Extractive Text Summarization and Topic Modeling over Reddit Posts

Giorgio Carbone (matr. 811974)[1], Marco Scatassi (matr. 883823)[2], Gianluca Scuri (matr. 883256)[3]

Abstract
Reddit is a social news aggregation and discussion website where users post content (such as links, text posts, images, and videos) on a wide range of topics in domain-specific boards called "*communities*" or "*subreddits*." The following project aims to implement **text summarization** and **topic modelling** pipelines on the textual content of Redditors' posts. The dataset used to achieve these goals is the Reddit-based TL;DR summarization dataset **TLDRHQ**[1] containing approximately 1.7 million Reddit posts (submissions as well as comments) obtained through scraping techniques. Each instance in the dataset includes a Reddit post and its **TL;DR**, which is an acronym for "Too Long; Didn't Read" and is an extremely short summary of the post's content that is good practice for users to leave at the end of a post. While the Reddit usage has increased, the practice of write TL;DR didn't keep the pace. In this context, a system (such as a bot) capable to automatically generate the TL;DR of a post could improve Reddit usability. However, the high abstractivity, heterogeneity and noisy of posts make the text summarization task challenging. In this work a supervised extreme extractive summarization model is developed. Despite its lower complexity, results show that its performance are not so different with respect to the state of the art BertSumExt. Moreover the topic modeling analysis of the posts could be really useful in identifying the hidden topics in a post and evaluate if it's published in the right subreddit. In this project LSA and LDA techniques are used. On this dataset LSA outperformed LDA and identified 20 well defined topics providing the respective document-topics and topic-terms matrices.

Keywords
Reddit – TL;DR – Extractive Text Summarization – Topic Modeling

[1,2,3] *Department of Informatics, Systems and Communication, University of Milano-Bicocca, Milano, Italia*

Contents

1. Introduction

1.1 Reddit: the front page of the internet

Reddit [2] is an online platform that offers a unique blend of social news aggregation and discussion features. It was officially launched in June 2005 in San Francisco and has since grown to become one of the most popular websites in the world. In 2020, Reddit reached an average of 52 million daily users [3], and by 2022 it was home to more than 100,000 active communities, with 430+ million posts submitted only in 2022 [4]. The platform is designed to allow users to interact with content and other users in a variety of ways. Reddit users, called *Redditors*, can rate web content through up-voting, participate in discussions through communities, and create and publish their own content on a wide range of topics such as news, politics, science, and sports.
On Reddit, posts and comments are organised into communities known as *subreddits*. Subreddits are focused on certain topics and allow users to share domain-specific information and engage in discussions. Users submit posts as ”submissions” within subreddits and others can respond by commenting on them. Both submissions and comments contain a text body, or ”selftext”, which reflects the exchange of information about a specific topic among users.
Overall, Reddit provides a dynamic and interactive experience for users who want to stay informed, share information, and connect with others who share similar interests. With its vast array of topics, user-generated content, and engaged community, Reddit offers something for everyone.

A common practice among Reddit users is to leave at the end of a post a ”TL;DR”, which stands for ”Too Long; Didn’t Read” and is used to provide a brief summary of a lengthy post for those who may not have time to read the entire text. This allows for users to quickly understand the main points of a post even if they don’t have the time to read through the entire text.
An example of a Reddit post with its TL;DR is shown in Figure 1, from which it emerges how extremely short and highly abstractive these summaries are.

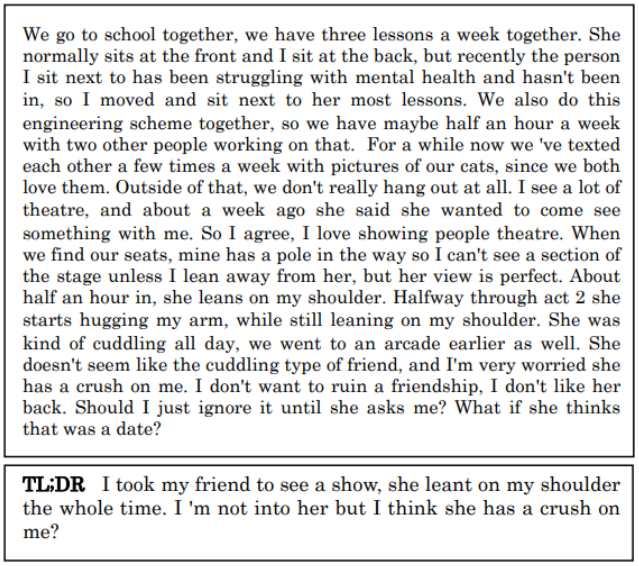


Figure 1. Example of a Reddit post and its TL;DR. From [1].

TL;DR summaries are useful for platform users, but data suggest that their use is not increasing at the same rate as the number of posts. While the number of posts submitted has grown drastically since 2005, the proportion of those that include a TL;DR summary is decreasing.[1].
In this context, a system, such as *AutoTLDR Bot*[5] based on *SMMRY*[6] algorithm, that is able to automatically generate the TL;DR of a post, can be very useful.

1.2 Research questions

The aims of this project are:

- to train a model to perform text summarization on Reddit posts (submissions and comments) obtaining a very short summary resembling a TL;DR.
- to perform topic modeling on the full documents of TLDRHQ's test set using Latent Dirichlet Allocation (LDA) and Latent Semantic Analysis (LSA) unsupervised algorithms.

2. Dataset and Data Exploration

2.1 Dataset: TLDR9+, A Large Scale Resource for Extreme Summarization of Social Media Posts

The dataset used to perform the presented tasks is known as **TLDR9+**[1], a large scale summarization dataset, released in 2021 by *The Georgetown University Information Retrieval Lab* [7], containing over 9 million posts extracted from Reddit discussions[8]. To obtain the dataset, the authors downloaded the entire data dump, from the social media data repository platform Pushshift[9], of Reddit posts covering the period 2005-2021, keeping only those containing the TL;DR. In particular, we used a fine-grained subset of TLDR9+, called **TLDRHQ**, containing High-Quality instances of TLDR9+ and obtained with the help of human annotation.
The dataset's authors performed this subsetting in order to limit noise associated with user-generated content in social media (spam, bad grammar and spelling errors), but as we will see (Data Pre-processing, Section 3), this was not sufficient. In fact, TLDRHQ required the application of appropriate phases of data cleaning and text normalization before it could be used for the described tasks.

TLDRHQ has a total of **1,671,099 instances**, each associated with a Reddit user's post (both subreddit user submissions and comments) containing TL;DRs within the posted source text and published in the period *2005-2021*.

2.2 Data Exploration

Dataset overview

Each occurrence of TLDR9+ and TLDRHQ is accompanied by the following attributes:

- **id**: The ID of the post, which is not the actual ID from the Reddit discussion forum, but rather a generated ID that follows a specific format. For example, the presence of the RS (RC) character in the ID indicates that the instance is of type Submission (Comment). **Data type**: string.
- **document**: The text of the user's post, which is split by sentences and may include `</s><s>` tokens indicating sentence boundaries. **Data type**: string.
- **summary**: A user-written summary or TL;DR of the post. The summary consists of few (typically only one or two) sentences. **Data type**: string.
- **ext_labels**: Extractive labels of the post's sentences. **Data type**: list of integers.
- **rg_labels**: The rouge scores of the post's sentences with respect to the TL;DR of the associated post. It is computed as the mean of Rouge2 and RougeL scores. **Data type**: list of floats.

Attribute	value
id	train-TLDR_RS_2012-02-4890.json
document	”i ’m looking for a new pair of headphones that i will carry around with me when i travel and walk around campus .</s><s> i do n’t want to spend more than $ 50 .</s><s> i do n’t like earbuds because they do n’t stay in very well and i ca n’t eat with them in - when i chew they fall out (it ’s the little things that bother me) .</s><s> i wear glasses so the headphones ca n’t be too tight or i ca n’t wear them for very long because the pressure hurts my ears and head .</s><s> i ’m not an audiophile but i do appreciate quality .</s><s> i prefer over-ear style .</s><s> i ’ve tried [skullcandy] (https://www.amazon.com/stores/page/E0223BFE-16C7-4883-9EBD-D9CEB63EA4C5?ingress=0&visitId=3fe3a4aa-46cc-4d87-9068-cf0f35225b6a), sony , and some other weird brand a while a back and so far the sony ’s have the least amount of pressure but also the least amount of volume .</s><s> i ca n’t turn them up because they do n’t cover the ear and i ’m not that guy who walks around and forces people to listen to distorted music from headphones .”
summary	”want new headphones - prefer over-ear . i wear glasses so ca n’t be too tight . around $ 50 . thanks !”
ext_labels	[0, 0, 0, 1, 0, 1, 0, 0]
rg_labels	[0.10165, 0.11729, 0.07898, 0.36880, 0.03765, 0.15032, 0.04066, 0.10461]

Table 1. Example of TLDRHQ instance.

TLDRHQ is supplied as divided into several **.JSON** files, each consisting of approximately 25,000 instances.
A split of the dataset into training, test and validation sets is also proposed by the authors, and we decided to adopt it in the course of the project. In the proposed split: training, test and validation sets contain 1590132, 40481 and 40486 observations respectively and have a size of 3.8 GB, 97.9 MB and 99 MB.

Missing values, duplicates and null values in `ext_label` attribute

The analyses revealed that:

- No **missing values** are present in any of the attributes of the dataset.
- There are no **duplicate** values in the `id` variable, which suggests that indeed each instance of the dataset is related to a unique post. On the other hand, `document` and `summary` variables had 38119 and 67084 duplicate values respectively, out of a total of 1.67M instances. The analysis showed that most of these duplicate posts fall into one of the following categories: automatic announcements by the subreddit; periodic announcements; messages by bots; reports by moderators and spam. Duplicate posts were removed ad shown in Section 3.
- It was also found that a total of 489 instances had only zeros in the list associated with the `ext_label` variable, indicating that none of the sentences in the string document validates the TLDR summary. These instances are of little use to the text summarization task and were therefore eliminated at the end of the text pre-processing phase.

Other statistics and analysis

The Figure 2 shows the relative frequency, in terms of the number of posts present in TLDRHQ for a certain year divided by the total number of posts present. As can be seen, most of the posts are published from 2013 on-wards, which shows the popularity of this writing style among Reddit users, although there are also earlier posts from the first period after Reddit was founded.

