Automatic text summarization, 2018

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Today there are many documents, articles, papers and reports available in digital form. These volumes of text are invaluable sources of information and knowledge that need to be effectively summarized to be useful. In automatic text summarization machine learning techniques are often used to generate summaries. A prior step to the generation of summaries is usually the extraction of nuggets. This paper presents the two approaches we use for the extraction of nuggets, as well as a description of their effectiveness and shortcomings.

1 Introduction

With the dramatic growth of the internet, people are overwhelmed by the tremendous amount of online information and documents. This expansion in availability of data has demanded extensive research in the automatic generation of summaries from a collection of different type of text.

Automatic summarization is the process of shortening a text document with software, in order to create a summary with the major points of the original document.

In general, there are two different approaches for text summarization:*extraction* and *abstraction*

2 Implementation

For all our Implementations we chose two build flexible models which would not only be able to choose whole Sentences as nuggets but also sub parts. Justification for that was drawn from short inspection of the nuggets that workers had chosen. For this we first calculated the percentage of nuggets that were sentences by counting nuggets that start with uppercasing and end on a punctuation mark and dividing it by the total nugget amount. Additionally we also plotted a histogram of nugget lengths that can be seen in figure 1. The results were that only about 49 percent of the chosen nuggets are actual sentences and most of the nuggets are shorter than 15 words. We then came up with two main approaches on how to deal with that flexibility.

2.1 First approach

For the first approach we chose to take a wordwise approach that was also inspired by findings additional to those mentioned above. This finding is that workers may choose different start or endpoints for nuggets leading to problems for approaches that would only consider nuggets important that match exactly for multiple workers. An example in our dataset for this can be found in table 2.1.

Submitted to 32nd Conference on Neural Information Processing Systems (NIPS 2018). Do not distribute.

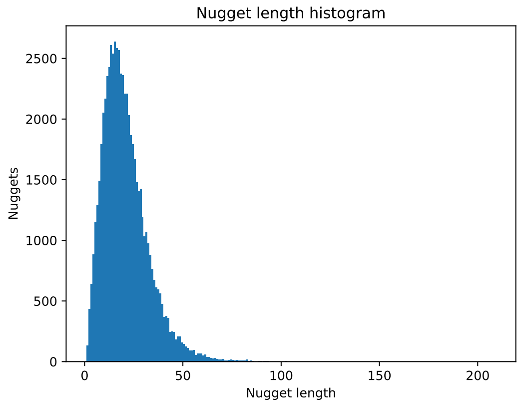

Figure 1: Nugget length distribution

Worker ID	Nugget
87b87beadeaabc197b466e265837af98	While some experts admit that neurofeedback has promise, they believe that it should be used only in combination with medication.
f0a942943de19e11972338c883ad1da2	some experts admit that neurofeedback has promise, they believe that it should be used only in combination with medication.
87954087f1d66c24165db6afc992e136	neurofeedback has promise,
ec7e848297d5f2666f07c0d779ca074d	neurofeedback has promise, they believe that it should be used only in combination with medication.

To avoid loss of such parts of sentences that are contained in multiple nuggets we therefore view each word of a sentence as a training / prediction instance. We do this by taking a word window with size 1 around the particular word of interest which therefore contains the word and its left and right immediate neighbor. All those words are then transformed into word vectors from embeddings pretrained by Google [1] and averaged. Similarily we also transform all word of the query of a specific document into word embeddings and average them. In a third step both averages are averaged together once again to derive our word feature representation. We chose this final averaging step because we thought the arithmetic properties of word embeddings would retain information about the relation between word window and query. This also keeps the feature representation smaller than doing a concatenation for an example and therefore allowed us to still use classifiers that can not train on mini batches.

As labels each word gets assigned a number which represents the amount of nuggets in the same sentence that contain that specific word. The different amounts of workers that chose a nugget then form the label of a classification task. A summarization of the process from a word of a sentence to feature representation can be found in pseudocode in 1. Here it is assumed that a sentence is padded accordingly.

To form nuggets in a sentence we first transform each word into the above described feature representation and then assign a class or score with the trained classifier. Afterwards nuggets are formed by concatenating consecutive words that have scores above a pre-defined threshold δ. The whole process is defined in pseudocode in 2

[1] https://code.google.com/archive/p/word2vec/

E : Pre trained word embeddings
QueryEmbedding : Averaged query embedding
WindowEmbedding : Averaged word embedding of the word window
SentenceWords : List of words in the sentence
QueryWords : List of words in the query

```
for i = 0; i < length(Query); i = i + 1 do
|   QueryEmbedding += Querywords[i]
end
QueryEmbedding /= length(Query);
WindowEmbedding = E[SentenceWords[j-1]] + E[SentenceWords[j]] + E[SentenceWords[j+1]];
WindowEmbedding /= 3;
return (QueryEmbedding+WindowEmbedding) /2 ;
```

