John Zavidniak

CSCI 4830

Automatic Summarization

Automatic summarization is type of natural language processing program which takes some source text as input and produces a summary. The summary should be unambiguous, it should avoid redundancy, and it should contain the sentences which human readers would define as important and necessary for understanding the material covered by the source text. In modern times many articles are written by many different people covering a wide variety of topics. There is so much new information available each and every day such that it would simply not be possible for anyone to read everything posted in order to keep up with all that's new. One solution to this problem is to have a summary associated with each new article written each day. Of course, this would require quite a large amount of people and would essentially make this problem impossible to solve. However, with automatic summarization these summaries may be produced and associated with any given article automatically. It would allow people to quickly browse through short summaries to get the gist of the recent news and enable them to choose specific events to read further into. Following that idea, good automatic summarization is absolutely essential for search engines. One may type in a search including keywords related to a recent event that occurred, and the search engine may return a vast number of articles written about this event. How does the user choose which article to read? A summary attached to a link to each article is one way to do this. The user can simply determine which summary contains the information they want know, then click the link to the associated article. Due to these various applications of the problem, as well the apparent intrinsic value of the problem, it seems that working on a solution is worthwhile.

Before discussing my solution, I would like to take some time to look at previous solutions. By doing this it will be easier to contrast my solution with others and determine what is and is not unique about what I have come up with. One approach I found to be quite interesting is the one taken by

Houda Oufaida, Omar Nouali, and Philippe Blache. Their solution can be found at the end of the paper. Though they tackled the problem of automatic text summarization for Arabic texts, the methods used may also be applied to texts of other languages, including English. The solution which they implemented involves using an algorithm known as Minimum redundancy and maximum relevance (mRMR), which examines the source document and pulls out a set of sentence which have as little in common as possible with each other while maintaining relevance in accordance with the information presented in the source text. This method is an extractive-based summarization as opposed to abstractive-summarization. It is argued in their paper that abstractive based techniques require too much NLP processing, considerably slowing down the algorithm. Furthermore, they argue that even if the current algorithms could be ran at an acceptable speed, they still are not mature enough to produce a summary which would be comparable to an extractive-based summarization.

Another solution which differs quite a bit from the one previously mentioned has been produced by Minqing Hu and Bing Liu. Hu and Liu have implemented an automatic summarization algorithm for summarizing customer reviews of various products. To summarize their solution (sadly not by an automatic summarizer), they rely heavily on a specific method of feature identification and extraction. This is done is three steps, the third of which will be discussed in more detail:

1. Mine product features which have been identified by customers.
2. Identify opinion sentences in each custom review, and determine the polarity of the review.
3. Summarize the results.

The workings of step three, the sumarization step, will now be described in detail. The algorithm first groups together sentences containing product features of similar polarity (positive features and negative features). Those sentences are then ordered by the frequency of each feature mentioned, and finally are output into two summaries: A summary of positive product reviews, and a summary of negative product reviews. It is worth noting that Hu and Liu have completely ignored the

issue of pronoun resolution, which may lead to certain features being missed as some mentioned features may be unable to be associated with a certain product.

Both of the two solutions are powerful and work to some extent in their given domain, but have certain shortcomings which my solution is able to overcome. The first thing my solution does when summarizing is, when given a list of sentences S, produces a new list of sentences S'. This is done in such a way that S' should contain the same information as S, but without some of the redundancy that may be present in S. I accomplish this in a rather simple but effective way. Ignoring stop words, if two sentences share thirty-percent or more of the same tokens, only the longer of the two sentences is included. Thirty-percent may not seem like a lot, but consider the following two sentences from one of the articles used to test this summarization algorithm:

- "Boulder police investigating phone scammers pretending to be officers."
- "Boulder police are investigating a phone scam in which callers pretending to be officers ask their targets for money to dismiss arrests warrants or fines, according to a news release."

These sentences share the six words "Boulder", "police", "investigating", "phone", "pretending", and "officers", and the average length of the two sentences (ignoring stop words) is 14. This results in a ranking of 42.8% similarity, and by reading the two sentences it is clear that the second sentence contains all the information mentioned in the first sentence plus more. Therefore, the second sentence may be included with the first left out with no loss of information. This example shows the usefulness this simple solution provides, and by doing this as the first step, other more sophisticated algorithms may be applied on S'.

The algorithm implemented in this solution is TF-IDF, which immediately is able to gain performance better than both of the mentioned solutions. The solution proposed by Oufaida, Nouali, and Blache does not take into account term frequencies in related but different documents. As a result,

sentences which they pick out as relevant may in fact be nothing of great importance. If a user wants to read one of a few articles on the same subject, picking one based on a summary produced by mRMR would prove to be difficult due to the similar summaries they would all have. This would occur due to articles on similar subjects including similar vocabulary, and sentences containing certain keywords in all articles would likely be chosen by mRMR to be included in the summary of each article, resulting in similar summaries for potentially unique articles. My solution avoids this by comparing term frequencies across a multitude of articles. Now, if it was purely TF-IDF then issues solved by mRMR would occur – most notably issues regarding redundancy. TF-IDF does not perform any extensive redundancy checks. Though it does factor in term frequency in the source document, there are cases where this simply would not work. Consider a long article which contains two sentences that would be caught by my redundancy checker. These two sentences may contain certain terms which appear in them and nowhere else. Therefore they would be scored highly by TF-IDF and included in the summary, resulting in a redundant summary. However, because my redundancy checker is ran before TF-IDF is ran, one of those two redundant sentences would be removed, and therefore TF-IDF will produce a summary not containing the redundancy it previously would have contained. It is then clear that my solution takes the idea behind mRMR and applies it to TF-IDF in order to get a unique summary with a minimal amount of redundancy. This is a solution which was not found implemented in other automatic summarization implementations.