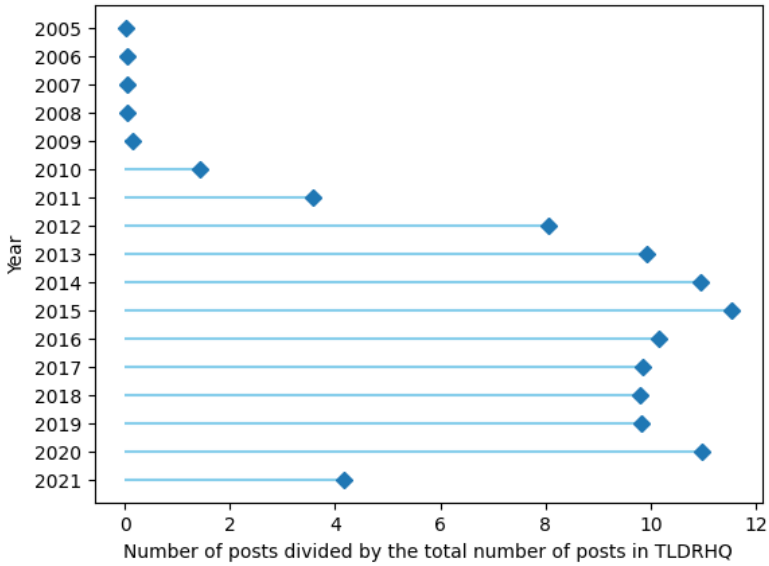


Figure 2. Lollipop chart depicting the relative frequency of posts with respect to the year of publication, in terms of the number of TLDRHQ posts published in a certain year divided by the total number of posts in the dataset.

We then analysed the proportion between the number of posts that are user submissions in the subreddits (presenting 'RS' in the id variable) and the number of comment posts related to (presenting 'RC' in the id variable). Submission-type posts are 53.8% of the total.

Continuing, the variables `document` (the text of the post) and `summary` (the corresponding TL;DR) are compared in Table 2 using some statistics. Specifically, the total word count, average word count per document, total sentence count, average sentence count per document, average word count per sentence, and overall unique word count are compared. It should be noted that these statistics were obtained on TLDRHQ dataset that underwent some of the normalization operations described in Section 3, specifically text cleaning and tokenization, but before stop-words removal and lemmatization. As expected, TL;DR summaries are very short summaries of the content of posts.

	document	**summary**
words count (tot)	468M	38M
words count (avg)	291	24
sentences count (tot)	24M	3.8M
sentences count (avg)	15	2
words count per sentence (avg)	20	11
unique words	738K	254K
compression rate	12.1	

Table 2. Statistics of the variables "document" and "summary" of TLDRHQ. The statistics are obtained on the dataset on which text cleaning and tokenization have been applied, but not stop-words removal and lemmatization.

From the value of the average word count per document and that of the average word count per summary, we obtain a compression rate value ($compression\ rate = \frac{document\ words\ count\ (avg)}{summary\ words\ count\ (avg)}$) of 12.1 that shows that authors tend to write much shorter TLDRs that highly shorten the post's text, which is expected as the purpose of TLDR summaries is to provide a brief summary of the text.

3. Data and Text Pre-processing

The methodological approach to the pre-processing step of this project can be divided into several phases described in the following paragraphs. All operations, apart from the preliminary data cleaning, were performed in multiprocessing using python's **Swifter**[10] library, so that the large amount of calculations could be distributed over all CPU processors.

1. Data Cleaning: this first pre-processing phase, which is applied to the entire dataset, mainly involved the **removal of duplicate values** for the `document` and `summary` variables. All the following steps have been applied to the variable `document` only.

2. Sentence Splitting: Sentences splitting of documents is then performed using as separator the string `</s><s>` already present in the dataset. This step is performed at the beginning to prevent the string from being deleted in subsequent steps. A list of strings is obtained from the original string.

3. Text Normalization: Text Cleaning
A series of text cleaning operations are applied to document sentences, with the aim of eliminating anything other than words or punctuation.

- **3.1 HTML tags and entities removal:** Removal, through the use of the **re**[11] library of all HTML tags (e.g. `<span>`) and entities (e.g.) contained in documents, left over from the web scraping process.
- **3.2 Extra White spaces Removal:** Removing excessive white spaces between strings in the document using the re library.
- **3.3 URLs Removal:** Removing URLs in documents using the re library.
- **3.4 Emoji Removal:** Removing emoji in documents using the re library.
- **3.5 User Age Processing:** On Reddit, it is common to use expressions of the type `AgeGender` (such as *25m*) to indicate the age and gender of the posting user. These expressions have been converted into words, using re (e.g. *25m* becomes *25 male*).
- **3.6 Numbers Processing:** Numbers have been treated differently depending on their nature:
  - **Ordinal Numbers**(such as 1st, 2nd, 3rd and so on) were converted into words using the re and **num2words**[12] library.
  - **Numbers in the middle of sentences** were converted into words using num2words and re.
  - **Remaining numbers** have been removed.
- **3.7 Control Characters Removal:** Removed formatting characters such as `\n \t \v \m` using re.
- **3.8 Extra White spaces Removal:** The operation is repeated to eliminate the spaces generated by the previous steps.

4. Text Normalization: Words and punctuation
Documents obtained from text cleaning were then processed through:

- **4.1 Case Folding:** Case-folding is applied by reducing all letters to lower case using the in-built `.lower()` function.
- **4.2 Repeated characters processing:** Correction of words with 3 or more repeated characters within them using re inside a custom function. Excess characters are removed and reduced to two.
- **4.3 Fix and Expand English contradictions:** Contractions are shortened versions of words or phrases, created by removing certain letters and replacing them with an apostrophe, contractions removal contributes to text normalization. This phase required two steps:
  - **Contractions Fixing**: The authors of the dataset applied forms of pre-processing to the texts, which led to the generation of errors in the text. The components of the contractions were written as separated by white space, often placed in an unintuitive place (e.g. 'wouldn't' is present as 'would n't'). This phase saw the removal of white space so that the contraction is then identifiable.

  - **Expanding Contractions**: English contractions were expanded using the `.fix(phrase, slang=True)` function of the **contractions**[13] library (e.g. *wouldn't* becomes *would not*). The `slang=True` parameter also causes slang expressions to be expanded, as they are widely used on social networks (e.g. *ain't* becomes *is not*).

- **4.4 Special Characters and Punctuation Removal:** Finally, all characters other than letters are removed from documents using re library.

5. Tokenization: The documents are uni-gram tokenized (i.e. phrases are split in single-words lists) using the `word_tokenize` function of the **NLTK**[14] library. We obtain for each document a list of lists (sentences) of strings (single words), saved in the `document_normalized` variable.

6. Stop-Words and 1-character tokens removal: Stop-words are common words in a text that do not bring semantic meaning to the sentence, such as articles, prepositions, conjunctions and adverbs. Words belonging to the English stop-words list of NLTK (`stopwords.words('english')`) were removed, as they were considered irrelevant to the proposed tasks. Similarly, those tokens consisting of only one character were also removed.

7. Lemmatization: Lemmatization is a word normalization process that consists of reducing a word to its basic form (lemma). Lemmatization can be useful for reducing vocabulary size. Lemmatization is applied to tokenized documents using `WordNetLemmatizer` module form NLTK.

8. POS tagging: Part-Of-Speech tagging is a process of marking each word in a text with its corresponding part of speech, such as noun, verb, adjective, etc. POS tagging is used to extract grammatical information from text and can be useful for various natural language processing tasks, including text summarization and topic modeling. For each token, the corresponding tag was obtained using the `pos_tag_sents()` function of NLTK's `tag` module. For each observation we obtain a list of tag lists with the same dimensions as document normalised saved as `pos_tags`.

The Table 3 compares some statistics for the two variables **document_tokenized** to which was applied: data cleaning, text cleaning and tokenization and **document_normalised** to which was also applied: stop-words removal, 1-characters removal and lemmatization.

	Text Cleaning and Tokenization	+Lemmatization and Stop-Words Removal
words count (tot)	468M	213M
words count (avg)	291	133
words count per sentence (avg)	20	9
unique words	738K	715K

Table 3. Word statistics for the variable **document_tokenized** to which was applied: data cleaning, text cleaning and tokenization and **document_normalized** to which was also applied: stop-words removal, 1-characters removal and lemmatization.

Furthermore, in Figure 4, it's shown an example of the effect of the pre-processing phase on a post sampled from the dataset.

Attribute	Value
id	train-TLDR_RC_2013-08-cm-46361.json
document (Raw Text)	it 's my first day of college guys so wish me luck ! </s><s> (why am i still sitting in my dorm room) lim kim 's voice is so great . </s><s> i 'm working through her previous releases but rain is just so good and so pretty i 've had it on repeat .</s><s> i finally got around to looking at miryo 's solo release after loving her rapping on black box and wow !!!!, i ca n't believe i dismissed it last year .</s><s> i think [leggo] (http:www.youtube.comwatch?v=ka8swvkkdhm) with narsha is my favorite track :) . </s><s> i 'm still really liking shinee 's [better off] (http:www.youtube.comwatch?v=iji8ztlsf_w) , and [ns yoon ji 's what do you know] (http:www.youtube.comwatch?v=k6fzvq7kkxm) has the same sort of overall sound to it . </s><s> i do n't know how to describe it , but i really like these two songs .
document_tokenized (Text Cleaning and Tokenization)	[["it", "is", "my", "first", "day", "of", "college", "guys", "so", "wish", "me", "luck"],["why", "am", "i", "still", "sitting", "in", "my", "dorm", "room", "lim", "kim", "s", "voice", "is", "so", "great"],["i", "am", "working", "through", "her", "previous", "releases", "but", "rain", "is", "just", "so", "good", "and", "so", "pretty", "i", "have", "had", "it", "on", "repeat"],["i", "finally", "got", "around", "to", "looking", "at", "miryo", "s", "solo", "release", "after", "loving", "her", "rapping", "on", "black", "box", "and", "wow", "i", "can", "not", "believe", "i", "dismissed", "it", "last", "year"],["i", "think", "leggo", "with", "narsha", "is", "my", "favorite", "track"],["i", "am", "still", "really", "liking", "shinee", "s", "better", "off", "and", "ns", "yoon", "ji", "s", "what", "do", "you", "know", "has", "the", "same", "sort", "of", "overall", "sound", "to", "it"],["i", "do", "not", "know", "how", "to", "describe", "it", "but", "i", "really", "like", "these", "two", "songs"]]
document_normalized (+Lemmatization and Stop-Words Removal)	[["first", "day", "college", "guy", "wish", "luck"],["still", "sitting", "dorm", "room", "lim", "kim", "voice", "great"],["working", "previous", "release", "rain", "good", "pretty", "repeat"],["finally", "got", "around", "looking", "miryo", "solo", "release", "loving", "rapping", "black", "box", "wow", "believe", "dismissed", "last", "year"],["think", "leggo", "narsha", "favorite", "track"],["still", "really", "liking", "shinee", "better", "n", "yoon", "ji", "know", "sort", "overall", "sound"],["know", "describe", "really", "like", "two", "song"]]
POS_tags	[["ADV", "NOUN", "NOUN", "NOUN", "ADJ", "NOUN"],["ADV", "VERB", "NOUN", "NOUN", "NOUN", "ADJ", "NOUN", "ADJ"],["VERB", "ADJ", "NOUN", "NOUN", "ADJ", "ADV", "NOUN"],["ADV", "VERB", "ADP", "VERB", "NOUN", "NOUN", "NOUN", "VERB", "VERB", "ADJ", "NOUN", "NOUN", "VERB", "VERB", "ADJ", "NOUN"],["VERB", "NOUN", "ADP", "ADJ", "NOUN"],["ADV", "ADV", "VERB", "NOUN", "ADV", "ADJ", "NOUN", "NOUN", "VERB", "ADV", "ADJ", "NOUN"]]