Algorithm 1: Feature building process

input : List of words in a sentence
Classifier : The trained word classifier
wordscore : List of tuples containing word and their score
delta : The nugget word threshold
tempNugget : List of words that form a nugget
Result: List of Nuggets

```
foreach word w of input do  wordscore.append(w, Classifier(word)) ;
for (w, score) in wordscore do
|   if score > delta then
|   |   TempNugget.append(w)
|   else if score < delta and tempNugget not empty then
|   |   Result.append(TempNugget) ;
|   |   TempNugget = empty list;
|   else
|   |   skip;
|   end
end
return Result ;
```

Algorithm 2: Nugget prediction process

3 Evaluation

3.1 Nugget Evaluation

We tried to evaluate both our main approaches on the labeled dataset .During training of the second approach we noticed that the neural network used was not able to learn anything other than predicting the majority class. We tried several different network parameters, as well as data sub- and oversampling as well as turning it into a regression task however none of those measures seemed to help to alleviate the problem. This is why we abandoned the approach for the final predictions and also don't report performance measures for that approach. For the evaluation we chose 2 out of the 10 labeled topics and split them into one development set topic and a test set topic respectively. As for all the machine learning models we use the popular scikit-learn python module. Since our computational ressources were limited we only were able to test three different models in a reasonable timeframe such as Decision trees, logistic Regression and Random forests. For the latter training however took significantly longer than for the other two so less hyperparameters were tested. The final scores on the Test set can be found in table 1.

3.1.1 Nugget results discussion

The overall performance of the nugget selection were quite dissappointing and can have many different causes. One obvious factor is the relatively unconservative selection of nuggets which is evident in the high recall scores. This happened mainly because we had to set the threshold of nuggets to 1 as otherwise no nuggets would have been predicted. We tried to sum class probabilities and modelling the task as regression but the heavy imbalance between label values always led to similar

Table 1: Nugget Evaluation Scores

Model	Recall	Precision	F1
DecisionTree	0.901	0.1501	0.257
RandomForest	0.887	0.137	0.237
LogisticRegression	0.891	0.09	0.163

problems in the results. A bigger dataset would have allowed us by the means of subsampling to provide a more balanced class distribution and therefore possibly achieve better results. Also the query or topic should be a rather important information about what is might be a nugget in a sentence. The provided labeled dataset only has ten such topics though which might also not be enough data to allow for a generalizable solution in the first place. However it may also be concluded that the chosen feature representation due to the averaging or relatively narrow view on the sentence does not provide a clear enough boundary between important and unimportant words/nuggets.

To address the unclear effect of this feature representation we originally thought about using our second approach which however failed as well. We mainly guess that it was due to a lack of annotated data as it was easy to create vast amount of negative labeled nuggets but the relatively limited amount of queries and annotated nuggets might be problematic for learning a decently sized recurrent neural network.

3.2 Manual evaluation

The summaries are given to human annotators for evaluation. The annotators are students who attend the same course but are in another work group (?). For evaluation Likert Scales are used. Since reference summaries do not exist it can't be evaluated by comparing a summary with a gold standard. Furthermore the annotators shouldn't have to read all ... source documents of a summary to judge the summary itself. This process woud be too time-consuming. Instead items are used on the Likert Scale which can be judged by only reading the summary itself. In total there are eleven categories: "Grammaticality", Non-redundancy", Referential clarity", "Focus", "Structure", "Coherence", "Readability", "Information Content", "Spelling", "Length" and "Overall Quality". For each category the annotators should assign a score from 1 (= very poor) to 5 (= very good), a weight and a confidence (both scales also from 1 to 5) of their grading. For each category the annotators are also free to give a comment to explain their rating. Each summary is evaluated by four to five different annotators.

Besides the summaries of all groups summaries created by two simple approaches (footnote) are evaluated as well. These summaries serve as baseline summaries. The first approach is ... The second approach is ...

Most categories seem like any text evaluation categories like "Spelling" and "Grammaticality". Other categories seem especially summary-related. These are the categories "Information Content" and "Focus". They represent the goal of a summary very well which is to present the most important content of the summarized texts. Since all summarized texts in this corpus are about a certain query the focus should be visible, too.

The resulting evaluations can be used for assessing the quality of the summaries produced by our system. It is important for the evaluation that we only work at the nugget extraction. This input is given to another group which then produced the summaries. In this way we are completely responsible for the results in some evaluation categories while other evaluation results also depend on the steps of building the hierarchy and actually creating a summary. The output which we after the nugget extraction are whole sentences (more about the output in section ...). The summary is then only built out of these sentences. In this way all categories which just operate on a sentence level are completely our responsibility. Among these categories are strictly only the two categories "Spelling" and "Grammaticality". We are also highly responsible for the categories "Information Content", "Focus" and "Non-Redundancy". All extracted sentences should ideally contain importannt information related to the query. Furthermore it can be argued that in the step of nugget extraction nuggets with the same meaning as another nugget are ignored. The categories "Referential Clarity", "Structure" and "Coherence" in comparison are very dependent on the ordering of the sentences. It can be argued that "Referential clarity" is also influenced by the nugget extraction. For sentences with a pronoun the system should also extract the reference sentence. Otherwise the sentence is not well usable in the next steps. This is not done in the step of nugget extraction, but in later steps. The category "Length" especially depends on the last step, the summary creation. "Readability" and of course "Information Content" are very general categories which can't be assigned to any particular step. The focus of our analysis will be all steps which can be influenced by our work, the nugget extraction. Thus the categories "Structure", "Length" and "Coherence" will only be shortly discussed.

