Text Mining Project
Text summarization of 20 news groups datset

Davide Meloni*

Universitá degli Studi di Milano-Bicocca
Milan, Italy

Alberto Raimondi*

Universitá degli Studi di Milano-Bicocca
Milan, Italy

Abstract

The following project aims to create a viable unsupervised text summarization algorithm to extract the most representative sentences from a given corpus.

Introduction

Text summarization is the task of creating a textual output that contains the most information from a given input text; there are two kinds of text summarization: abstractive, in which the output text is a new coherent text on its own, and extractive, where the aim is to return the most relevant parts of the input text with not necessity of contextual coherence. The summarization on textual inputs is still an open field of research where the influence of recent advances in the field of deep learning brought new life and progress. Abstractive summarization techniques mostly rely on recurrent neural networks to output sentences that are coherent with the context and make sense with each other, due to the high computational costs of these architectures and their instability an extractive summarization approach was preferred for this project while still using deep learning techniques. The summarization task can be supervised or unsupervised, in the supervised case we have human-made reference summaries that we train our model to predict, this makes the surpassing of human level performance impossible but gives us a clear and efficient metric to compare. In the unsupervised case we incur in the problem of evaluation in which we do not have a clear metric to guide us in the optimisation of our algorithm and we have to rely on case specific metrics.

The dataset

The data-set used is the 20 News groups dataset, it consists in a collection of 18828 text files of e-mails sent on 20 various thematic newsgroups. The length of the files ranges from a few words to some pages. The topic and the writing styles can vary wildly. The author individuates 6 macro-categories in which to cluster the 20 newsgroups that are used for the task of text classification. The dataset was not originally made for text summarization so we don't have access to reference summaries, this forces us to take an unsupervised approach to our task.

Pre-processing

The pre-processing phase is one of the most important and delicate phases of the text mining pipeline, a good choice can make the difference between a good model and an exceptional one, for this reason every step of this phase was take with the maximum care and lot of alternatives were evaluated.

The final pre-processing steps taken where the following: After loading the E-mail we remove the header information by deleting the lines that start with "From:", "To:" and "Subject:"; We then proceed to convert all the words to lowercase and remove the numbers by substituting them with an arbitrary token. After removing some useless punctuation we convert all the "end of phrase" punctuation to a full stop and apply the NLTK sentence tokenizer. The sentence tokenizer splits the corpus into sentences on which we proceeds to apply a word tokenizer and POS (Part-of-Speech) tagger; we then use this information to feed each word to a lemmatizer to convert them in their dictionary form resolving word polysemy. Finally we remove all the stopwords from the sentences and move to the next step.

Vectorization

To feed our sentences to a model we need to compute a standardized representation for them based on their meaning, in this way we can compare them and create an algorithm for summarization. The GloVe model (PSM14) is a set of precomputed word embeddings computed on the various datasets, specifically the one used in our experiments was composed on 300 dimensional vectors trained on the common crawl dataset (Com12). The innovation of the GloVe model from the previous WordToVec like implementation is that the embedding vectors are computed by minimizing a metric based on the co-occurrence matrix of the words, two words that have a high co-occurrence ratio will be have a low distance on the 300 dimensional embedding space. This property and the fact the the resulting vectors belong to an Euclidean space allow us to get a lot information on the meaning of the word in our distributed representation. After the extraction of the representation of each word - skipping the words with no precomputed representation- we computed a representation for each sentence by averaging the vectors of its words. The result is a mapping of every sentence in our dataset to an Euclidean embedding space with 300 dimensions.

TextRank

The TextRank algorithm (MT04) is a variation on the renowned PageRank (PBMW99) algorithm by Google that works on textual documents. The classical implementation of TextRank takes the tokenized texts and builds a directed graph where each sentence (or document) is a node and the edges are weighted by the number of common tokens between them and the weight of the nodes from where the edges originate.

The result is a scoring of our sentences based on their importance in relation with the others and this would allow to extract information but only based on the common words between the sentences. Our approach instead of computing the similarity

* Both authors contributed equally to this work.

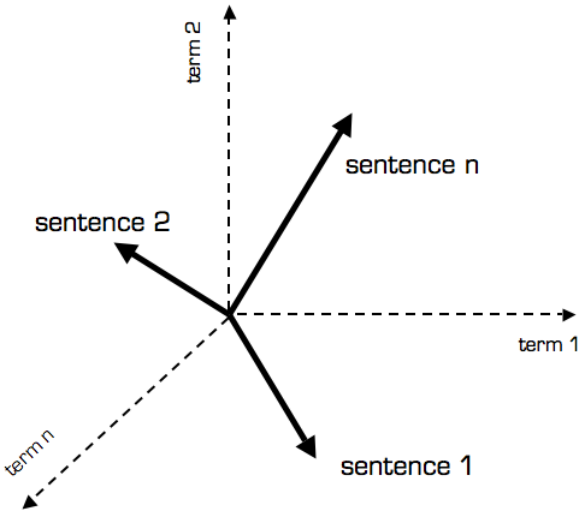

Figure 1: An example of a 3D embedding space

score on the number of common tokens between two sentences computes the cosine distance for each pair of the previously obtained sentence representations and then builds a graph that is fed to a classical PageRank algorithm. The cosine similarity is defined as

$$cos(x, y) = \frac{x \cdot y}{\|x\|\|y\|}, \quad (1)$$

where x and y are two n-vectors, '$\cdot$' is the vector product between the vectors and $\|.\|$ is the L2 norm operator. Therefore the more similar two vectors are, the closer to one is their cosine similarity.

