AutoSurvey: Extractive Multi-Document Text Summarization for Generating Literature Surveys

Shane Rhoads
email

Kobee Raveendran
kobee.raveendran@knights.ucf.edu

1. PROBLEM STATEMENT

Literature surveys are a valuable resource for researchers exploring a foreign technical domain. They serve as a centralized collection of information on recent or foundational works in a digestible manner, and give the reader a solid conceptual, comparative understanding of the works discussed.

However, obtaining a comprehensive compilation of influential research requires extensive manual effort, and intuitively portraying the numerous concepts and methods requires vast domain-specific knowledge. Without spending significant time, it is also difficult to find relationships between papers not directly cited by each other. For these reasons, literature surveys can be scarce, particularly in new or unpopular research areas.

To alleviate the burden of manual survey creation, we seek to develop a system capable of automatically generating meaningful, high-level literature surveys. We formulate the problem as a combination of multi-document summarization (MDS) and single-document summarization tasks. Multi-document summaries serve to outline the relationships between the selected papers, useful for introduction or conclusion sections of literature surveys. The single-document summaries allow for more detailed coverage on the intricacies specific to each paper, providing the bulk of a typical related works section.

The two most prominent approaches to text summarization are abstractive and extractive summarization. Abstractive methods attempt to generate a *novel* summary by grasping the key topics from the text and generating grammatically-sound sentences from scratch. Extractive methods instead generate summaries by selecting important sentences directly from the source text.

While abstractive summarization is ideal and much closer to human-level summarization in theory, in practice most abstractive methods may face issues producing coherent summaries on single documents [1], and even moreso across multiple documents. This is partly due to the burden of learning the grammatical structure of sentences being placed on the model, a responsibility absent in extractive approaches.

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

Among abstractive methods that perform decently, all require immense compute resources, training time, and training data to work well. Since full-paper datasets of sufficient size are not widely available, we must parse content from downloaded papers individually, making the data-reliance of abstractive methods severely crippling. Even if such a dataset were available, abstractive summarization may not fare well if given only one document per topic, or several loosely-related documents. In abstractive summarization datasets like WikiSum [2], each reference document is likely to cover the same central topic, with minimal divergence. In related research papers, however, the overarching task may be shared but the bulk of the paper's content lies in the proposed method, which is fundamentally different from the material in the other papers.

This is an introduction part, in which you should include a clear motivation on the problem being addressed in this project (why should I care?). You will need to also define the problem in broad terms so that you can outline the motivation.

As the motivation is made clear, you want to also state the problem more formally; in terms that an NLP expert will understand.

2. RELATED WORK

At this stage, you don't want to provide a comprehensive list of related work, but rather you want to consider this as a part in which you will provide a list of the resources that will be used to assist you conducting the project. Find out some of the related work that would be relevant to this project, and summarize how similar or different your work to them is. Even better, highlight in broad terms what would the δ you think you will achieve by this magnificent work be.

2.1 Latent Dirichlet Allocation

2.2 TextRank

2.3 GrassHopper

3. HOW TO TRAIN YOUR DRAGON

To generate multi-document summaries, we considered expanding on existing extractive multi-document summarization methods. Extractive summarization entails selecting key sentences from the source texts, and using them directly in the summary. First, documents must be segmented into sentences, each assumed to convey its own idea. Then, sentence vectors can be compared (currently exploring methods for this), ranked and selected for the final summary. This process can be modified and repeated for individual documents to form more detailed explanations of each constituent paper in isolation.

Due to the limitations of current summarization methods, we will only be considering conceptual content in the form of sentences. For other facets of papers, such as figures, tables, and formulas, extractive methods may not adequately capture meaning or properly judge relevance to textual concepts, so they will be disregarded in preprocessing.

Here should be the actual proposal. What methods are you proposing? Describe the method you will use in designing your training method for your dragons. Relate to the problem statement. The description should be specific so that reproduction of the training method you used for your dragon are reproducible. Avoid copying others' work and others' style, and be creative.

4. EVALUATION

We will use existing literature surveys and a set of their associated papers as the human-generated summaries and source texts to benchmark against. We do not need a large training set, only a set of documents per summary, so we will compile this data manually.

Since summaries for a set of documents are not absolute (i.e. many valid summaries can exist for a set of documents), we will use quantitative and qualitative metrics to evaluate our method. The generated surveys can be evaluated quantitatively by computing ROUGE [3] scores between the generated and reference summaries. ROUGE scores typically measure overlaps between the reference and generated summaries, and use either N-grams or common subsequences (LCSS). However, since human summaries are often abstractive and paraphrased, ROUGE scores can be slightly misguided, so we will also provide the generated and reference summaries for human qualitative comparison.

In this section, you want to describe two things: the data that you will use for evaluating the method against the problem stated above, and the results. As for data, describe your source of dragons, in as much details as needed, but not too much that I won't have the time to read. Be realistic.

Here, I know that you won't have results, so don't worry. Describe to me the evaluation metrics that you will use for evaluating the approach on the dataset above. Describe concisely the steps you propose to use for evaluation against those methods/metrics. The description should be intended for the non-expert so that she is able to reproduce the results of your work.

A bonus would be if you could propose to compare your work against a baseline.

5. EXPECTED OUTCOMES AND RISKS

Here is your chance to tell me what you expect of outcomes.

Also you want to tell me what are the risks associated with the project, and how you plan to deal with them.

6. PLAN AND ROLES OF COLLABORATORS

Divide your project into components, and tell me who is going to work on what, how much time each will work on each item. Have a timeline for the project. Tasks may include coding, testing, evaluation and analysis, write-up, presentation, etc.

6.1 Components

6.1.1 Paper text extraction

Kobee: ∼1 hour

6.1.2 Method Brainstorming and Research

Shane: ∼10 hours
Kobee: ∼6 hours

6.1.3 Data Preprocessing

TODO
Both: ∼6 hours

6.1.4 Method Adaptation/Implementation

TODO Both: ∼20+ hours

6.1.5 Writeup

Both: ∼8-10 hours

6.2 Timeline

Component	Estimated Date
Text extraction [code]	2/20
Method brainstorming & research	3/5
Preprocessing pipeline [code]	3/10
Implementation [code]	4/2
Writeup	4/9

7. REFERENCES

[1] K. Eskici and L. A. Perez, "Multi-document text summarization," 2017.

[2] P. J. Liu, M. Saleh, E. Pot, B. Goodrich, R. Sepassi, L. Kaiser, and N. Shazeer, "Generating wikipedia by summarizing long sequences," *CoRR*, vol. abs/1801.10198, 2018.

[3] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*, (Barcelona, Spain), pp. 74–81, Association for Computational Linguistics, July 2004.