In the following we compare the results of our group with the results of the other groups and the two baseline approaches. Our average overall score is 2.86. The average overall scores of the other groups are 0.39 to 0.74 points better. In contrast to the baseline approaches our summaries are much better. The baseline approaches only have an average overall score of 1.61 and 1.62. So our approach is more than one point better than the baselines. Now we take a closer look at the different categories. "Overall Qualiy" isn't discussec here because it does not highlight a particular aspect of a summary. Compared to the other groups our summaries are worst in all categories except for "Referential Clarity". In the category "Information Content" which is very important for summaries we outdo both baseline approaches significantly at least. The categories we are best at are "Spelling" with ..., "Non-Redundancy" with ... and "Grammaticality" with .... The other groups also perform best at "Grammaticality" and "Spelling"??? This is not surprising since all groups extracted whole sentences for the summarization. These sentences should be mostly grammatical, correctly spelled sentences. Perhaps there are some exceptions since the sentences are taken from forum posts. categories we are worst in are "Structure" with 2.86 points, "Coherence" with 2.88 points and "Information Content" with 2.9 points. "Structure and "Coherence" are also the categories the other groups perfom worst at.

Since we use only full sentences for the creation of the summaries it is surprising that the results uin "Grammaticality" and "Spelling" are not near the maximum score. The comments of the annotators hint at certain repeatedly made mistakes. Many of them are related to the fact that the source texts are taken from forum posts which can contain mistakes like this. Some sentences contain punctuation error like missing dots or quotes. Annotators critisize incomplete sentences like "The study of mechanical self propulsion in vehicles." which often seem like headlines. There are also summaries which consist of only one long sentence like "Developing performance-enhancing behavioral therapies for individuals prenatally exposed to alcohol and focusing remediation efforts on disabilities that affect quality of life and everyday functioning Information about illicit drugs, alcohol, prevention and treatment programs can be obtained on the following websites: Being raised in a family where abuse of alcohol or other substances (illegal drugs or prescription medications) occurs can lead to a host of challenges for children." All these problems can be solved in different ways. A possible solution for punctuation errors is to check if a sentence ends with a punctuation sign and to check if parentheses and quotes are properly closed. For the removal of incomplete sentences a POS tagger can be used. It should check if a sentence contains at least a noun and a verb. Extremely long sentences can be just filtered out with a certain threshold length. In this way also too short sentences which can also cause problems can be filtered out.

Now we take a look at differnt errors in rhe category "Spelling". This category contains some punctuation errors, too. It seems like annotators do not know in which category these kinds of errors belong. In this case the annotation protocol needs to be specified. A mistake unique to the category "Spelling" is incorrect upper- and lowercasing. Another mistake is wrong whitespacing, like in "loans , you". The upper- and lowercasing could be handled by a POS tagger so that only proper names are uppercased and everything else is lowercased. Additional whitespaces can be easily removed with a regular expression.

As we see the categories "Grammaticality" and "Spelling" contain many mistakes which can be fixed quite easily. That means that actual improvement in these categories can be achieved well.

Now we will take a look at the categories "Information Content", "Focus" and "Non-Redundancy". "Information Content" is one of our system's greatest weaknesses. Annotators' comments point towards the relatedness of "Information Content" and "Non-Redundancy", "Focus" and "Readability". If a text contains only one fact over and over, if it contains facts unrelated to the topic or if it is not understandable there is no real information gain. So it is very important to optimize the results in these categories to impart as many information as possible. The score in focus of 3.14 is much better than of baseline 1 (2.15) but slightly worse than the score in "Focus" of baseline 2. We integrate the query in our nugget extraction by averaging the query with a nugget. It seems like we need addditional features to incorporate the query. This can be focus of future work. The results of our system of our system in "Non-Redundancy" are worse than the ones of baseline 1 but similar to the results of group 5 and baseline 2. The similarity to group 5 is very interesting since we used the pipeline after the nugget extraction from this group. It hints that group 5's system does not properly remove duplicates while creating a summary. An extreme example is the following summary which consists of four sentences with a content nearly identical: "Computer Explorers uses innovative and creative ways to excite young learners about science, technology, engineering and math subjects. The local Computer Explorers uses technology in creative ways to engage students in science, math, English and other core academic subjects. Computer Explorers is an education company that uses technology in creative ways to engage students in science, math, English and other core academic subjects. Computer Explorers is a local education company that uses technology in innovative ways to engage students in science, math, English and other core academic subjects". It sees like no similarity detection is used. This does not nessecarily have to be done in summary creation but can be also done in the nugget extraction, at least if full sentences are extracted.