Table 4. Example of the application of the various pre-processing steps to the instance with *id: train-TLDR_RC_2013-08-cm-46361.json.*

4. Text Summarization

Text summarization is an NLP task aimed at identify and extract the most important information within a text. This make it possible to provide final users with a short piece of text containing, ideally, all the essential points of the source document. It can be performed in several ways, depending on the type of documents and the approach used to extract summaries. In particular, we can categorised text summarization task along three main axes:

1. Input-type:
    - Single-document
    - Multi-document
2. Purpose:
    - Generic
    - Domain-specific
    - Query-based
3. Output-type:
    - Extractive
    - Abstractive

As explained in [1], given that TLDRHQ dataset is made up of Reddit posts and comments, it is well suited for the task of **extreme extractive summarization** (i.e. the extraction of one sentence length summaries). Moreover, compared to the previous categories, the task faced is *single-document*, *generic* and *extractive*. Specifically, extractive summarization is a text summarization approach, that generate a summary selecting a subset of sentences of the original document and recombining them together. This is done following three main steps:

1. Creating an intermediate representation of the input, which captures relevant information of the document.
2. Score sentences of a document with respect to their representation.
3. Select summary sentences.

The approach used in this paper is **indicator based** and uses supervised machine learning techniques to score sentences. The steps are:

1. Create a set of 8 features to represent each sentence.
2. Define a ML model to predict whether a sentence should belong to summary (1 label) or not (0 label).
3. Use the MMR(Maximal Marginal Relevance) to select the final summary.

Step 3 is only necessary if more than one sentence is selected. Indeed, in this work, three length summaries are extracted and compared for each document. Next sections will go deeper in each of the steps performed to accomplish the described task.

4.1 Data pre-processing

In addition to the pre-processing described in section 3, two more actions are performed:

1. Removal of documents without extractive summary, that lead to discard 489 documents.
2. Removal of document with only one sentence, that lead to discard 7268 documents.

Both this kind of documents are not useful to train a machine learning model. Indeed, indifferently from the characteristics of the document sentences, the model should learn to predict:

1. 0 label for each sentence of documents without extractive summary.
2. 1 label for the single sentence that composes the one sentence documents.

4.2 Feature matrix generation

Starting from the literature [15, 16, 17] and taking into account the high number of documents in the corpus (leading to a final matrix with 24265857 rows) as well as the limited amount of processing power, 8 features are chosen to represent sentences. Those are described in table 5.

Name	Description
sentence_relative_positions	$\frac{\text{sentence position within the document}}{\text{number of sentences in the document}}$
sentence_similarity_score	$\sum_{s \in D \setminus \tilde{s}} cosine\_similarity(s, \tilde{s})$ for each sentence $\tilde{s}$ in document D
word_in_sentence_relative	$\frac{\text{number of words in sentence}}{\text{number of words in the document}}$
NOUN_tag_ratio	Ratio of *nouns* to words in sentence
VERB_tag_ratio	Ratio of *verbs* to words in sentence
ADJ_tag_ratio	Ratio of *adjectives* to words in sentence
ADV_tag_ratio	Ratio of *adverbs* to words in sentence
TF-ISF	TF-ISF is a variant of TF-IDF, applied at sentence level for text summarization. According to [17] it is defined as: $w(s_i) = \sum_{j=1}^{J_i} \left[\mathrm{F}(u_j) \times \log\left(\frac{n}{n_{u_j}}\right) \right]$ $x_1(s_i) = \frac{w(s_i)}{\max(w(s_i))}$ where $F(u)$ is the frequency of u in the document, n is the number of sentences in the document, n_u is the number of sentences of the document in which u occurs and J_i is the number of uni-grams s_i. **NOTICE** in [17] bi-grams are used instead of uni-grams.

Table 5. Features used to represent sentences.

The decision of using uni-grams instead of bi-grams, in order to compute the TF-ISF features, derives from the following observations:

1. TF-ISF can be seen as a TF-IDF computed considering a document as the corpus and its sentences as documents
2. post are mostly composed by a small number of sentences ($<= 20$) and sentences can be considered as very short documents

So computing TF-ISF using bi-grams probably would have led in very small value of such quantity.

To enable the computation of those indicator over the entire corpus, documents have been splitted in several datasets and the library *pandarallel* has been used in order to parallelize the calculus.

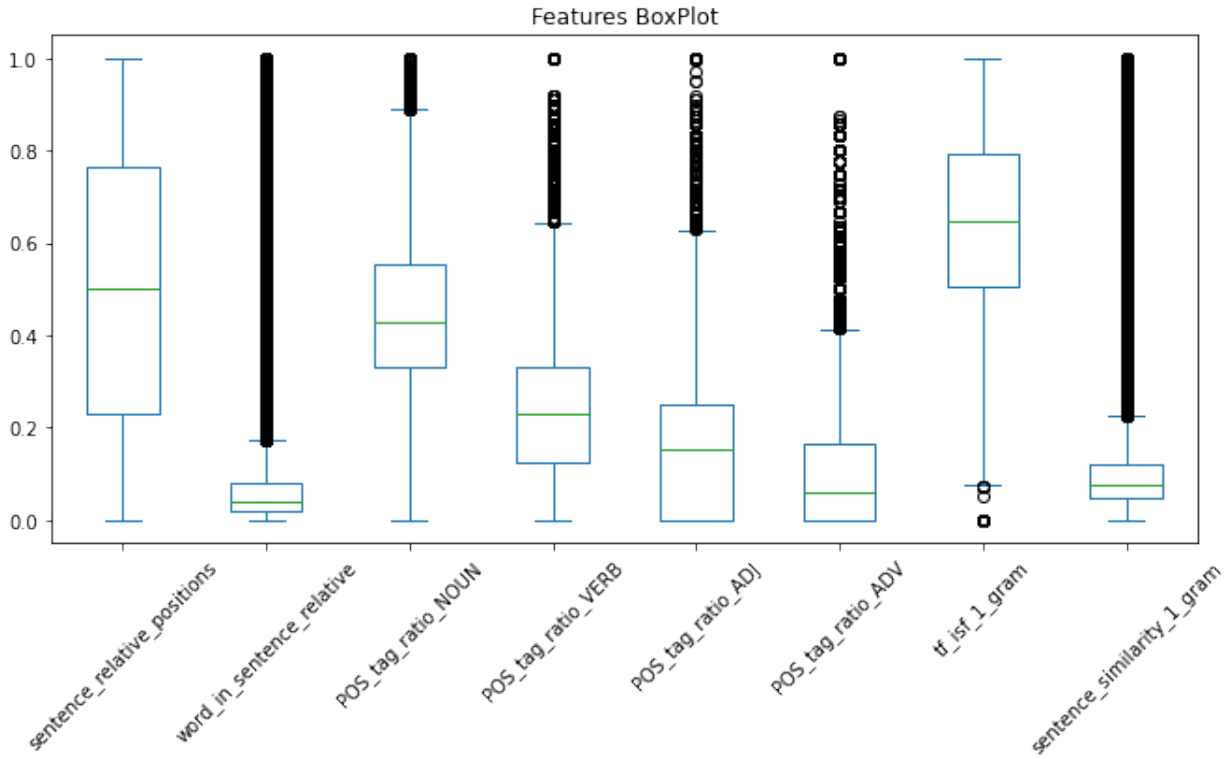


Figure 3. Features Boxplot: it can be seen how sentences are predominantly short with respect to the length of the document; that, NOUN is the main tag present within sentences followed by VERB, ADJ and ADV; that, even using uni-grams, sentence similarity tends to be quite small (there are yet some outlier for which this observation doesn't held).

