entertainment**: song, music, character, guitar, beat, audience.
6. **Drawing/Creative**: fabric, paint, engine, design.
7. **Investing**: business, sale, sell, price, vehicle, payment.
8. **DIY Activities/Creative**: fabric, wire, paint, metal, glue, wood, tape, hole.

4 Text Summarization

The best automatic summarization techniques and their effectiveness to deliver critical information of a given text are highly dependent on both the nature of the text itself and the expectation of the user. In this regard, the WikiHow corpus posed two unique challenges: on the one hand, the articles presented a wide range of styles, subjects and authors, making any form of generalization extremely difficult; on the other hand, reference summaries were artificially created, generally extremely concise, not conveying what they should have, and hard to compare with results of automatic summarization. In this project we implemented and evaluated the performance of a series of extractive summarization techniques, both unchanged and modified in an attempt to better suit the corpus' characteristics, aiming to study their behavior when faced with these challenges.

The results hereby presented were obtained from a sub-sample of the original dataset selected by record characteristics and statistical significance. The differences in the articles' length and structure were taken into consideration and explored in depth to provide both an overall picture of the corpus' characteristics and define valid criteria for the selection of the aforementioned sample. Furthermore, the choice of working on a sample was justified by the unsupervised nature of the implemented algorithms. There was no point in running models over the entire datasets beyond improving the statistical significance of the results, which were already satisfactory.

4.1 Data Exploration

Thanks to the dataset structure we were able to analyze the corpus both on a paragraph and an article basis. In particular we focused mainly on studying the length in terms of words and sentences of both articles and paragraphs, as well as their respective summaries.

Starting from the analysis of paragraphs' characteristics we decided to drop from the dataset those records which consisted of less than 5 words or that were wrongly tokenized and presented no sentences, resulting in a loss of about 3.71% total of records. We then applied a second filter based on the statistical characteristics of the records, dropping all records whose text's or headline's lengths in words or sentences exceeded the mean by more than 3 standard deviations; this resulted in a loss of about 3.5% of the records.

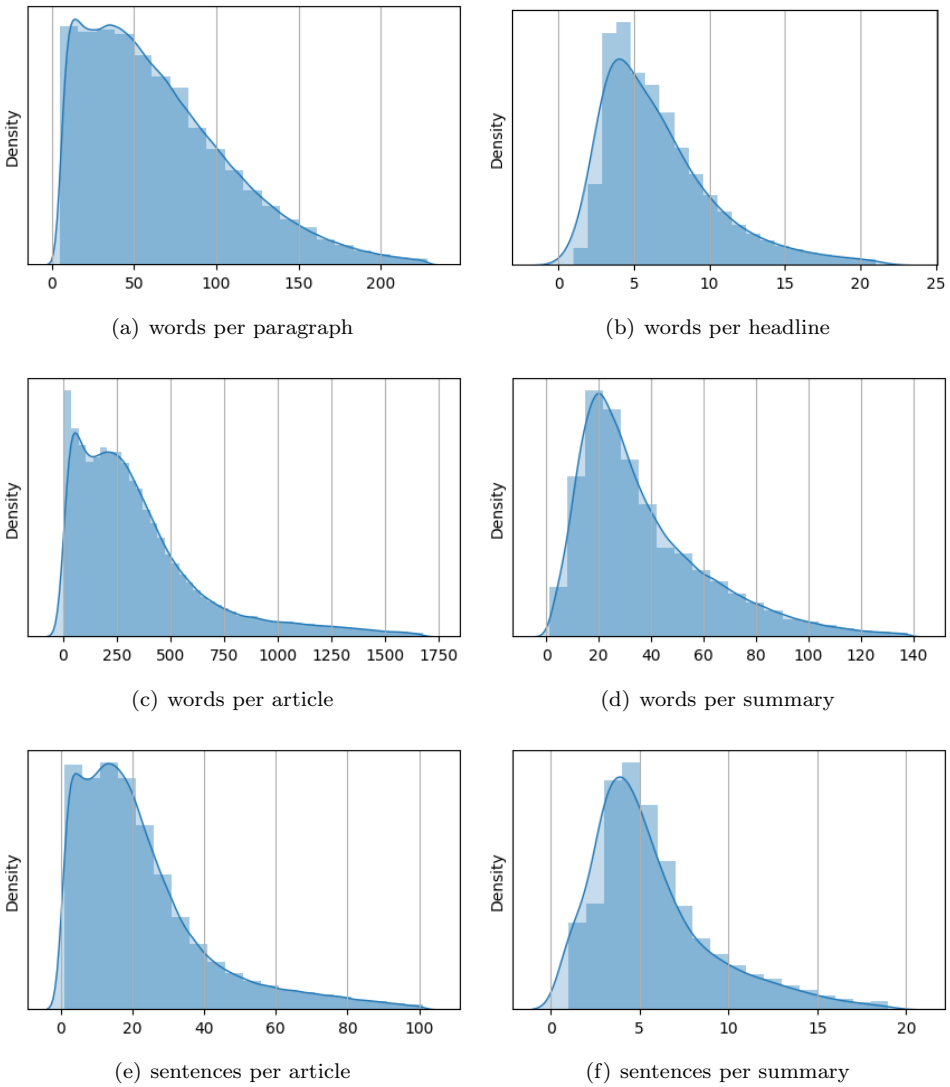
The summary statistics of the remaining paragraphs, presented in Table 1, highlight the concise nature of the reference summary. In fact, headlines are on average from 5 to 6 words long, while text paragraphs are on average about 125 words long, resulting in an extreme reduction of more than 20 times of the original text lengths. The distribution of the paragraphs statistics, visualized in figure 7, shows that lengths overall are skewed towards short paragraphs and summaries.

