UNIVERSITÁ DEGLI STUDI DI MILANO-BICOCCA

Scuola di Informatica

Corso di Laurea in DATA SCIENCE

X-Sum - Text Summarization

Giannelli Alessio, Imbonati Lorenzo, Valoti Davide

Anno Accademico 2022/23

Contents

1 Introduction

Text mining consists of a series of techniques that, through natural language processing, enable the transformation of free, unstructured text from generic documents into structured data. Some of the reasons for using text mining techniques is to extract meaning from text that is difficult for a machine to interpret, to classify topics or documents, or to recognize the topics covered within them.

Thus, the goal of the project is to apply text mining techniques specifically to summarize the content of articles in a few lines and subsequently to extract from these same articles information about the most representative topics or subjects of the different documents. In fact, it is worth noting that supervised or unsupervised learning techniques such as classification and clustering could also have been applied, but based on the data chosen, the first two techniques mentioned were deemed more useful and interesting to the analyses. It should be added that in order to apply Topic Modeling, it was necessary to define vector representations for the documents, including Bag of Words.

2 Data Collection / EDA

The choice of dataset is certainly an important step, and after careful evaluation it was decided to opt for the X-SUM database, i.e., a collection of online articles from the British BBC. Retrieval of the data needed for analysis was done by directly downloading the different articles, effectively creating the database. The import generated more than 200.000 rows of data divided into three main columns: 'document' represents the article, 'summary' the abstract proposed by the BBC while 'id' is the identification code of each article. However, it was decided to keep only the sports articles in the analysis in order to have a more even distribution of the data. To achieve this it was necessary to add the category information for each article, so as to have a more complete picture regarding the subdivision and completeness of the articles. To do this, we simply linked each article in the dataframe to the corresponding URL via ID code and extracted the associated category from it. Once these operations were performed, a final data frame containing about 50.000 purely sports articles was obtained, with about 60 sports covered such as soccer, rugby, tennis, formula one and many others. There are no missing values and it can be seen that the most represented category is 'soccer' with about 25.000 observations.

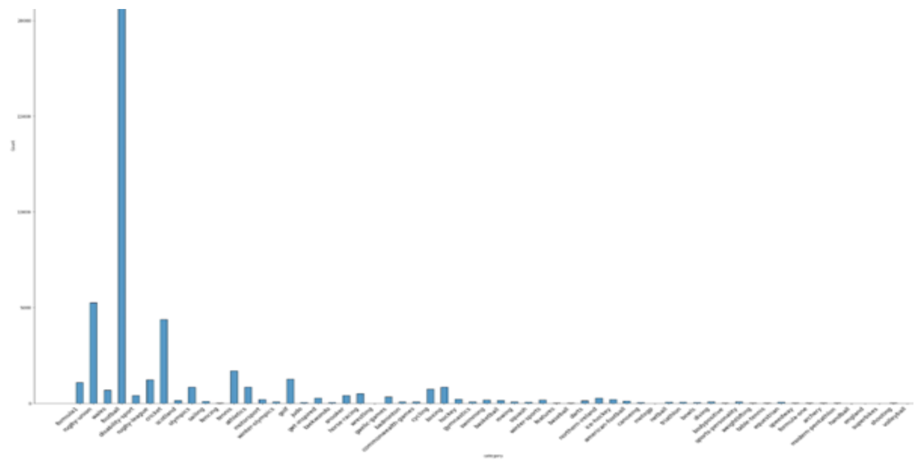


Figure 1: Distribution of sport document classes

3 Preprocessing

The pre-processing phase, i.e. the process of transforming raw data, is certainly one of the most important with regard to text processing, since without it, incorrect and inconsistent analyses could be conducted. There are several operations to be used depending on the task, the results to be obtained, the context and the analysis of interest. The cleaning steps that were performed for the summarisation and topic modeling tasks are analyzed.

3.1 Text homogeneity

One of the first operations to be carried out concerns the uniformity of the text; in fact, in order to carry out analyses that are correct, it is necessary to make the observations homogeneous with each other. The steps implemented are removal of excess spaces, conversion of all characters from uppercase to lowercase, removal of punctuation and symbols, removal of numbers, removal of repeated characters and spelling correction, removal of links and emoji, and finally removal of HTML tags.

3.1.1 White space removal

Step required to remove double spaces present between two words or to remove spaces present at the beginning or end of a sentence. The operation was performed using a simple Regex by replacing each character defined as "\n\r"" with a null character.

3.1.2 Lower case conversion

Essential operation to prevent the same word, but written in capital letters, from being considered as another term. It was performed simply by applying the "lower()" function to each word in the dataset.