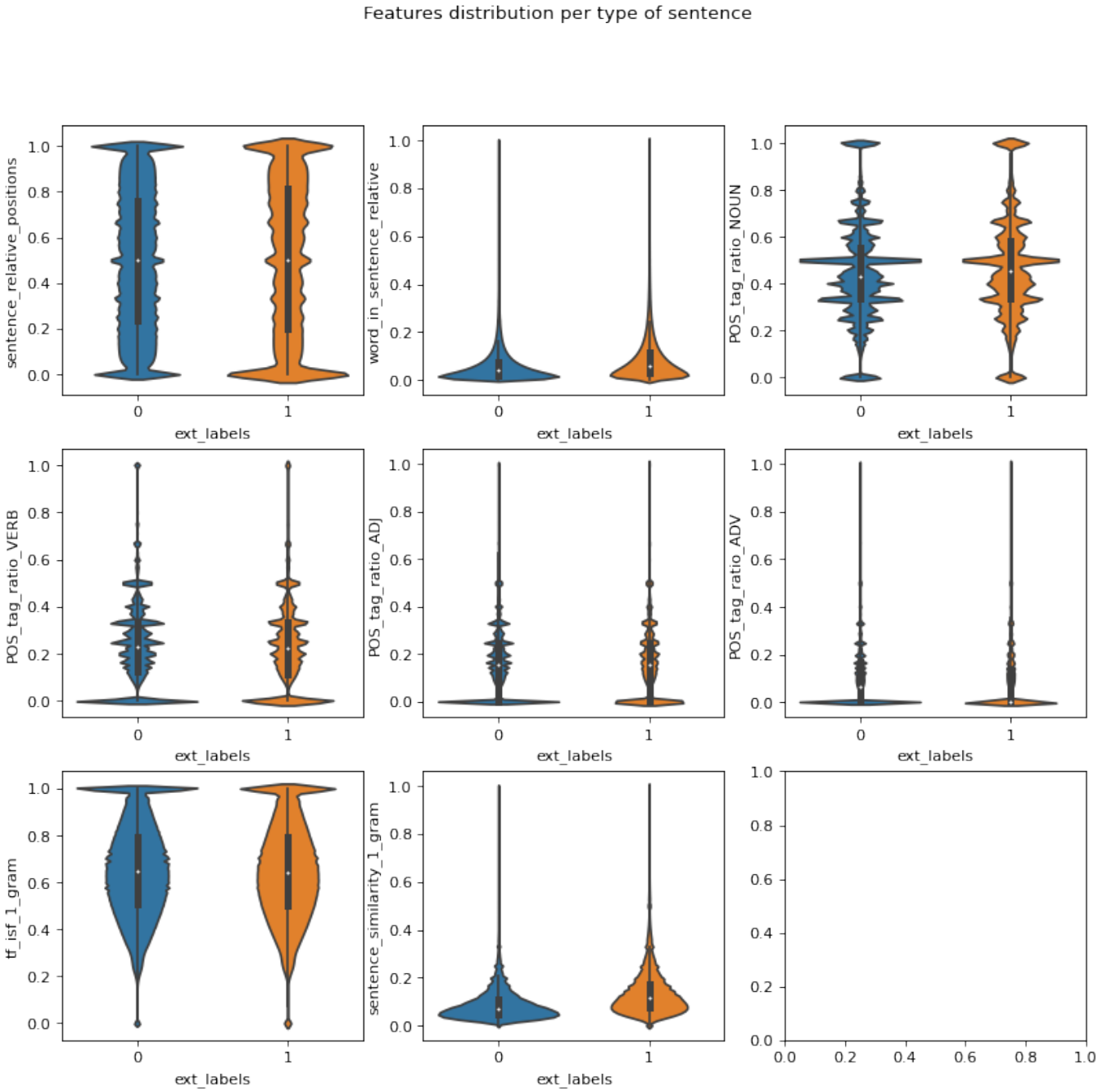


Figure 4. Features Violinplot: it is possible to observe that single features distributions are very similar between summary and non summary sentences. The main exceptions regard word in sentence and sentence similarity. Summary sentences tend to have more words and a greater similarity score than non summary sentences.

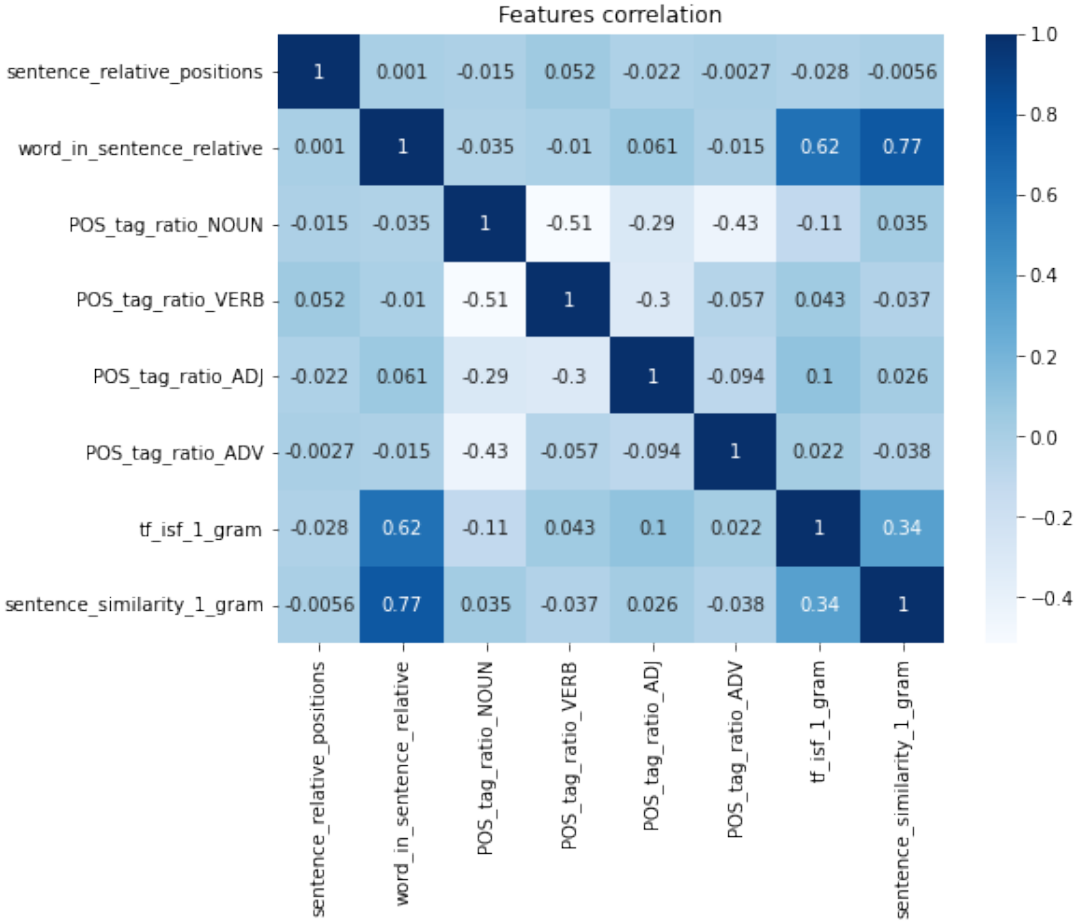


Figure 5. Features Corrplot: it can be notice that the number of words is quite strongly positive correlate with TF-ISF and sentence similarity. Other correlation are not significant.

4.3 Undersampling

From this sections on, the training set will be the union of those that in section 2 were referred as training set and validation set. This is done in order to apply a cross-validation procedure to select the best model, as will be described in the following subsection. The initial training dataset was extremely unbalanced as shown in fig 6. Specifically the ratio of summary to non summary sentences was about equal to 0.12.

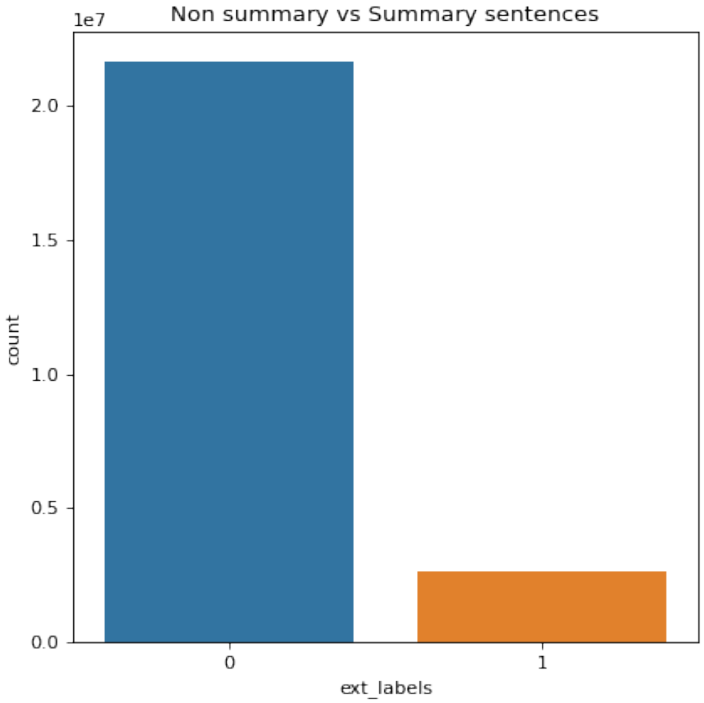


Figure 6. Countplot showing the unbalance between non summary sentences (0 label) and summary sentences (1 label).

To reduce the number of non summary sentences, two undersampling techniques are applied, in order:

1. CUR undersampling

2. ENN undersampling

4.3.1 CUR undersampling

The CUR decomposition is a matrix decomposition techniques that approximates an initial matrix A as the product of three matrices C, U and R:

$$A \approx CUR$$

where matrix C is obtained as a subset of A's columns, R as a subset of A's rows and U is determine in such a way to minimise the reconstruction error (i.e. $\|A - CUR\|$). So, this kind of decomposition try to find out the most representatives rows and columns of the initial matrix A.

In our specific case, the matrix A consisted in the subset of non summary sentences (all and only the observations with label 0), the number of columns fixed equal to 8 (we want a final matrix with the whole set of initial features) while searching for the most representative 60% of initial rows. Because of the large dimension of the training set, whose initial shape was (24265857, 8), and the necessity to perform a truncatedSVD in order to compute the matrix R, it was not possible to apply this technique to the whole dataset. Therefore, "matrix A" has been splitted in 600 submatrices of shape about (36110, 8).

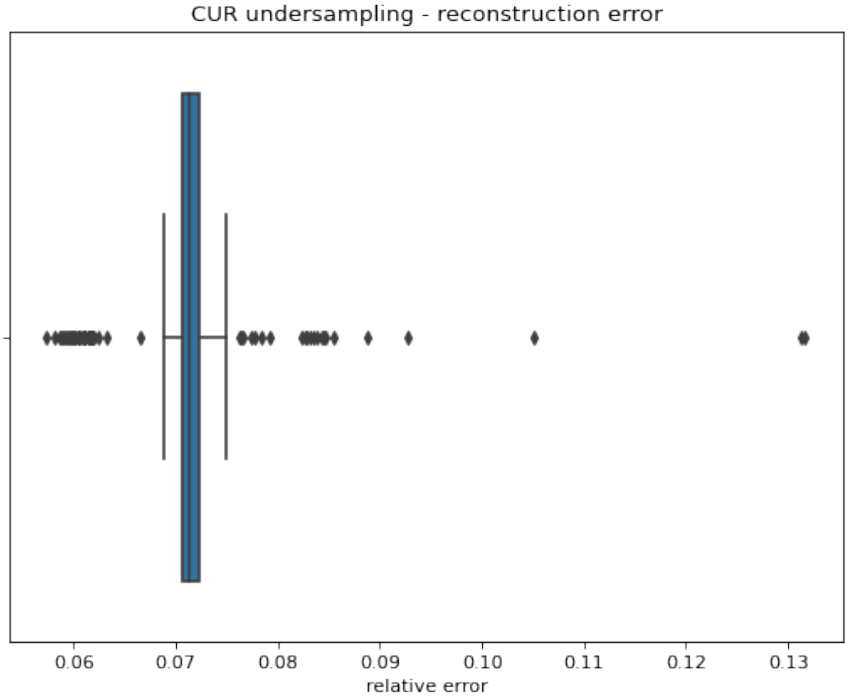


Figure 7. Boxplot of the relative reconstruction errors obtained after CUR undersampling: it is possible to see how, almost halving the rows of each subset, the reconstruction error is, in median, no greater than the 8%.