		Text		**Summary**	
		words	sentences	words	sentences
Paragraph	median	118	7	5	1
	mean	125.03	8.21	5.68	1.00
	std	43.88	2.49	2.84	0.00
Article	median	274	18	30	5
	mean	353.02	22.38	37.16	5.61
	std	311.85	18.64	24.96	3.45

Table 1: *summary statistics of paragraphs and articles length by words and sentences*

Moving on to analyze articles characteristics we aggregated the filtered paragraphs by article name and joined, on one hand, the paragraphs to obtain the articles' text and, on the other hand, the headlines to obtain the target summaries. Similarly to what has been done with paragraphs, we applied a statistical filter to the obtained articles by dropping all records whose text's or headlines's length in words or sentences exceeded the mean by more than 3 standard deviations; this resulted in a loss of about 4.4% of the articles.

The summary statistics of the remaining articles, presented in Table 1, showed a significant reduction in the difference between texts and headlines mean lengths, now having an average word length ratio of less than 10:1. Lengths presented again an overall skewness of the distributions towards shorter articles and summaries, both in terms of words and sentences, as it can be clearly seen in figure 7.



(a) words per paragraph

(b) words per headline

(c) words per article

(d) words per summary

(e) sentences per article

(f) sentences per summary

Figure 7: *distributions of paragraphs and articles length by words and sentences*

4.2 Extractive summarization

As anticipated, in our project we focused on extractive summarization techniques, applying to WikiHow articles two widely used algorithms, namely: TextRank (TR), and Latent Semantic Analysis (LSA). Additionally, we implemented two modified versions of said techniques, designed to summarize texts on a paragraph basis; we will refer to them as TextRank by paragraph (TRp) and LSA by paragraph (LSAp).

The idea behind these two variations of the original algorithms was to leverage the texts' structure and segmentation to extract from each paragraph the most information-relevant sentence. This approach to summarization was inspired on one hand by the the step by step or options-list-like format of most of the WikiHow articles, where usually to each paragraph corresponds a distinct action or solution to a given "how to" question. On the other hand, this was exactly the approach used by the dataset creators to define the target summaries, and we wanted to see how forcing the algorithms to work on texts paragraph by paragraph, imitating the summaries' construction process, could help or hinder their performances.

We implemented three different parametrization of the algorithms by varying the number of sentences that composed the extractive summaries. We used the text's number of paragraphs, which coincided by construction with the number of sentences of the reference summary, as a starting point for setting the target length of our extractive summaries. Subsequently, we studied the effect of reducing the automatic summaries' number of sentences by two and three folds relative to the text's number of paragraphs (we will be referring to this procedure as reduction, its level indicating the fraction of the original number of paragraph used as target lengths for the summaries). In doing so we were able to produce summaries of similar word length to the reference ones, at the expense of losing valuable information in the process of extremely reducing the number of sentences.

While for TR and LSA the process of varying the number of target sentences was straight forward given their implementation, for our paragraph by paragraph implementation, TRp and LSAp, we opted for the following two steps approach:

1. the algorithm selects one sentence from each paragraph, constructing an intermediate summary.

2. the intermediate summary is fed to the non by-paragraph version of the algorithm and reduced to the desired target length.

4.3 Evaluation

In an effort to challenge the algorithms and to avoid trivial solutions, as in the case of one sentence paragraphs, we decided to further filter the corpus by favoring longer and more complex articles. Articles were considered eligible for summarization if they had: a minimum 3 sentences per paragraph, a paragraph length of at least 10 words, a minimum of 6 paragraphs, and a text length of at least 100 words. From this pool of articles, consisting about a fourth of the original corpus, we proceeded to select a random sample of 1000 articles to be the subject of summarization and evaluation.

We evaluated the results of our models on an article basis using three different ROUGE (Recall-Oriented Understudy for Gisting Evaluation) metrics: ROUGE-1, ROUGE-2, and ROUGE-L; additionally for each article we computed an average ROUGE by averaging the results of said metrics and used it as a summary statistics for comparing models. The average ROUGE was studied for each model and reduction level in terms of standard deviation and mean with relative 95% confidence intervals. The results of our analysis are summarized in Table 2.