3.1.3 Punctuation and symbol removal

Important operation to have observations composed only of textual data, since punctuation does not provide any additional information. We imported a list containing the various punctuation marks from the "string" library and defined a simple function that uses this list to search for and delete the various punctuation marks within the text. We emphasize that you can customize this list by simply adding the symbols you wish to delete.

3.1.4 Numbers removal

This is a nonessential but useful operation since the numbers, once the texts are converted to tokens, do not bring any additional information to the analysis. They were therefore eliminated by applying a simple Regex that replaces the characters defined as "\d+", i.e., numbers, with a null character.

3.1.5 Spelling correction

It can often happen, especially on social media, to make spelling mistakes, and this problem must be solved to avoid analyzing the same term as distinct words because of such an error. In the case under analysis, even though these are news articles written by the BBC, it was decided to perform this type of check by initially eliminating those characters repeated more than twice within the same word. Then, through the use of the “TextBlob” library, possible incorrect spelling of words was corrected. This is a computationally time-consuming but important process.

3.1.6 Emoji, links and HTML tags removal

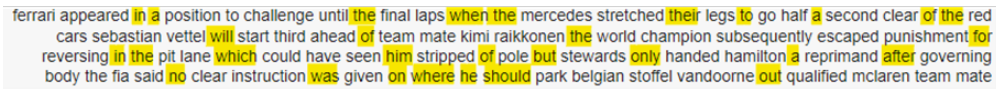
One of the latest operations is the removal of those elements that are useful to the reader of an article, such as precisely links to other articles, but without value for analysis. Another element are emoji, increasingly present within posts or articles, which only risk unnecessarily burdening the analysis. Removal of links is done using a simple Regex that replaces anything that starts with “http \ S+” or ends with “www[A-Za-z]*ċom” with a space. For the removal of HTML tags, the ”html.parser” object of the “BeautifulSoup” library is used. Finally, as for emoji, they must first be searched within the observations using the “findall()” function of the “demoji” library. Once found they can be removed but in the case under analysis no emoji was found within the articles.

3.2 Normalization

Once the first phases of data cleaning have been completed, it is necessary to proceed with the normalization phase in order to make the observations suitable for subsequent operations and analyses. This process is divided into several steps, not all essential but dependent on the task to be performed. It was decided to proceed by applying the removal of stop words, the creation of bigrams and trigrams, finally the application of tokenization and lemmatization techniques.

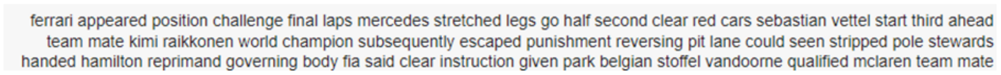
3.2.1 Stop words removal

The removal of stop words, i.e., all those words that are very frequent within the texts but do not provide any information about the content, is a very important step because it allows the size of the dataset to be greatly reduced without any information loss. Stop words are commonly all those words identified as articles, prepositions, pronouns, conjunctions, and so on, but they can also be words that, depending on the context, are constantly repeated but do not contribute information. Therefore, a function was defined that replaces all those terms in the list of English-language stopwords in the “nltk” library with a space. You can see in the images below the amount of terms removed by this operation.



ferrari appeared in a position to challenge until the final laps when the mercedes stretched their legs to go half a second clear of the red cars sebastian vettel will start third ahead of team mate kimi raikkonen the world champion subsequently escaped punishment for reversing in the pit lane which could have seen him stripped of pole but stewards only handed hamilton a reprimand after governing body the fia said no clear instruction was given on where he should park belgian stoffel vandoorne out qualified mclaren team mate