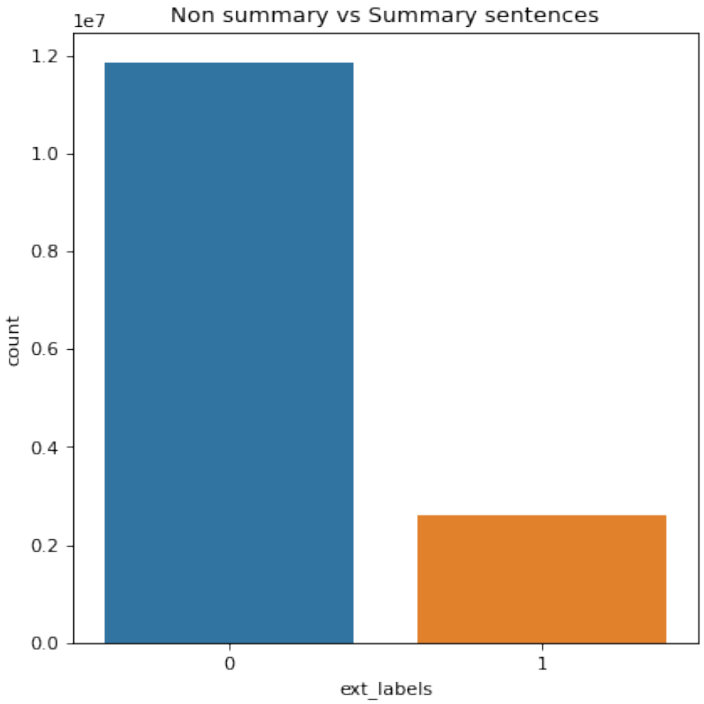


Figure 8. Countplot showing the unbalance between non summary sentences (0 label) and summary sentences (1 label) after CUR undersampling: after performing this selection the datasets has a ratio of summary to non summary sentences about equal to 0.22.

4.3.2 ENN undersampling

To further mitigate the imbalance problem as well as to reduce noisy in data, ENN undersampling is applied. As explained in the reference [18], the ENN (Edited Nearest Neighbours) is an undersampling technique that make use of a nearest-neighbour algorithm in order to remove samples that "do not agree enough with their neighbourhood". This is done "asking" a fixed number of neighbours (3 in this paper) to evaluate whether a specific observation belong or not to its group correctly. Then, only observation who receive approval from all neighbours are kept.

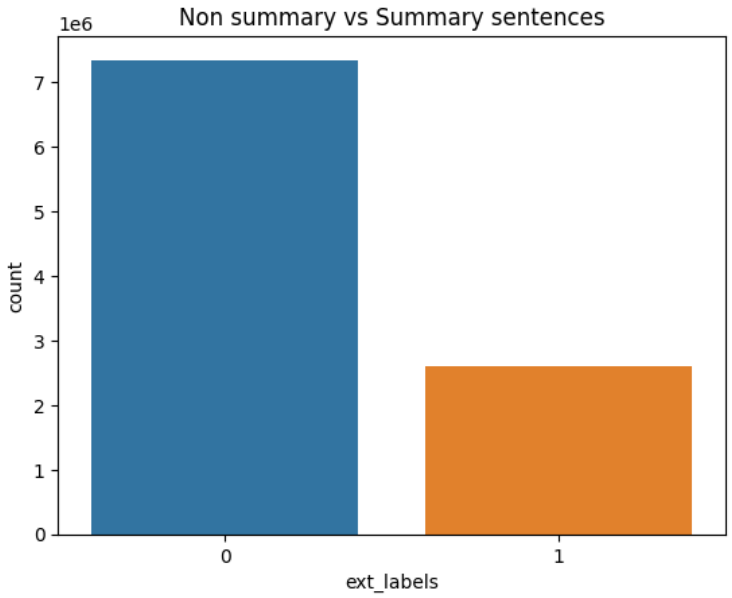


Figure 9. Countplot showing the unbalance between non summary sentences (0 label) and summary sentences (1 label) after ENN undersampling: this lead to a final with a ratio of summary over non summary sentences about equals to 0.35.

4.4 ML model selection

Different models with several set of parameters configurations has been trained to accomplish the binary classification task of identify summary and non summary sentences. In particular, *Random forest* and *Hist gradient boosting classifier* has been compared using a 3-fold cross validation, results are shown in table 7.

Index	classifier	params
1	RandomForest	{n_estimators': 50, min_samples_split': 20, max_depth': 25, 'class_weight': {0: 1, 1: 5}}
2	RandomForest	{n_estimators': 80, min_samples_split': 2, max_depth': 25, class_weight': {0: 1, 1: 3}}
3	RandomForest	{n_estimators': 50, min_samples_split': 20, max_depth': 16, class_weight': {0: 1, 1: 3}}
4	RandomForest	{n_estimators': 80, min_samples_split': 20, max_depth': 16, class_weight': {0: 1, 1: 5}}
5	RandomForest	{n_estimators': 80, min_samples_split': 20, max_depth': 25, class_weight': {0: 1, 1: 3}}
6	HistGradient	{max_iter': 500, max_bins': 255, learning_rate': 0.05, class_weight': {0: 1, 1: 3}}
7	RandomForest	{n_estimators': 80, min_samples_split': 20, max_depth': 16, class_weight': {0: 1, 1: 3}}
8	HistGradient	{max_iter': 500, max_bins': 127, learning_rate': 0.05, class_weight': {0: 1, 1: 5}}
9	RandomForest	{n_estimators': 50, min_samples_split': 20, max_depth': 25, class_weight': {0: 1, 1: 3}}
10	RandomForest	{n_estimators': 50, min_samples_split': 2, max_depth': 25, class_weight': {0: 1, 1: 5}}

Table 6. Models compared and their set of parameters.

Looking at general performance and especially at the generalization capability of the compared models, the best one resulted to be **Model 6**. In the test set its performance doesn't match so well what expected looking at table 7. This ca be seen looking at the ROC curves in figure 10.

Index	test_f1	train_f1	test_roc_auc	train_roc_auc	test_recall	train_recall	test_precision	train_precision
1	0.59	0.70	0.74	0.83	0.76	0.91	0.48	0.57
2	0.60	0.83	0.73	0.90	0.61	0.88	0.60	0.78
3	0.59	0.61	0.73	0.75	0.72	0.74	0.50	0.51
4	0.56	0.57	0.71	0.73	0.84	0.86	0.42	0.43
5	0.60	0.73	0.74	0.83	0.66	0.81	0.56	0.66
6	0.59	0.59	0.73	0.73	0.73	0.73	0.49	0.50
7	0.59	0.61	0.73	0.75	0.72	0.74	0.50	0.51
8	0.56	0.56	0.71	0.71	0.85	0.85	0.41	0.41
9	0.60	0.73	0.74	0.83	0.66	0.81	0.55	0.66
10	0.60	0.78	0.73	0.89	0.69	0.94	0.52	0.67

Table 7. Mean value of several metrics computed using a 3-fold cross validation.

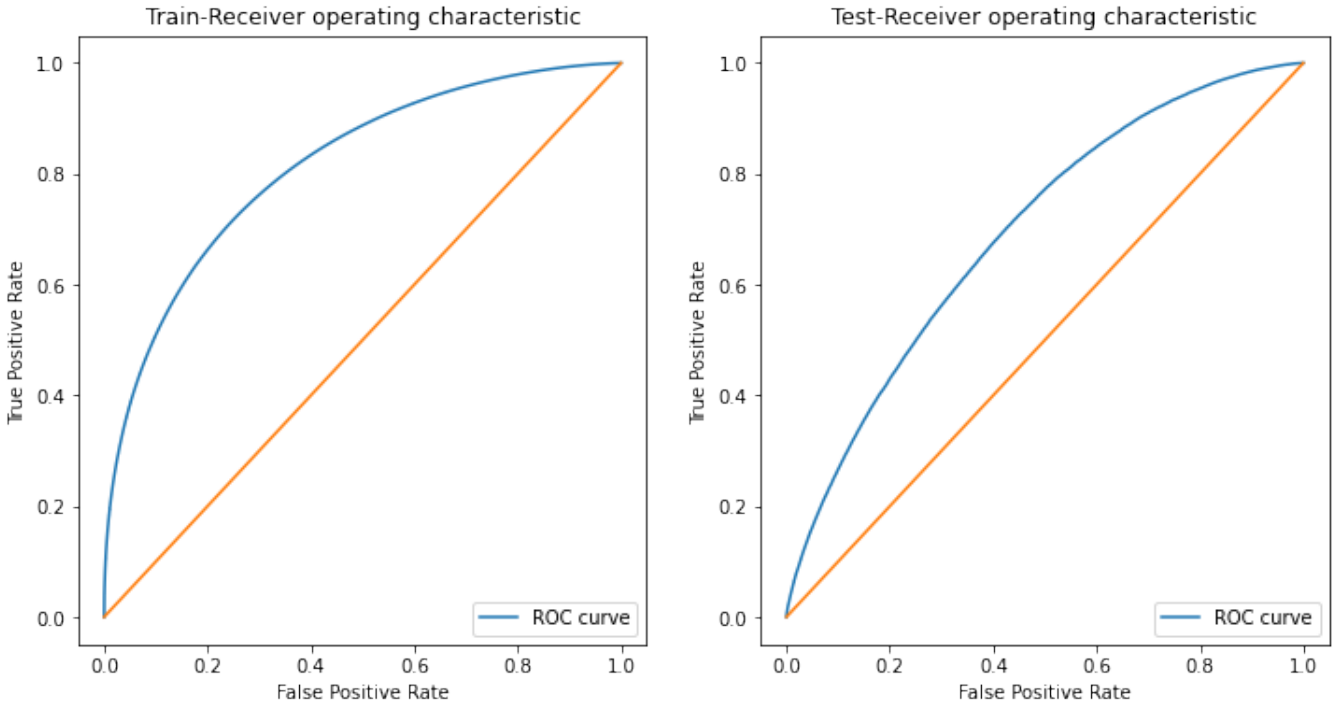


Figure 10. Roc curve on train and test datasets of Model 6

In particular, the model reached a *recall value* of 0.72, that is very similar to the mean_validation one. However, the *precision*, about equal to 0.16, is quite smaller than the mean value resulting from the cross validation.

4.5 Max Marginal Relevance (MMR)

MMR is a post selection procedure that make use of the sentences score, for example the probability output of a ML model, and similarity between sentences, in order to select the final summary.

Although the main purpose of this work is *extreme extractive summarization*, summaries with different length are generated and compared. So, while for a one sentence summary MMR is meaningless, it can be useful to generate longer summaries. We also recall that, information about similarity among sentences is already incorporated in one of the used features. However, while that feature measure similarity of a sentence with all the other in the documents, in the MMR similarity between couples of features are considered. MMR can be defined as follow:

$$MMR = \arg \max_{D_i \in R \setminus S} \left[\lambda score(D_i) - (1-\lambda) \max_{D_j \in S} \mathrm{Sim}(D_j, D_i) \right]$$

Where, D_i represent a sentence in a document D; R is the set of sentences not yet chosen; S is the current summary; $Score(D_i)$ is the score computed by the model and *Sim* is the cosine similarity. The parameter lambda has been set equal to 0.05, this,

because sentence similarity was very low in general (recall that box-plot in Figure 3 reported the statistics of the similarity sum) while the model typically predict high probability (remember the recall and precision value in the test set).