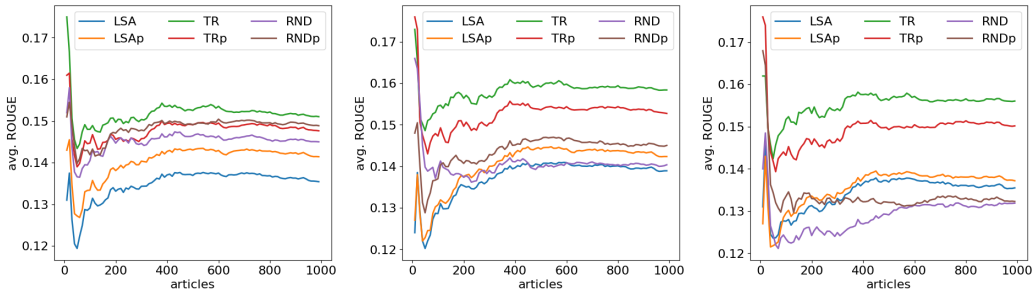
		LSA	LSAp	TR	TRp	RND	RNDp
reduction=1	mean	0.136	0.142	**0.151**	0.148	0.145	0.149
	95% CI	±0.003	±0.003	±0.003	±0.003	±0.003	±0.003
	std	0.047	0.0490	0.0525	0.051	0.048	0.051
reduction=2	mean	0.139	0.142	**0.158**	0.152	0.140	0.145
	95% CI	±0.003	±0.003	±0.003	±0.004	±0.003	±0.004
	std	0.051	0.053	0.056	0.057	0.053	0.056
reduction=3	mean	0.136	0.137	**0.156**	0.150	0.132	0.132
	95% CI	±0.003	±0.003	±0.004	±0.004	±0.004	±0.004
	std	0.054	0.054	0.058	0.059	0.057	0.059

Table 2: *average ROUGE statistics by reduction level*

As a way to contextualize the evaluation of our algorithms' results, we implemented two additional baseline summarization algorithms: Random (RND) and Random by paragraph (RNDp). RND composes summaries by selecting a given number of sentences from a text at random. RNDp composes summaries by selecting at random a sentence from each paragraph of a given text and, if needed, reduces the number of sentences of said summary to a target length by applying RND.

In our testing TR resulted in the best model in terms of mean average ROUGE for every reduction level; TRp being the third best for reduction level 1 and second best for reduction level 2 and 3. All implemented models seem to struggle in comparison to the random baseline model at reduction level 1, but progressively got better results as the reduction level was increased. To this point, LSA based models performed overall worse than the random counterpart both at reduction level 1 and 2.

Finally, to empirically ensure the consistency of our results and the impact that the dimension of the selected sample may had on the model ranking, we studied the behavior of the average ROUGE's mean as the number of articles increased. The trend, visualized in Figure 8, shows how models' ranking for any level of reduction remained mostly unchanged passed the 500 samples mark. Although not a rigorous proof, this combined with the confidence interval analysis conducted, supports our choice of sample size and the generalizability of our analysis over the models mean performance.



(a) reduction=1 (b) reduction=2 (c) reduction=3

Figure 8: *average ROUGE trend as the sample dimension increases*

5 Conclusions

In conclusion, this project has demonstrated the efficacy of text mining techniques, specifically Topic Modeling and Text Summarization, in extracting meaningful insights from a vast corpus of textual data.

The comparative analysis between Latent Dirichlet Allocation (LDA) and Latent Semantic Analysis (LSA) highlights distinct trade-offs in coherence metrics and computational efficiency. LDA exhibits superior coherence metric values, indicating better semantic coherence and topic interpretability. However, this advantage comes at the cost of significantly longer processing times, potentially limiting its scalability for large-scale text analysis tasks. Conversely, LSA demonstrates faster processing speeds but comparatively lower coherence metrics, suggesting a trade-off between computational efficiency and semantic fidelity. Thus, the selection between LDA and LSA hinges on the specific requirements of the analysis, balancing the need for coherence and interpretability against computational constraints.

By studying the application of both traditional models and novel variation of well established summarization algorithms we gained numerous insights on their strengths and weaknesses. We analyzed and quantified their performances and how factors like the target sentences' number reflects on metrics results when tasked with summing up the specific guide-like article style of WikiHow. Overall our proposed paragraph by paragraph approach to summarization proved in testing to increase the results of LSA and decrease TextRank's, the latter obtaining the best results in all the settings studied. The dataset proved to be challenging, with factors like the target summaries' extreme conciseness and synthetic construction being the main suspects for the poor results' diversification, in terms of ROUGE, between the model. Indeed, random based baseline methods often challenged and sometimes beat the studied models, especially for low reduction values.

Looking ahead, future development and improvement in this domain could include exploring more advanced machine learning models and techniques, such as neural network-based approaches, which may further enhance the performance of text mining tasks on large-scale datasets like WikiHow.

References

[1] WikiHow: A Large Scale Text Summarization Dataset
https://arxiv.org/abs/1810.09305

[2] G. Pasi and M. Viviani. Text Mining and Search: course lecture notes, 2023
https://elearning.unimib.it/

[3] Sumy, module for automatic summarization of text documents and HTML pages
https://github.com/miso-belica/sumy