Figure 2: Sample document before stop words removal

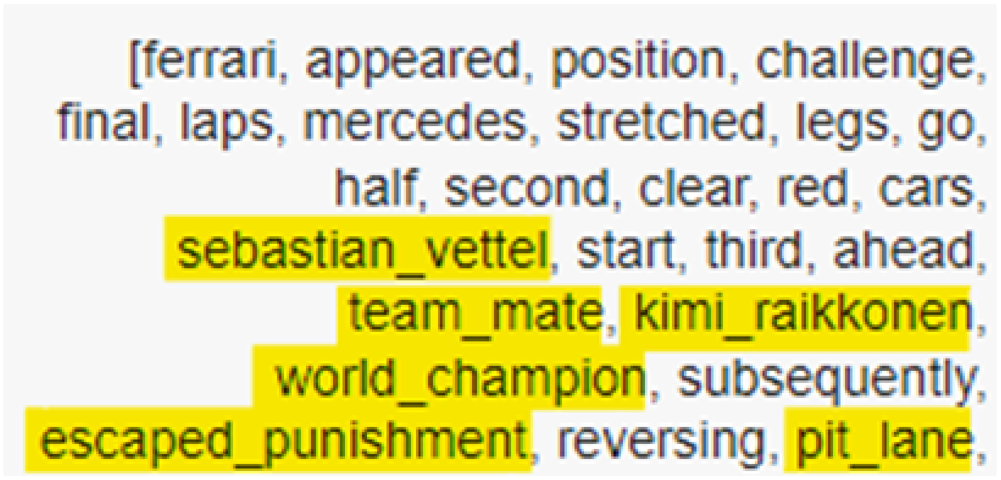


ferrari appeared position challenge final laps mercedes stretched legs go half second clear red cars sebastian vettel start third ahead team mate kimi raikkonen world champion subsequently escaped punishment reversing pit lane could seen stripped pole stewards handed hamilton reprimand governing body fia said clear instruction given park belgian stoffel vandoorne qualified mclaren team mate

Figure 3: Sample document after stop words removal

3.2.2 Bigrams and trigrams

This is an operation that allows two or more terms to be considered as one and is applied to only those occurrences within a text that occur very frequently consecutive to each other. To define these n-grams, the "gensim.models" library was used, which allows them to be constructed fairly automatically by setting a threshold and a the number of neighboring words to be analyzed.



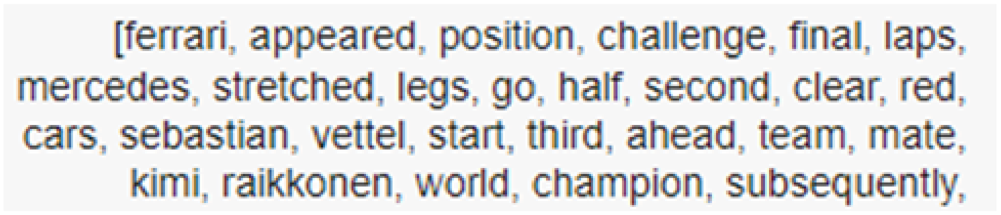
```
[ferrari, appeared, position, challenge,
final, laps, mercedes, stretched, legs, go,
half, second, clear, red, cars,
sebastian_vettel, start, third, ahead,
team_mate, kimi_raikkonen,
world_champion, subsequently,
escaped_punishment, reversing, pit_lane,
```

Figure 4: Some of the bigrams generated: interesting the pairing of the driver's name-surname or some terms used a lot together such as pit-lane and team-mate.

3.2.3 Tokenization

This is a technique for subdividing text into tokens, i.e., individual words or groups of words. It is very useful because it allows us to obtain observations from which words can be analyzed individually as a single entity, and there are several types of tokenization. It was decided to use a simple Regex tokenizer that allows text strings to be subdivided by spaces.

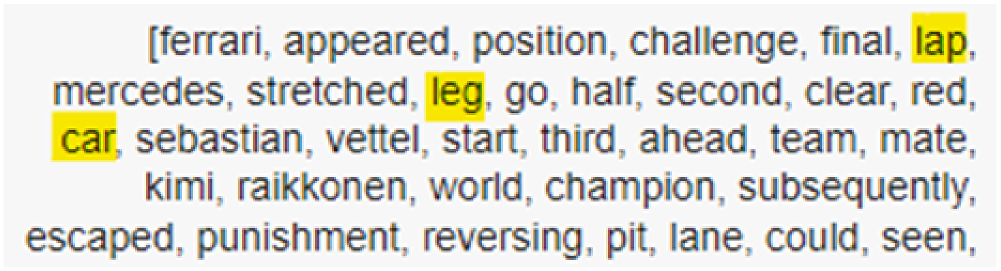
You essentially get a list of elements and no longer a string of characters.



```
[ferrari, appeared, position, challenge, final, laps,
mercedes, stretched, legs, go, half, second, clear, red,
cars, sebastian, vettel, start, third, ahead, team, mate,
kimi, raikkonen, world, champion, subsequently,
```