4.6 Summary evaluation

Three different summaries length has been extracted for each test document:

- one-sentence summary, only one sentence (extreme summarization)
- two-sentence summary, two sentences
- sqrt-sentence summary, $\sqrt{\text{sentences in document}}$ sentences (to generate summary with a document dependant length)

In table 8 are reported results obtained by authors of [1] using extractive and abstractive summarization techniques. In particular, for extreme extractive summarization, the state of the art techniques BertSumExt has been used.

Compared to results obtained in [1], our model reach poor performance with respect to all type of Rouge-score computed. Nevertheless, those are still comparable, with the ones in Table 8, despite the lower level of complexity. In addition, our model, or at least a conceptually similar one, could be explored and analysed better in light of the used features. Table 10 and 11 show that, increasing the number of sentences included in the summaries lead to minimal improvement in performances.

	TLDR9+			TLDRHQ		
Model	RG-1(%)	RG-2(%)	RG-L(%)	RG-1(%)	RG-2(%)	RG-L(%)
BERTSUMEXT (Liu and Lapata, 2019)	20.94	4.98	14.48	28.40	11.35	21.38
BERTSUMABS (Liu and Lapata, 2019)	23.05	9.48	18.07	28.96	12.08	22.08
BART (Lewis et al., 2020)	23.59	9.69	18.62	32.44	14.85	27.39
ORACLE-EXT	30.26	9.74	20.60	45.29	25.47	36.86

Table 8. RougE (F1) results of the state-of-the-art summarization models on the test sets of the proposed TLDR summarization datasets (TLDR9+, and TLDRHQ).

	rouge1_f1(%)	rouge2_f1(%)	rougeL_f1(%)
count	79545.00	79545.00	79545.00
mean	21.99	7.06	17.15
std	13.52	9.74	11.47
min	0.00	0.00	0.00
25%	12.00	0.00	9.23
50%	20.69	3.48	15.09
75%	30.77	11.43	23.33
max	100.00	100.00	100.00

Table 9. rouge F1 score for **one sentence summary** on the proposal test set of TLDRHQ

	rouge1_f1(%)	rouge2_f1(%)	rougeL_f1(%)
count	39842.00	39842.00	39842.00
mean	24.34	7.83	17.44
std	10.52	7.44	8.08
min	0.00	0.00	0.00
25%	16.90	2.08	11.76
50%	23.91	6.35	16.47
75%	31.33	12.05	22.22
max	95.65	85.71	95.65

Table 10. rouge F1 score for **two sentence summary** on the proposal test set of TLDRHQ

	rouge1_f1(%)	rouge2_f1(%)	rougeL_f1(%)
count	79544.0	79544.0	79544.0
mean	25.0	8.0	18.0
std	11.0	8.0	9.0
min	0.0	0.0	0.0
25%	17.0	1.0	11.0
50%	24.0	6.0	16.0
75%	32.0	12.0	23.0
max	100.0	100.0	100.0

Table 11. rouge F1 score for **sqrt sentence summary** on the proposal test set ofTLDRHQ

4.6.1 Some examples

Looking at the following examples, some observations can be made:

- In Figure 11, can be seen how the one sentence summary capture quite well all the information inside the TL:DR (indeed it is equal to the reference extractive summary). While, the two sentence summary add some text that is not essential
- In Figure 12, can be seen how the one sentence summary is not representative of the TL;DR. Instead, the two sentence summary capture the most relevant information inside the post
- In Figure 13, can be seen again a good one sentence summary. However, in this example it can be noticed that the extractive reference summary is wrong. Examples like this one, negatively affected the performance of our model
- In Figure 14, can be seen an example in which the model couldn't capture the crucial information in all summaries

Original Post	even with deniability , the problem is that something has hooked into the account to store these scripts in their cloud system tied to the account and pull it down whenever a sync is ran . why the system is able to create these files out in a standard malware target location , instead of keeping contained to user appdata folders for config files of macros and lighting profiles is beyond me . sure , you could craft a macro to craft those files if you really felt like it , but that would be something that requires a button press , not an automatic thing on sync .
TL;DR	someone has figured out how to hook some scripts into the cloud storage system tied to razer 's accounts and it will pull down automatically with no way for you to remove it .
Extractive reference summary	even with deniability , the problem is that something has hooked into the account to store these scripts in their cloud system tied to the account and pull it down whenever a sync is ran .
One sentence summary	even with deniability , the problem is that something has hooked into the account to store these scripts in their cloud system tied to the account and pull it down whenever a sync is ran .
Two sentence summary	even with deniability , the problem is that something has hooked into the account to store these scripts in their cloud system tied to the account and pull it down whenever a sync is ran . why the system is able to create these files out in a standard malware target location , instead of keeping contained to user appdata folders for config files of macros and lighting profiles is beyond me .
Sqrt sentence summary	even with deniability , the problem is that something has hooked into the account to store these scripts in their cloud system tied to the account and pull it down whenever a sync is ran .

Figure 11. Example 1

Original Post	so i live at home with my parents and we all work full time . my parents both work day shift while my schedule varies . over the course of the past year i 've noticed that i 'm honestly quiet the jerk and i 'd like to change that . i 'm closest to my parents because i do n't have much of an extended family that 's worth writing about . so sometimes my parents do things that bug me . my mom for examples tends to burp a lot when she eats and it really disgusts me . i 'm not sure how i can go about saying " hey , that 's really disgusting can you not " without saying it like that . my dad tends to eat with his mouth open a lot which again , is rather annoying at a restaurant . the same applies here , how do i say something without being a jerk . one thing they both do is smoke and i want them to quit because of obvious reasons . i know i ca n't force them to stop but it hurts me knowing that everytime they have a smoke it takes a couple minutes away from them . i love my parents and these are just a few of things that seem to bug me . i guess i just want some advice on how to be a better person to my parents .
TL;DR	been a jerk to my parents , how do i fix it .
Extractive reference summary	the same applies here , how do i say something without being a jerk .', ' i guess i just want some advice on how to be a better person to my parents .
One sentence summary	so sometimes my parents do things that bug me .
Two sentence summary	so sometimes my parents do things that bug me . the same applies here , how do i say something without being a jerk .
Sqrt sentence summary	so sometimes my parents do things that bug me . my mom for examples tends to burp a lot when she eats and it really disgusts me . the same applies here , how do i say something without being a jerk .

Figure 12. Example 2

Original Post	perk sends this after over a week of back and forth . " thank you for your response . we have now closed any additional accounts in your household and have unflagged the accounts linked xxxxxxxxx happy perking ! please feel free to contact us if you have any additional questions or concerns and we will be happy to assist you . thank you for choosing perk ! i redeemed for my account yesterday and i got this your reward has been cancelled your perk account has been found in violation of our terms of service by our fraud detection software . your account has been flagged and your rewards have been cancelled . why bother unflagging me if you are just going to say " just kidding they are all cancelled " perk is really pissing me off . i refuse to use it until they figure their crap out .
TL;DR	unflagged my account and then cancelled all rewards i tried to redeem for .
Extractive reference summary	i redeemed for my account yesterday and i got this
One sentence summary	your account has been flagged and your rewards have been cancelled .
Two sentence summary	your account has been flagged and your rewards have been cancelled . perk is really pissing me off .
Sqrt sentence summary	thank you for your response . your account has been flagged and your rewards have been cancelled . perk is really pissing me off .

Figure 13. Example 3

Original Post	[light] : casual confessions i met this german girl through my best friend , we drank at my house and then head to the club . we party , we dance , we have a good time . i then proceed to go to the couch and sit down , i was tired . my best friend and the other girl we were with go to the bar to order something . the german girl then comes and sits besides me , we talk and then i put my arm around her and she comes closer . i could n't believe it ! this is one of the most beautiful girls i 've ever met . she 's tall , blonde , blue eyes ... perfect . i start to caress her arm , and then it happened . she goes in for the kiss . i ca n't believe what is just happening ! we make out for a solid 10 - 15 . but it goes beyond that . when i had my arm around her , i felt comfortable , and when she leaned in for the kiss , i felt like the whole world stopped , i felt happy ... i forgot about all my problems ... like if she was the one . i remember when i got home i was drunk and happy , but then it sank in that i 'm never going to see this girl again . then i got really sad about it , like if all of this was just a happy coincidence that was n't really meant to happen . i woke up today , feeling kinda hangover . but all that i could think about was her . her accent , her face , those beautiful blue eyes and they way she smiled at me . all i can think off is her , and the fact that i 'm never going to see her again it 's really hurting me . i go back and remember when we were kissing . it was romantic , how she hold me . the way she would stop and look at me , then continue to kiss me . is like we were in love , an impossible love .
TL;DR	the love of my life may very well be on a plane back to germany , and i 'm never going to see her again , and that fact is destroying me .
Extractive reference summary	all i can think off is her , and the fact that i 'm never going to see her again it 's really hurting me .
One sentence summary	i go back and remember when we were kissing .
Two sentence summary	this is one of the most beautiful girls i 've ever met . i go back and remember when we were kissing ..
Sqrt sentence summary	this is one of the most beautiful girls i 've ever met . but all that i could think about was her . i go back and remember when we were kissing . it was romantic , how she hold me . the way she would stop and look at me , then continue to kiss me .

Figure 14. Example 4

5. Topic Modeling

Topic Modeling is an NLP task aimed at detecting word and phrase patterns within them and automatically clustering word groups and similar expressions that best characterise a set of documents. It groups words into different clusters, where: each word in the cluster is likely to occur “more” for the given topic, different topics have their respective clusters of words along with corresponding probabilities and different topics may share some words and a document can have more than one topic associated with it. The results of this task are 3 components:

- list of topics (clusters of words that make sense together)
- probability of words for each topic
- distribution of topics in a document

To perform this NLP task there are different techniques involving the use of statistical methods or unsupervised machine learning technique. In this project Latent Dirichlet Allocation (LDA) [19] and Latent Semantic Analysis (LSA) [20] are performed.

These are unsupervised techniques and in this project are applied to the full documents contained in the test set.

Both of these techniques look for patterns in word distribution, which is why it is important that documents contain only words with high semantic. For this reason an additional data exploration and pre-processing phase is required.