This automatic summarization algorithm also has a solution for pronoun resolution, something overlooked by Hu and Liu. When parsing a sentence, two searches are done: The first is for singular and plural proper nouns (NNP and NNPS), and the second is for personal and possessive pronouns (PRP and PRP$). If a sentence contains a PRP or a PRP$ but not an NNP or an NNPS, then it is decided that there exists a pronoun which refers to a noun which has not been defined. To solve this, a method of backtracking is utilized. The sentences which have already been looked at are searched in a last-to-first order. The first sentence to contain an NNP is taken and added to the summary. This works

because the first time a PRP or PRP$ is used, there is a very high probability that it will be mentioning the last NNP or NNPS defined. It is also worth noting that there is a check done which ensures that this backtracking does not add a sentence to the summary which was also previously added. Therefore, this solution is capable of producing a summary which resolves unreferenced pronouns – something Hu and Liu's solution overlooks. Oufaida, Nouali, and Blache also fail to mention anything about pronoun resolution when discussing their solution, and since this is not a feature handled by mRMR, it may be concluded that they failed to implement this as well.

It can then be concluded that this solution, which removes redundancy, applies a multi-document analysis of term frequency, and handles pronoun resolution, is a novel solution which is capable of outperforming many other automatic summarization implementations. The proof of the validity of this summarization algorithm will be outlined below. First, two people have agreed to read all the articles along with the associated summaries produced by this automatic summarizer. They have been instructed to grade the summaries on a scale of one to five, with a score of one signifying disagreement and a score of five signifying agreement, for the following characteristics: redundancy, ambiguity, information, and readability. The table below outlines the result of this grading on ten summaries of ten articles:

	Redundancy	Ambiguity	Information	Readability
Summary 1	(1, 1)	(2, 1)	(4, 5)	(5, 5)
Summary 2	(1, 1)	(1, 1)	(4, 4)	(4, 3)
Summary 3	(1, 1)	(1, 1)	(2, 3)	(4, 5)
Summary 4	(4, 2)	(4, 3)	(3, 5)	(2, 3)
Summary 5	(1, 1)	(5,5)	(1, 3)	(4, 5)
Summary 6	(1, 1)	(1,1)	(5, 4)	(5, 5)
Summary 7	(2, 4)	(4, 2)	(4, 5)	(3, 4)
Summary 8	(1, 2)	(2, 1)	(5, 5)	(5, 4)
Summary 9	(2, 2)	(3, 1)	(4, 4)	(3, 5)
Summary 10	(1, 1)	(1, 1)	(5, 5)	(3, 3)
Averages	(1.5, 1.6)	(2.4, 1.7)	(3.7, 4.3)	(3.8, 4.2)
Final Average	1.55	2.05	4	4

From looking at the table, it is clear that the summarizer performed quite well. The biggest issue was ambiguity, and the cause of this seems to be the pronoun resolution algorithm. Since the gender of a noun referring to a person is not taken into account, trying to resolve the pronoun “she” or “he” can lead to problems of choosing a noun referring to the incorrect gender. The best score belonged to redundancy which got a tiny 1.55/5, showing that the redundancy removing algorithm works as expected. Finally, both readability and information scored quite highly, indicated that the summarizer is picking the correct sentences as well as organizing them in an easy to understand way.

Though this automatic summarization program has accomplished quite a bit and can produce unambiguous non-redundant readable summaries full of relevant information, there is still much improvement to be made. The first of which is adding some method to determine the gender of nouns referring to people. This would improve the pronoun resolution algorithm, and thus improve the ambiguity score. Second, readability took a hit most likely due to how direct quotes are parsed. Currently each sentence inside of a quote is being treated as its own unit when parsed by TF-IDF, and this causes some quote segments to be included in the summary, but not the entire quote. This hurts readability because if the first sentence of the quote is chosen, then there is an opening *“* without a closing *”*, and the same applies if the last sentence of the quote is chosen. Finally, if a sentence in the middle of the quote is chosen, then the reader cannot even recognize that it was a direct quotation, and context may be lost thus hindering readability. Though neither of these features are implemented currently, they will certainly be the first revisions made if this project is worked on in the future.

This automatic summarizer has achieved quite a bit by utilizing a combination of various algorithms all working in unison. Ambiguities are crushed, redundancies forced to vanish, the most important information is brought forth, and the text produced is readable and extraordinary. It is able to tackle problems simply not handled by pure mRMR or TF-IDF solutions, and it is able to outperform the solution written by Oufaida, Nouali, and Blache in certain areas, as well as the solution produced by

Hu and Liu. I look forward to continuing work on this project in the future and seeing just how advanced it will get.

References:

http://www.sciencedirect.com/science/article/pii/S1319157814000329

http://dl.acm.org/citation.cfm?id=1014073

http://www.diva-portal.org/smash/get/diva2:352481/FULLTEXT01.pdf

http://nlp.stanford.edu/software/tokenizer.shtml

http://stackoverflow.com/questions/1833252/java-stanford-nlp-part-of-speech-labels

http://nlp.stanford.edu/IR-book/pdf/06vect.pdf

http://i.stanford.edu/~ullman/mmds/ch1.pdf

http://www.emeraldinsight.com/doi/full/10.1108/00220410410560582