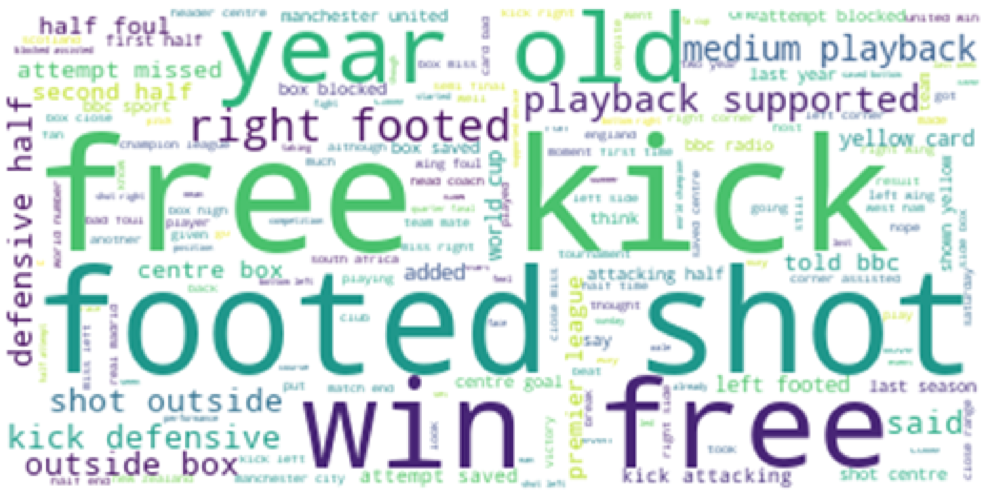
3.2.4 Lemmatization

Lemmatization is a technique of returning words to their inflected or root form. Practically words that are declined or verbs conjugated in different forms are restored to a single form and can be considered identical words. In this way, the dimensionality of the dataset is reduced, although in some particular situations there may be a loss of information; in addition, there may be situations for which a word can be traced back to more than one root. The WordNetLemmatizer function of the library of the same name was used for the lemmatization operation.


[ferrari, appeared, position, challenge, final, lap, mercedes, stretched, leg, go, half, second, clear, red, car, sebastian, vettel, start, third, ahead, team, mate, kimi, raikkonen, world, champion, subsequently, escaped, punishment, reversing, pit, lane, could, seen,

3.3 Wordcloud

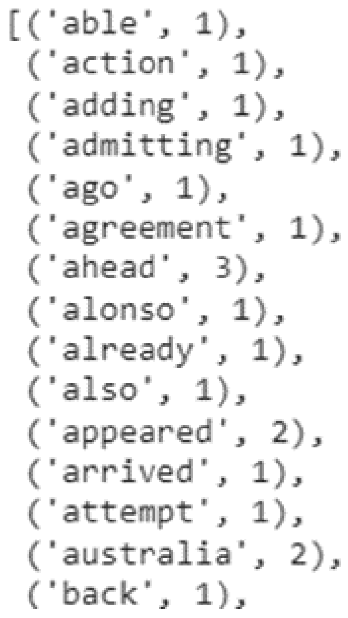
Wordcloud is a technique to visualize the most used words within the articles under analysis. As can be seen, some of the most used terms are 'free_kick', 'footed_shot' which bring to mind soccer which precisely accounts for almost half of all the sports articles analyzed. Another prominent term is 'win' which instead can be applied to most sports.



4 Document representation

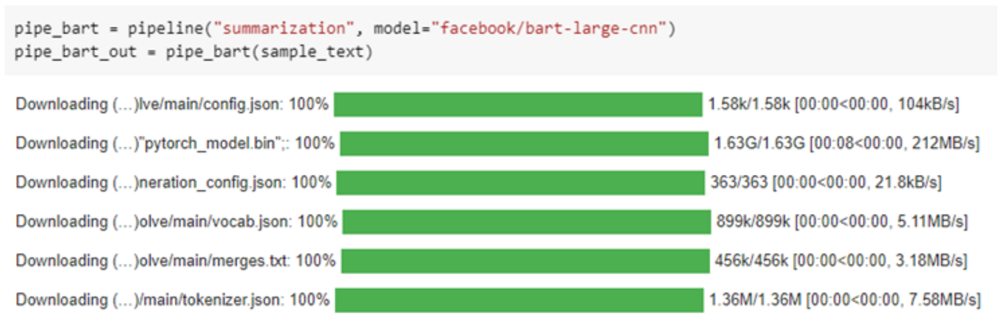
Document representation aims to encode the semantic information of the whole document into a real-valued representation vector, which could be further utilized in downstream tasks. Recently, document representation has become an essential task in natural language processing and has been widely used in many document-level real-world applications.