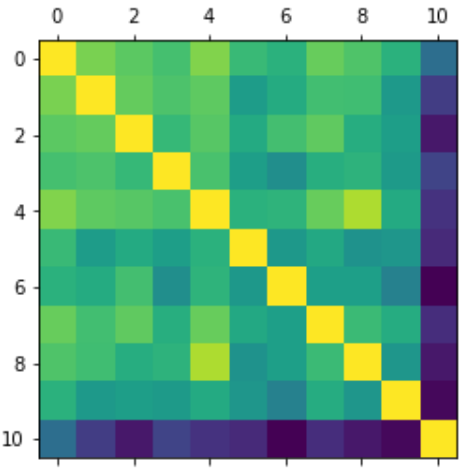

Figure 2: The matrix of cosine similarities between the ten sentences of a corpus.
Notice the low similarity of the last sentence to all the other ones due to it being closing regards, often common on e-mail texts

The result is an importance score based on the average meaning of each of the sentences on the whole text and this allows for effective text summarization.

Summarization

Often the importance score of a sentence is strictly correlated with its length, therefore a long sentence has logically more probability to contain meaningful words, so we can consider to weight the score of each sentence for its length. Then, once ordered the sentences of each corpus according to their scores, we can return the best n-sentences that summarize the corpus in the order they appear in the original text.

Evaluation

The evaluation of the results of text summarization is still an open problem, with a dataset that contains reference summaries we could use a similarity metrics, such as Rouge (Lin04), to compare the output of our model to a human made summary. However with a dataset without reference summaries we have difficulties to get a objective measure of our model effectiveness on the task. Due to the characteristics of the chosen dataset we have access to a kind of target summary that is really effective in virtue of being made by the original author of the e-mail himself: the subject. By using a similarity measure of our model's output with the e-mail subject we can have an effective quantitative metric of how much our model is achieving a representative summary of the e-mail. In our experiments we use the cosine distance between the average vectors of the sentences and the subject on the GloVe embedding space as the similarity measure and the results are encouraging.

Another explored performance metric was the cosine similarity of the average representation of the whole e-mail with the summary sentences that the model extracted for it. This metric was however tautological as the sentences are chosen by their similarity with the whole e-mail and single sentences email would always achieve a perfect score, so we reputed it biased and avoided its usage.

Results

The following is an example of the summarization of an e-mail done by our model.

Target e-mail

From:
mark@madman.demon.co.uk
(Mark Willams)
Subject: HELP WANTED: Faults on IDE drives
I have a 105MB IDE drive and am having a few problems! I get 'Data error on drive C' messages when reading some files. The problem is also steadily getting worse.
I have run some diagnostic software (PCTools V7.1) and it says that the drive is OK - but it does have to retry some sectors and it briefly flashes up an error message (which is too quick to read).
Does anybody know of any cheap or free software which could mark these sectors as bad (DOS doesn't) or preferably perform a low level format. I have heard that the latter is possible on an IDE. Technical answers would be appreciated. It would be nice to be able to use the disk again!!!
I am running MS-DOS 5 on an AT clone.

Summary	I have run some diagnostic software PCTools V71 and it says that the drive is OK - but it does have to retry some sectors and it briefly flashes up an error message which is too quick to read. Does anybody know of any cheap or free software which could mark these sectors as bad DOS doesnt or preferably perform a low level format? It would be nice to be able to use the disk again.

Below is the results obtained on our metric by various hyperparameters configurations

Parameters	Subject-Score	Text-Score
#words=10, α=0	0.714	0.946
#words=50, α=0	0.727	0.969
#words=100, α=0	0.736	0.996
#words=10, α=0.1	0.701	0.961
#words=50, α=0.1	0.722	0.991
#words=100, α=0.1	0.724	0.997
#words=10, α=0.5	0.680	0.952
#words=50, α=0.5	0.711	0.990
#words=100, α=0.5	0.713	0.993

The parameter "#words" represent the maximum number of words for each summary , while α is the power to which we elevate the sentence length, used as weight for the score of each sentence.

Conclusions

The results of the model are encouraging, even if the metric used is not standard in the field we can see that the performance seems good by looking at the results subjectively. The usage of GloVe embedding allow our model to use complex information about the meaning of the words like synonymy and polysemy. The summaries seem to be good encodings for the texts, in fact the Text-Score is very high, even with few words. The Subject-Score is quite lower, but its relation with the summaries is still relevant.

Future perspectives A possible expansion on this work would be the modification of the average sentence vector using a weighted average using TF-IDF to as the weight for each word.

References

[Com12] Common crawl. Online, March 2012.

[Lin04] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*, 2004.

[MT04] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, 2004.

[PBMW99] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab, 1999.

[PSM14] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.