5.1 Data exploration and pre-processing

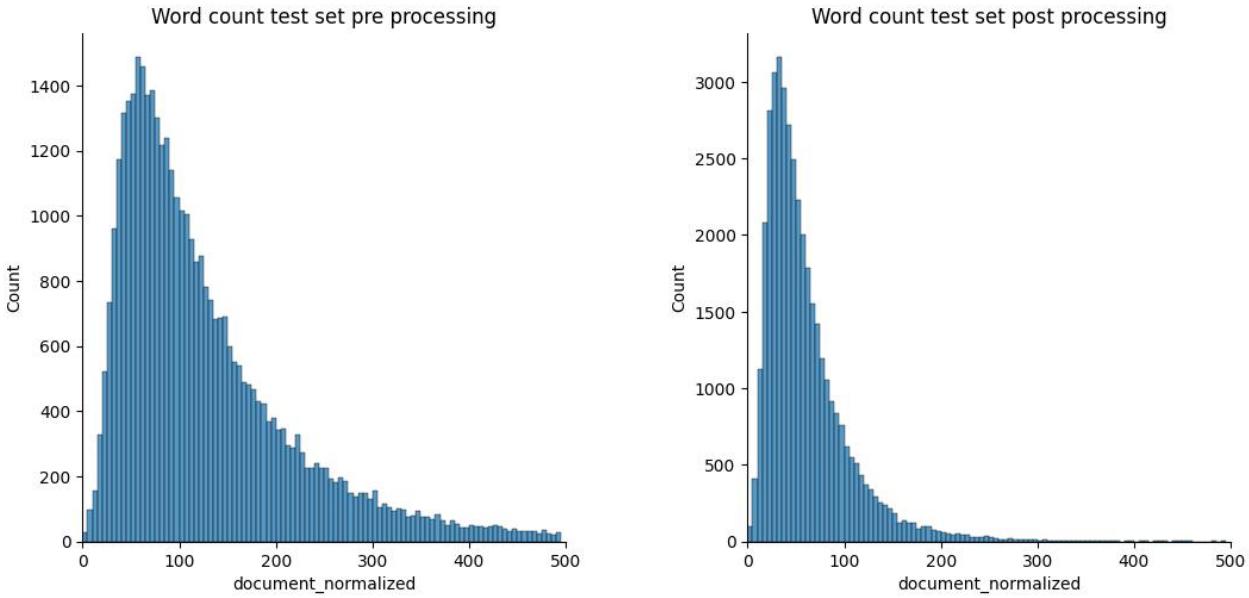
In order to keep only words with high semantic value 3 actions were performed on the dataset:

- verb and noun filtering based on POS tagging
- common and rare words removal (cut-off levels: words that appear in less than 5 documents and words that appear in more that the 50% of the documents)
- custom stop-words [’time’, ’something’, ’going’, ’year’, ’week’, ’month’, ’day’, ’get’, ’got’, ’nt’, ’want’, ’know’, ’thing’, ’could’, ’can’, ’go’, ’say’, ’said’] removal

These action drastically reduced the number of unique words from 90444 to 75879 with the first step, from 75879 to 75861 with the second one and from 75861 to 13536 with the last step. The result of these reductions can be seen in the word-cloud comparison (fig 15) and in the word-per-document comparison (fig 16). We obtained lists of tokens with high semantic value for each document.

(a) Before pre-processing **(b)** After pre-processing

Figure 15. Word-cloud comparison.



(a) Before pre-processing

(b) After pre-processing

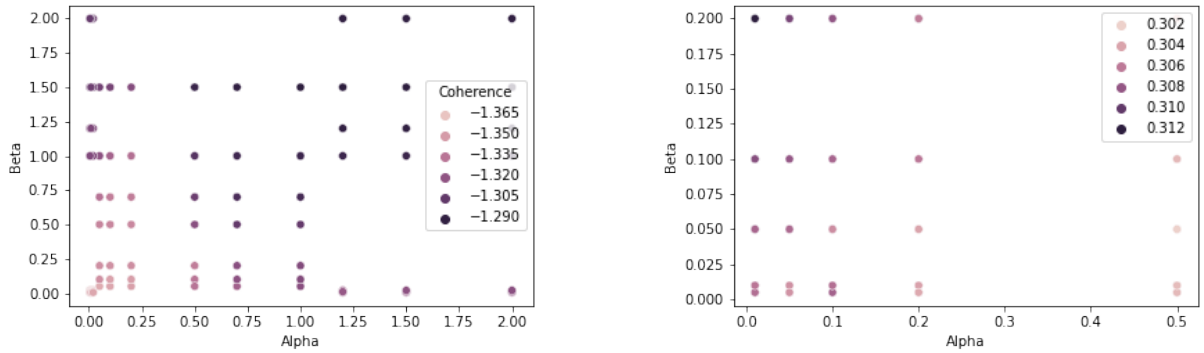
Figure 16. Words-per-document number comparison

5.2 LDA Latent Dirichlet Allocation

With this technique the topics are considered hidden (latent) and must be uncovered via analysing joint distribution to compute the conditional distribution of hidden variables (topics) given the observed variables (words in documents). The LDA makes two key assumptions:

- documents are a mixture of topics
- topics are a mixture of words

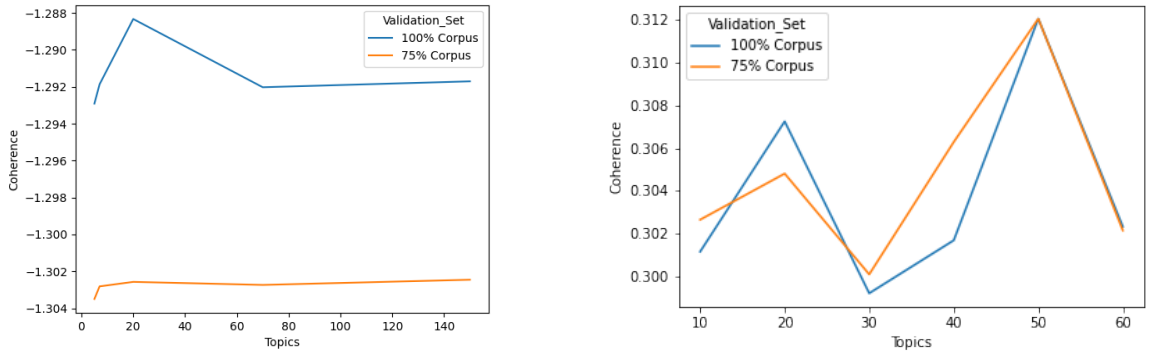
These assumptions imply that both topics in document and words in topic follow the Dirichlet distribution, this mean that documents are related to a few topics and topic are related to a few words.
To perform this technique the first step is the representation of the entire test set as Bag-of-Words creating the term-document frequency matrix. Then the LDA procedure is applied using the function implemented in the library Gensim [21]. It requires the parameters α, β and k. These are respectively the two parameters of the Dirichlet distribution doc-topics, topic-words and the number of desired topics. To tune these hyper-parameters the coherence value is used, in particular we implemented two grid searches to find the best combination which maximise the *UMass* value and the *CV* value. The range of the topic values was [5, 150] for the first one and [10, 60] for the second one.



(a) *UMass* coherence measure.

(b) *CV* coherence measure.

Figure 17. Coherence values with different α and β values and $k = 20$.



(a) *UMass* coherence measure. **(b)** c_v coherence measure.

Figure 18. Coherence values with different k values.

In the picture 17 a strange behaviour is shown by the UMass coherence measure because the best values correspond to α and β values greater than 1. The c_v coherence instead indicates $\alpha = 0.01$ and $\beta = 0.2$ as best values. Regarding the number of topics in picture 18 both the measures show a spike at $k = 20$. Performing the LDA procedure with these values the following topics are obtained.

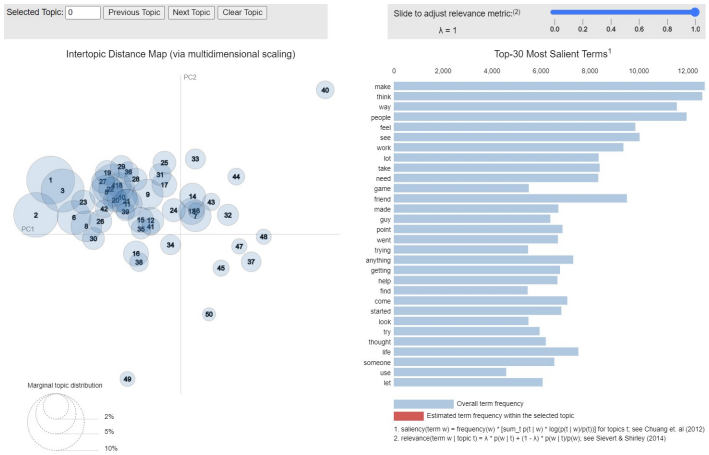


Figure 19. Topics visualization created with pyLDAvis library.

Topic	Top10Word	Score
0	people	0.006476
	make	0.006464
	think	0.005886
	way	0.005062
	work	0.004934
	need	0.004512
	take	0.004422
	see	0.004334
	friend	0.004197
	feel	0.004126
1	make	0.005896
	think	0.005468
	feel	0.004897
	lot	0.004266
	friend	0.004263
	point	0.004202
	work	0.004014
	way	0.003892
	life	0.003842
	see	0.003689

(a) Dataset of topics-terms with associated values.

	Topic_0	Topic_1	Topic_2	Topic_3	Topic_4	Topic_5	Topic_6
0	0.000801	0.000801	0.000801	0.000801	0.000801	0.000801	0.000801
1	0.468698	0.000048	0.099292	0.000048	0.000048	0.000048	0.000048
2	0.000541	0.000541	0.000541	0.000541	0.744031	0.000541	0.000541
3	0.000144	0.000144	0.000144	0.000144	0.000144	0.000144	0.000144
4	0.000426	0.000426	0.000426	0.000426	0.000426	0.000426	0.000426
...	...	...	...	...	...	...	...
40276	0.000202	0.000202	0.000202	0.000202	0.299709	0.000202	0.000202
40277	0.000426	0.000426	0.000426	0.000426	0.000426	0.000426	0.000426
40278	0.000180	0.000180	0.000180	0.000180	0.000180	0.000180	0.000180
40279	0.000184	0.000184	0.000184	0.000184	0.000184	0.000184	0.000184
40280	0.051924	0.000042	0.000042	0.000042	0.000042	0.000042	0.000042

(b) Dataset of documents-topics with associated values.

Figure 20. The two matrix obtained with LDA.

	Topic_0	Topic_1	Topic_2	Topic_3	Topic_4	Topic_5	Topic_6	Topic_7	Topic_8	Topic_9
0	people	make	think	work	think	feel	way	make	make	make
1	make	think	feel	way	lot	way	think	work	see	think
2	think	feel	people	need	way	think	make	think	think	people
3	way	lot	make	people	make	let	people	people	way	need
4	work	friend	anything	think	people	make	see	point	lot	way
5	need	point	need	take	feel	anything	friend	friend	friend	feel
6	take	work	point	point	see	lot	everything	way	made	friend
7	see	way	see	make	friend	friend	getting	life	people	getting
8	friend	life	way	getting	anything	see	lot	anything	take	lot
9	feel	see	take	help	need	people	anything	help	let	work

Figure 21. Top 10 terms associated at 10 of the 20 topics.