For the execution of the Topic Modeling task, it was decided to use the Bag of Words method, which allows a text to be represented as the bag (multiset) of its words, neglecting grammar and even word order but maintaining multiplicity. This is a widely used method when the occurrence of words is needed. This made it possible to generate a dictionary used later to perform LDA.



```
[('able', 1),
 ('action', 1),
 ('adding', 1),
 ('admitting', 1),
 ('ago', 1),
 ('agreement', 1),
 ('ahead', 3),
 ('alonso', 1),
 ('already', 1),
 ('also', 1),
 ('appeared', 2),
 ('arrived', 1),
 ('attempt', 1),
 ('australia', 2),
 ('back', 1),
```

As for the Summarization task, however, it was decided to use pre-trained models compared with each other, using the pre-processing and document representation techniques already implemented within them. In fact, functions such as 'pipeline' and 'AutoTokenizer' were imported from the 'transformers' library and applied to the different models so as to follow the same procedure defined by the implementers.



```
pipe_bart = pipeline("summarization", model="facebook/bart-large-cnn")
pipe_bart_out = pipe_bart(sample_text)
```

```
Downloading (…)lve/main/config.json: 100%  1.58k/1.58k [00:00<00:00, 104kB/s]
Downloading (…)"pytorch_model.bin";: 100%  1.63G/1.63G [00:08<00:00, 212MB/s]
Downloading (…)neration_config.json: 100%  363/363 [00:00<00:00, 21.8kB/s]
Downloading (…)olve/main/vocab.json: 100%  899k/899k [00:00<00:00, 5.11MB/s]
Downloading (…)olve/main/merges.txt: 100%  456k/456k [00:00<00:00, 3.18MB/s]
Downloading (…)/main/tokenizer.json: 100%  1.36M/1.36M [00:00<00:00, 7.58MB/s]
```

Figure 5: Example of pipeline for the BART model.

5 Topic modeling

Topic Modeling is a text mining technique that allows you to identify the main topics present in a set of documents. Taking advantage of the data coming from the preprocessing phase, it was decided to create a semantic model through the Latent Dirichlet Allocation (LDA) algorithm. The LDA is a generative algorithm, which generates a probabilistic model of the data and allows to calculate the probability of generating a given document starting from the topics. The basic idea of LDA is that each document is generated by the combination of a set of latent topics, which can be identified through the analysis of the words contained in the documents.

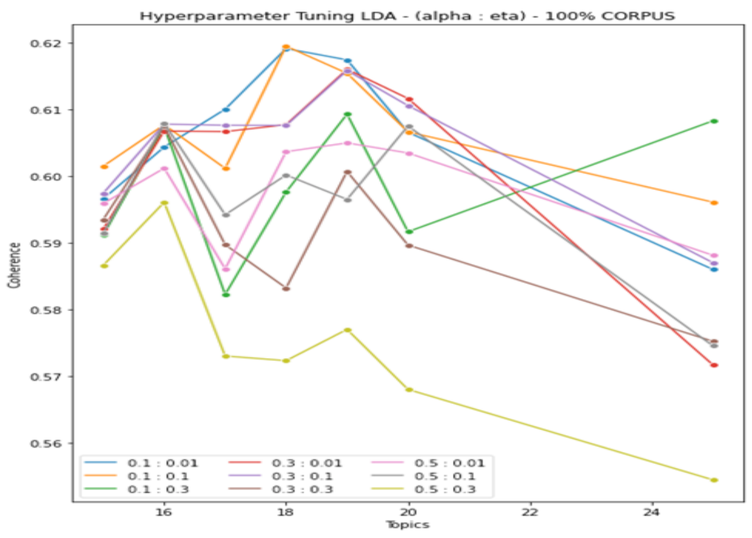
5.1 Input

First of all, the pre-processed tokens (unigrams, bigrams and trigrams) were converted into a dictionary composed of numerical indexes which each uniquely identify a single word with the relative occurrences within the corpus. For the creation of the dictionary, a filtering process was applied in which only the words that occur more than 5 times are considered and which report a marginal frequency greater than 50% within the documents. Then the list of tuples to be passed as input to the LDA was created using the Bag of Words technique.

5.2 Hyperparameter Tuning

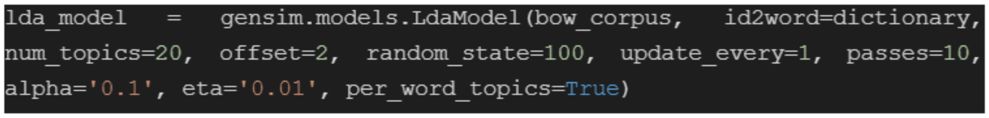
The tuning phase was driven by coherence, Topic Coherence measures score a single topic by measuring the degree of semantic similarity between high scoring words in the topic. These measurements help distinguish between topics that are semantically interpretable topics and topics that are artifacts of statistical inference. There are different coherence metrics, for training our model we choose the C_v measure that is based on a sliding window, one-set segmentation of the top words and an indirect confirmation measure that uses normalized pointwise mutual information (NPMI) and the cosine similarity.

