Memorial University of Newfoundland **Fall Semester 2018**
Jacob House and Noah Gallant **Computer Science 4750**

Course Project Proposal

Motivation to study computerized text summarization — referred to by the term *automatic abstracting* by those in the field — stemmed from curiosity about the mechanisms used in an automatic abstraction PowerShell script found years ago on the Internet [11]. Further research has shown automatic abstraction's usefulness in many interesting fields, including but not limited to the legal and medical professions, scholarly research, and search engine result sorting and summarization.

Three previously-studied methods of NLP-based automatic abstraction have been considered: nested trees, evolutionary algorithms, and graphs for evaluation and summarization, as well as basic sentence extraction.

The nested tree structure method uses both inter-sentence and inter-word dependencies. A document is represented as a nested tree composed of both document trees, which have nodes representing sentences, and sentence trees, which have nodes for words. The relationship between sentence and word nodes is defined by this inter-sentence and inter-word dependency relationship. Performing text summarization involves trimming the tree of less important information until the desired compactness is reached [6].

Evolutionary algorithms (EA) encompass many techniques, however the most common of these uses algorithms to assign weights based on text features. This provides more accurate weighting of important and unimportant information. Examples of such features include assigning a score based on sentence location (e.g. awarding 1 to the first sentence, $\frac{4}{5}$ to the second, and so on, until we reach the fifth sentence which scores $\frac{1}{5}$; remaining sentences receive a score of 0), font style (e.g., scoring capitalized text higher), numerical content (e.g., scoring sentences with quantitative numerical information higher), and word similarity, which assigns a weight based on how often a word appears in different sentences. An EA fitness function then combines the weights to produce a complete text summarization [8].

Graph-based text summarization assigns each sentence a node and edges are used to connect both sentence nodes that have common words and sentences appearing next to each other in the text. The summarization is done by evaluating the nodes with the most edges as important sentences with higher fitness [7].

The Implementation of this design will largely follow the graph based method outlined above. This will involve a singleton class encompassing the entire text, this singleton will contain nodes representing every sentence in the text. Each sentence node will be connected to its adjacent sentences in the text with an edge; sentence nodes that share common words will also be connected with an

edge. To make the detection of similarities between sentences easier, the words from each sentence will be stored as a Trie-DFA. A summary of the text will be made by returning the sentences with the largest number of edges. The number of sentences returned for the summary will depend on the provided summary constraints.

This method of forming a summary will extract the importance of the text by finding the similarities between sentences and is therefore expected to be more ideal for larger amounts of texts as a single paragraph is comprised of very few sentences which would result in very few comparisons between sentences.

Pre-processing is also a requirement for this implementation. This involves separating the text into individual sentences to be added to nodes. Splitting a text into sentences is more complicated than simply breaking a sentence at a period, as periods do not always terminate sentences. In many cases, periods are used for abbreviations or in other special situation, and marking these as individual sentences could fragment the summary.

Some Concerns with this implementation include the difficulty of duplicating the same topic, which through our research is a problem that is common, but not yet perfectly solved. This implementation also will not synthesize summary sentences; rather it only returns a subset of existing sentences, which may appear broken and unformatted.

Note that only four of the references included below have been cited in the above text. References not cited will be included in the final report and are present to demonstrate that sufficient relevant literature is available.

References

[1] H. P. Edmundson. "Problems in Automatic Abstracting". In: *Commun. ACM* 7.4 (Apr. 1964), pp. 259–263. ISSN: 0001-0782. DOI: 10.1145/364005.364088. URL: http://doi.acm.org/10.1145/364005.364088.

[2] H. P. Edmundson and R. E. Wyllys. "Automatic Abstracting and Indexing — Survey and Recommendations". In: *Commun. ACM* 4.5 (May 1961), pp. 226–234. ISSN: 0001-0782. DOI: 10.1145/366532.366545. URL: http://doi.acm.org/10.1145/366532.366545.

[3] Mahak Gambhir and Vishal Gupta. "Recent automatic text summarization techniques: a survey". English. In: *The Artificial Intelligence Review* 47.1 (Jan. 2017). Copyright - Artificial Intelligence Review is a copyright of Springer, 2017; Last updated - 2018-10-06, pp. 1–66. URL: https://search-proquest-com.qe2a-proxy.mun.ca/docview/1857255406?accountid=12378.

[4] Jaya Jagadeesh, Prasad Pingali, and Vasudeva Varma. "Sentence Extraction Based Single Document Summarization". In: (Jan. 2005).

[5] Gunnel Källgren. "Automatic Abstracting Content in Text". In: *Nordic Journal of Linguistics* 11.1-2 (1988), pp. 89–110. DOI: 10.1017/S0332586500001761.

[6] Yuta Kikuchi et al. "Single Document Summarization based on Nested Tree Structure". In: *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Baltimore, Maryland: Association for Computational Linguistics, 2014, pp. 315–320. DOI: 10.3115/v1/P14-2052. URL: http://aclweb.org/anthology/P14-2052.

[7] Nandhini Kumaresh and Balasundaram Sadhu Ramakrishnan. "Graph Based Single Document Summarization". eng. In: *Data Engineering and Management: Second International Conference, ICDEM 2010, Tiruchirappalli, India, July 29-31, 2010. Revised Selected Papers*. Vol. 6411. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 32–35. ISBN: 9783642278716.

[8] Yogesh Kumar Meena and Dinesh Gopalani. "Evolutionary Algorithms for Extractive Automatic Text Summarization". In: *Procedia Computer Science* 48 (2015). International Conference on Computer, Communication and Convergence (ICCC 2015), pp. 244–249. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2015.04.177. URL: http://www.sciencedirect.com/science/article/pii/S1877050915006869.

[9] Dragomir R. Radev. "Experiments in single and multidocument summarization using MEAD". In: *In First Document Understanding Conference*. 2001. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.119.4643.

[10] J. E. Rush, R. Salvador, and A. Zamora. "Automatic abstracting and indexing. II. Production of indicative abstracts by application of contextual inference and syntactic coherence criteria". In: *Journal of the American Society for Information Science* 22.4 (), pp. 260–274. DOI: 10.1002/asi.4630220405. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/asi.4630220405. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/asi.4630220405.

[11] Prateek Singh. *PSSummary*. 2018. URL: https://bit.ly/2yzK67O.

[12] Caroline Uyttendaele, Marie-Francine Moens, and Jos Dumortier. "Salomon: Automatic Abstracting of Legal Cases for Effective Access to Court Decisions". In: *Artificial Intelligence and Law* 6.1 (Mar. 1998), pp. 59–79. ISSN: 1572-8382. DOI: 10.1023/A:1008256030548. URL: https://doi.org/10.1023/A:1008256030548.