As it can be seen in picture 21 the topics obtained are not well distinguished and many of them have overlapped terms. Even with other hyper-parameters sets (smaller β) the result doesn't improve probably due to the little amount of words for every document.

5.3 LSA Latent Semantic Analysis

LSA computes how frequently words occur in the documents and the whole corpus and assumes that similar documents will contain approximately the same distribution of word frequencies for certain words. LSA assumes the distributional hypothesis which assume that the words that are close in meaning will occur in similar pieces of text. To perform this task two steps are required: first of all the document-term matrix was computed with TF-IDF, then using the sklearn library [22] the TruncatedSVD was applied to the matrix. TruncatedSVD perform the singular value decomposition ($M = U\Sigma V^*$) and requires the number components (topics) desired, in this project $k = 20$ is used according to LDA analysis (fig 18). Performing the LSA procedure with these values the following topics are obtained.

Topic	Top10Word	Score
0	feel	0.12
	friend	0.12
	make	0.12
	think	0.12
	people	0.11
	see	0.11
	way	0.11
	lot	0.10
	relationship	0.10
	work	0.10
1	relationship	0.19
	friend	0.15
	girl	0.14
	dating	0.13
	told	0.13
	boyfriend	0.11
	feeling	0.11
	met	0.11
	talk	0.11
	asked	0.10

(a) Dataset of topics-terms with associated values.

	Topic_0	Topic_1	Topic_2	Topic_3	Topic_4	Topic_5	Topic_6
0	0.000324	-0.000775	0.000136	0.000138	-0.001131	0.000223	0.000585
1	0.010356	0.007441	0.000878	0.000560	-0.002514	0.004490	-0.007474
2	0.003139	-0.000578	-0.002514	-0.004765	0.000873	0.004682	0.003370
3	0.006234	-0.003351	-0.009461	-0.002798	-0.008088	0.002133	0.000580
4	0.003084	-0.000341	0.001185	-0.004371	-0.000259	0.003049	-0.005540
...	...	...	...	...	...	...	...
40276	0.005428	-0.007782	-0.000135	0.007908	0.006857	0.003586	-0.002509
40277	0.003139	-0.003985	0.005604	-0.002417	0.008045	-0.006301	-0.005815
40278	0.007664	0.002567	-0.000671	-0.001296	0.007052	-0.009454	-0.001591
40279	0.003446	-0.004008	0.003059	-0.000177	-0.003394	-0.003047	-0.003462
40280	0.009985	0.005502	0.009950	0.004914	-0.001033	-0.014162	0.006614

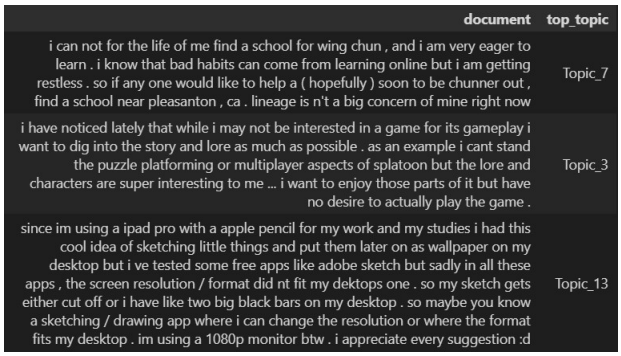
(b) Dataset of documents-topics with associated values.

Figure 22. The two matrix obtained with LSA.

	Topic_0	Topic_1	Topic_2	Topic_3	Topic_4	Topic_5	Topic_6	Topic_7	Topic_8	Topic_9
0	feel	relationship	went	game	game	game	school	school	guy	hundred
1	friend	friend	home	play	hundred	family	people	class	girl	thousand
2	make	girl	house	playing	money	parent	class	college	people	made
3	think	dating	car	player	play	life	friend	student	hundred	broke
4	people	told	room	played	work	play	college	experience	money	felt
5	see	boyfriend	came	friend	buy	playing	game	started	car	started
6	way	feeling	minute	guy	pay	mom	student	looking	make	ended
7	lot	met	took	team	hour	kid	everyone	girl	pay	went
8	relationship	talk	door	girl	playing	love	girl	grade	buy	met
9	work	asked	hour	fun	friend	dad	kid	help	date	twenty

Figure 23. Top 10 terms associated at 10 of the 20 topics.

These results are much more meaningful than the LDA analysis. As in can be seen from picture 23 the topics are now well distinguished. For example can be seen in picture 24 that {Topic 0: friendship, Topic 1: relationship, Topic 2: house, Topic 3: games, Topic 5: family, Topic 6: school}.



document	top_topic
i can not for the life of me find a school for wing chun , and i am very eager to learn . i know that bad habits can come from learning online but i am getting restless . so if any one would like to help a (hopefully) soon to be chunner out , find a school near pleasanton , ca . lineage is n't a big concern of mine right now	Topic_7
i have noticed lately that while i may not be interested in a game for its gameplay i want to dig into the story and lore as much as possible . as an example i cant stand the puzzle platforming or multiplayer aspects of splatoon but the lore and characters are super interesting to me ... i want to enjoy those parts of it but have no desire to actually play the game .	Topic_3
since im using a ipad pro with a apple pencil for my work and my studies i had this cool idea of sketching little things and put them later on as wallpaper on my desktop but i ve tested some free apps like adobe sketch but sadly in all these apps , the screen resolution / format did nt fit my dektops one . so my sketch gets either cut off or i have like two big black bars on my desktop . so maybe you know a sketching / drawing app where i can change the resolution or where the format fits my desktop . im using a 1080p monitor btw . i appreciate every suggestion :d	Topic_13

Figure 24. Example of the main topic associated to 3 documents {Topic 7: school, Topic 3: games, Topic 13: advices}.

6. Conclusions

An extractive summarization approach based on supervised learning has been developed. Despite it's simplicity (with the main efforts due to the poor hardware and the high volume of data), it shown not so different performance with the state of the art BertSumExt. Future developments could be the usage of a greater number and more complex features, in order to improve the semantic representation of sentences as well as their interrelations. Furthermore, stated the difficulty to develop a model capable to fit well the training set without overfitting, a finest undersample and tuning of model parameters could be explored.

Regarding topic modeling, the analysis allowed to obtain a meaningful list of 20 topics associated to every document contained in the test set of the dataset. The model that performed this task best, despite expectations, was LSA, which has extracted almost all defined and distinct topics. LDA performed poorly with different sets of hyper-parameters, maybe because of the small number of words considered in some posts. To achieve further improvements the topic modeling could be applied to the entire dataset to find better topics. Moreover tests can be done with different coherence measures other than UMass and CV or perplexity measure and with different pre-processing operations (e.g. expanding the custom stop-words list).

Bibliography

[1] Sajad Sotudeh et al. "TLDR9+: A Large Scale Resource for Extreme Summarization of Social Media Posts". In: (Nov. 2021), pp. 142–151. DOI: 10.18653/v1/2021.newsum-1.15. URL: https://aclanthology.org/2021.newsum-1.15.

[2] *Reddit - Dive into anything*. URL: https://www.reddit.com.

[3] Sahil Patel. *Reddit Claims 52 Million Daily Users, Revealing a Key Figure for Social-Media Platforms*. Ed. by The Wall Street Journal. URL: https://www.wsj.com/articles/reddit-claims-52-million-daily-users-revealing-a-key-figure-for-social-media-platforms-11606822200. (posted: Dec. 1, 2020).

[4] Reddit Staff Announcements. *Revealing This Year's (2022) Reddit Recap*. Ed. by upvoted: The Official Reddit blog. URL: https://www.redditinc.com/blog/reddit-recap-2022-global. (posted: Dec. 8, 2022).

[5] *AutoTLDR Bot*. URL: https://www.reddit.com/r/autotldr/comments/31b9fm/faq_autotldr_bot/.

[6] *SMMRY: Summarize articles, text, websites, essays and documents*. URL: https://smmry.com/about.

[7] ir@Georgetown - Home, ed. *The Georgetown University Information Retrieval Lab*. URL: https://ir.cs.georgetown.edu/. (accessed: January 16, 2023).

[8] Sajastu. *sajastu/reddit collector: Reddit Collector and Text Processor*. URL: https://github.com/sajastu/reddit_collector.

[9] Jason Baumgartner et al. *The Pushshift Reddit Dataset*. 2020. DOI: 10.48550/ARXIV.2001.08435. URL: https://arxiv.org/abs/2001.08435.

[10] jmcarpenter2. *swifter: A package which efficiently applies any function to a pandas dataframe or series in the fastest available manner*. URL: https://github.com/jmcarpenter2/swifter. (accessed: January 16, 2023).

[11] Python documentation, ed. *re - Regular expression operations*. URL: https://docs.python.org/3/library/re.html. (accessed: January 16, 2023).

[12] pypi documentation, ed. *num2words library - Convert numbers to words in multiple languages*. URL: https://pypi.org/project/num2words/. (accessed: January 16, 2023).

[13] kootenpv. *contractions: Fixes contractions such as 'you're' to 'you are'*. URL: https://github.com/kootenpv/contractions. (accessed: January 16, 2023).

[14] Steven Bird, Ewan Klein, and Edward Loper. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

[15] Vinícius Camargo da Silva, João Paulo Papa, and Kelton Augusto Pontara da Costa. *Extractive Text Summarization Using Generalized Additive Models with Interactions for Sentence Selection*. 2022. arXiv: 2212.10707 [cs.CL].

[16] Kam-Fai Wong, Mingli Wu, and Wenjie Li. "Extractive Summarization Using Supervised and Semi-Supervised Learning". In: *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*. Manchester, UK: Coling 2008 Organizing Committee, Aug. 2008, pp. 985–992. URL: https://aclanthology.org/C08-1124.

[17] Alexander Dlikman and Mark Last. "Using Machine Learning Methods and Linguistic Features in Single-Document Extractive Summarization". In: *DMNLP@PKDD/ECML*. 2016.

[18] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. "Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning". In: *Journal of Machine Learning Research* 18.17 (2017), pp. 1–5. URL: http://jmlr.org/papers/v18/16-365.

[19] David M Blei, Andrew Y Ng, and Michael I Jordan. "Latent dirichlet allocation". In: *Journal of machine Learning research* 3.Jan (2003), pp. 993–1022.

[20] Scott C Deerwester et al. *Computer information retrieval using latent semantic structure*. US Patent 4,839,853. June 1989.

[21] Radim Rehurek and Petr Sojka. "Gensim–python framework for vector space modelling". In: *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic* 3.2 (2011).

[22] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.