In topic modeling tuning, alpha and beta are hyperparameters for the Dirichlet prior distribution on the document-topic and topic-word relations: a document can have multiple topics (because of this multiplicity, we need the Dirichlet distribution); and there is a Dirichlet distribution which models this relation words can also belong to multiple topics, when you consider them outside of a document; so here we need another Dirichlet to model this. The results are shown in the graph below:



5.3 LDA final model

The model selected to capture the hot topic and the semantic content of the dataset has the following parameters:



```
lda_model   =   gensim.models.LdaModel(bow_corpus,   id2word=dictionary,
num_topics=20,  offset=2,  random_state=100,  update_every=1,  passes=10,
alpha='0.1', eta='0.01', per_word_topics=True)
```

and following performance:

Coherence CV	Coherence Umass	Perplexity
0.626	**-1.723**	**-7.626**

Table 1: Coherence and Perplexity table

The Coherence CV value of 0.62 indicates that the model has good internal coherence. However, the Coherence Umass value of -1.72 suggests that the model may have issues with external coherence. The Perplexity of -7.62 indicates that the model has good generalization ability.

5.4 Conclusion

We can use the performance obtained from the model and the graph in the lda_viz.html file to understand the themes encompassed in the 20 topics:

TOPIC 1	**TOPIC 2**	**TOPIC 3**	**TOPIC 4**	**TOPIC 5**
/	Football	Foundation of Football	Foundation of Football	Fifa & Decision
TOPIC 6	**TOPIC 7**	**TOPIC 8**	**TOPIC 9**	**TOPIC 10**
Barclays Premier League	Football Tournament	Olympiad	Transfer Market	Rugby
TOPIC 11	**TOPIC 12**	**TOPIC 13**	**TOPIC 14**	**TOPIC 15**
Foundation of Football	Baseball	/	Cricket	Formula1
TOPIC 16	**TOPIC 17**	**TOPIC 18**	**TOPIC 19**	**TOPIC 20**
/	Scottish Premier League	Football League One	African Cup	Women's Super League

Table 2: Interpretation of topic detected

The final result of the LDA model shows that the main topic revolves around football, with topics such as the foundation of football, FIFA and decision making, the Barclays Premier League, football tournaments, and the transfer market being discussed. Other sports such as rugby, baseball, cricket, and Formula1 also appear as topics, but to a lesser extent. The model also highlights different football leagues such as the Scottish Premier League, Football League One, African Cup, and Women's Super League.

6 Text Summarization

Text summarization is the technique for generating a concise and precise summary of voluminous texts while focusing on the sections that convey useful information, and without losing the overall meaning. The main goal of this method is to transform long documents into quickly readable summaries. This transformation would require quite a difficult effort by a human being and an automatic method would save a lot of time.

One of the most relevant issues is understanding the actual meaning of the text, since a semantic understanding of the text is required in order to summarize it, although it is usually done using machine learning techniques and probabilistic approach.

There are two main families of methods to solve a text summarization problem: extraction-based summarization and abstraction-based summarization.

Extraction-based summarization is a family of techniques that, starting from the dictionary of words of a specific document, try to determine which are the most relevant words and combine them in order to create a summary. The summarization problem is solved according to a selection criteria of relevant words based on the relevance of each of them, without the possibility of adding new words or synonyms.

Abstraction-based summarization, on the other hand, is based on more advanced deep learning techniques. This family tries to copy a typical human behavior, allowing the model to add and generate words that are considered relevant but are not in the vocabulary of a given document.

For the given analysis, we will use only abstraction-based summarization, since it allows better performance and is more similar to a human approach to the problem. In order to make a comparison, we also use a baseline by just selecting the first three sentences of each document and use it as a benchmark summary.

6.1 Baseline

The baseline pipeline is used only as a benchmark. It consists of the selection of the first three sentences of each document. The idea that the most relevant information of a document is inside the first part is the document was one of the most popular in the early stages of text summarization technique, while it is used mainly only for comparison purposes.

6.2 T5

T5 (Text-To-Text Transfer Transformer) is a transformer model that is trained in an end-to-end manner with text as input and modified text as output, in contrast to BERT-style models that can only output either a class label or a span of the input. This text-to-text formatting makes the T5 model fit for multiple NLP tasks like Question-Answering, Machine Translation and also Summarization tasks like the current one.

Both T5 and BERT are trained with the MLM (Masked Language Model) approach. The MLM is a fill-in-the-blank task, where the model masks part of the input text and tries to predict what that masked word should be.

The only difference is that T5 replaces multiple consecutive tokens with the single Mask Keyword, unlike BERT which uses Mask token for each word.

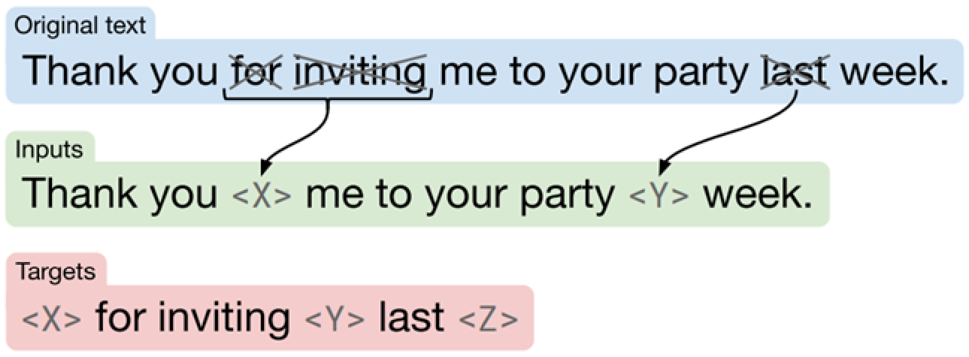

Original text
Thank you ~~for inviting~~ me to your party ~~last~~ week.
Inputs
Thank you <X> me to your party <Y> week.
Targets
<X> for inviting <Y> last <Z>

Figure 6: Image taken from "Raffel, Shazeer and Roberts, Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer"

6.3 BART

BART is a denoising autoencoder for pretraining sequence-to-sequence models. It is trained by corrupting text with an arbitrary noising function, and learning a model to reconstruct the original text. It uses a standard Transformer-based neural machine translation architecture.

That means, It uses a standard seq2seq/NMT architecture with a bidirectional encoder (like BERT) and a left-to-right decoder (like GPT). This means the encoder's attention mask is fully visible, like BERT, and the decoder's attention mask is causal, like GPT2. This means that a fine-tuned BART model can take a text sequence (for example, English) as input and produce a different text sequence at the output (for example, French).

6.4 Pegasus

PEGASUS (Pre-training with Extracted Gap-sentences for Abstractive SUmmarization Sequence-to-sequence) was specifically designed for abstractive summarization and is pre-trained with a self-supervised gap-sentence-generation objective. In this task, entire sentences are masked from the source document, concatenated, and used as the target "summary".

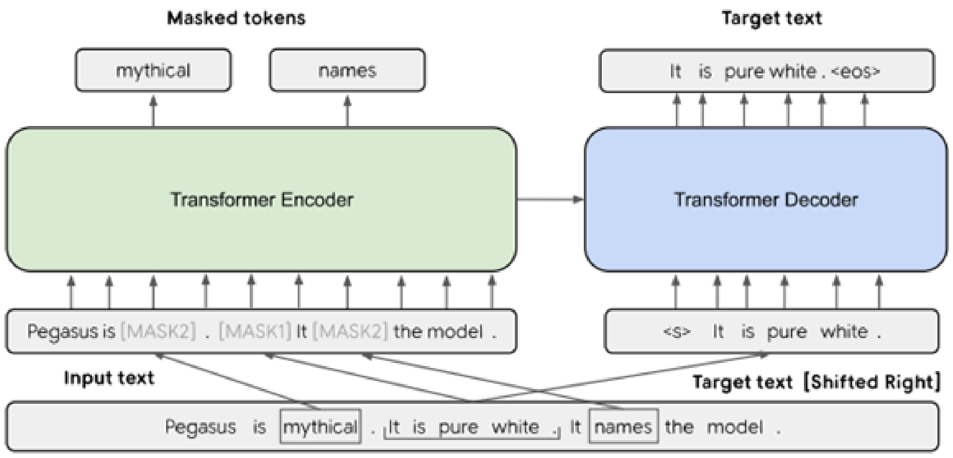

Masked tokens
mythical
names
Target text
It is pure white . <eos>
Transformer Encoder
Transformer Decoder
Pegasus is [MASK2] . [MASK1] It [MASK2] the model .
<s> It is pure white .
Input text
Target text [Shifted Right]
Pegasus is mythical . It is pure white . It names the model .

Figure 7: Image taken from "Zhang, Zhao and Saleh, PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization"

6.5 Evaluation results

The evaluation of the results of a text summarization method is an open issue in the text mining world. The main problem is that it is impossible to determine an ideal summary that can be used as ground truth, like for text classification tasks.

Moreover, the goodness of a summary cannot be measured by just counting the number of common words between the proposal summary and the actual one, since also the readability and the selection of relevant information are crucial aspects.

At the state of the art, one of the most used alternatives of a human evaluation of the results, clearly not a scalable method, is the Rouge statistic, that is the one used for the current work.

6.5.1 ROUGE

ROUGE stands for Recall-Oriented Understudy for Gisting Evaluation. It includes measures to automatically determine the quality of a summary by comparing it to ideal summaries created by humans.

The general formula for ROUGE-n is the following one:

$$\text{ROUGE-N} = \frac{\sum_{S \in \{ReferemceSummaries\}} \sum_{gram_n \in S} Count_{match}(gram_n)}{\sum_{S \in \{ReferenceSummaries\}} \sum_{gram_n \in S} Count(gram_n)}$$

where N is the length of the n-gram chosen.

For the given analysis we are considering two different types of ROUGE-N, that are ROUGE-1 and ROUGE-2, respectively for uni-grams and bi-grams.

A different approach is possible using ROUGE-L. ROUGE-L is based on the longest common subsequence (LCS) between our model output and reference. It means that is considering the longest sequence of words (not necessarily consecutive, but still in order) that is shared between both. A longer shared sequence should indicate more similarity between the two sequences.

For the given analysis we are considering two different types of ROUGE-N, that are ROUGE-L and ROUGE-L Sum. The main difference between the two is that, while ROUGE-L computes the longest common subsequence (LCS) between the generated text and the reference text ignoring newlines, ROUGE-L-sum does the same considering newlines as sentence boundaries.

	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-L SUM
BASELINE	0.168	0.020	0.107	0.108
T5	0.171	0.023	0.117	0.117
BART	0.203	0.041	0.134	0.166
PEGASUS	**0.472**	**0.269**	**0.412**	**0.414**

Table 3: Table with the ROUGE results on the test set

In the following tables, we report the results obtained on a specific document from the test set.

GROUND TRUTH	Defending Pro12 champions Glasgow Warriors bagged a late bonus-point victory over the Dragons despite a host of absentees and two yellow cards.
BASELINE	Simone Favaro got the crucial try with the last move of the game, following earlier touchdowns by Chris Fusaro, Zander Fagerson and Junior Bulumakau. Rynard Landman and Ashton Hewitt got a try in either half for the Dragons. Glasgow showed far superior strength in depth as they took control of a messy match in the second period.
T5	Gregor Townsend gave a debut to powerhouse fijian-born wallaby wing Taqele Naiyaravoro . the dragons gave first starts of the season to wing aled Brew and hooker Elliot Dee . it took 24 minutes for a disjointed game to produce a try .
BART	Rynard Landman and Ashton Hewitt score tries in either half for the Dragons. Chris Fusaro, Zander Fagerson and Junior Bulumakau score tries for the Warriors. Simone Favaro scores crucial try with the last move of the game. Dragons director of rugby Lyn Jones says errors cost his side.
PEGASUS	Glasgow Warriors began their Pro12 season with a bonus-point win over Newport Gwent Dragons at Scotstoun.

Table 4: Table with the summaries on the test set example

From both a human evaluation and an analytical one, it is clear the below summary obtained from the Pegasus model is the best one, still keeping a considerable short length of the summary.

	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-L SUM
BASELINE	0.175	0.025	0.099	0.099
T5	0.187	0.032	0.156	0.156
BART	0.141	0.029	0.113	0.113
PEGASUS	**0.399**	**0.105**	**0.350**	**0.350**

Table 5: Table with the ROUGE results on the test set example

6.6 Fine tuning of the best model

In order to improve the results obtained on the best model, we have implemented a fine tuning strategy on the Pegasus model.

Fine tuning allows us to adjust the weights estimated on a huge dataset in order to become more similar to a specific task. In this case, the goal of the strategy is to adapt the Pegasus model to sport documents, but still starting from the original weights.

To avoid GPU limitations, we randomly selected 10000 documents from the original training dataset and 1000 from the validation one with their respective summaries. The training phase uses 8 elements for each training batch and 4 for each validation one. This step required 8 epochs and around 9 hours of training. After this step the improvement on the validation set was very limited so we decided to not continue more.

	ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-L SUM
PEGASUS	0.472	0.269	0.412	0.414
PEGASUS FINE TUNING	**0.497**	**0.275**	**0.418**	**0.418**

Table 6: Table with the ROUGE results on the test set

The table above shows that the fine tuning model proposed improves the performance compared to the original one. It means that the new version is more capable to summarize sport documents, that is the main goal of the analysis.

In particular, the best improvement is observed on the ROUGE-1 metric, so we can interpret the results as an improvement in recognizing relevant unigrams in a starting document.