Inverse Hierarchical Multi-Document Summarization

Frank Bruni
UC Berkeley MIDS
frankbruni@berkeley.edu

Brandon Scolieri
UC Berkeley MIDS
brandonscolieri@berkeley.edu

Daphne Yang
UC Berkeley MIDS
daphne.yang@berkeley.edu

Abstract

This paper discusses the implementation of an inverse hierarchical text summarizer that creates abstractive summaries from seeded text data. Subsequent to generating a summary from a primary source (e.g. a news article), the process is repeated on a wider corpus of data (e.g. NYT articles for a certain range of time). The summaries are then clustered by content similarity via cosine similarity. Multi-document summarization using the proposed iterative pipeline architecture allows for more coherent and factually accurate contextual summaries.

---

1 Introduction

Multi-document summarization (MDS) is an important next step to combat information overload. The summarization of numerous news documents on a single topic would allow users to familiarize themselves with the context behind global events. News articles are generally presented as a single snapshot in time. MDS offers an opportunity for a well-rounded overview of an event. By providing context on the streams of events leading up to the article, readers can be more well informed when coming to their own conclusions and perspectives. With drawbacks in both extractive and abstractive models and the novelty of unified modelling, we propose the development of several summary pipelines to accomplish our task. According to research from Fabbri (2021), there is no current consensus on how to validate and score abstractive summaries. To address this issue we use multiple validation techniques (BLEU, ROUGE, Saturation Score (our own validation metric), and manual validation).

2 Background

Christensen (2014) utilized a newer approach to summarization called hierarchical summarization. Their work leveraged semantic relationships between article text to create a hierarchy of relatively short summaries that were ordered to allow users to “drill down for more details” on topics of interest in any given article. Their findings indicated that human subjects preferred hierarchical summaries ten times as often as flat multi-document summaries. This outlines a strong case for “inverse” hierarchical summarization to allow for broad summaries of related articles as context.

Wang (2019) used a similar approach to MDS by creating extractive summaries as inputs for more accurate abstractive summaries. Wang used reinforcement learning for an additional layer of complexity. Wang notes that by starting with an extractive model to provide inputs into their abstractive model, their model was able to meet slightly higher ROUGE-Avg scores. Reinforcement learning required higher compute

power and longer training times to achieve higher ROUGE scores.

A key tenet of MDS is clustering more similar documents using varying methods. In earlier works, such as in Erkin and Radek (2004), the use of tf idf weights were used to determine clusters of documents within a set. Later, in Liu and Lapata (2018), latent dirichlet allocation (LDA) statistical modeling was used to cluster text. LDA clustered text based on the idea that each document is a mixture of a number of topics and the presence or absence of words can be attributed to a document's overall topic. Drawbacks arise with LDA as the text documents become shorter in length, making it disadvantageous for our article text data.

Moreover, the purpose of our research differs as we aim to provide broader summaries for contextual support of a given article. However, past work on hierarchical summarization and similar works help to provide a framework from which we developed our own understanding.

3. Methods

3.1 Task

In our work, we used both an extractive BERT-based model as well as a pre-trained Sequence-to-Sequence BERT-based model to create varying model pipelines for inverse MDS. We aim to create pipeline architecture for summarization tasks specifically in the use case of allowing users to be able to obtain a high level summary of relevant background information to a news article of interest. We expect the machine-based metrics to score poorly due to the current lack of consistent and accurate appropriate summarization metric, as outlined in Fabbri (2021).

3.2 Data Collection

Earlier research into summarizers leveraged the original corpus of CNN/DailyMail news articles with hand-labeled summaries to assess a model's proficiency in a summarization task as published in Hermann (2015). While this corpus has allowed for relative success in the summarization of full articles using models including BERT and its variants, our interest in MDS of news articles, relative to date of publication, required us to collect a new dataset of news articles with metadata for each document containing key fields including: *Headline, Publication Date, First Paragraph,* and *Abstract.*

Our dataset was sourced from the New York Times API and contains only the first paragraph of the article due to limitations within the API service.

3.2.3 First Paragraph / Summaries

We believe that this usage of the first paragraph as seed text for MDS does not hinder the summarizer's ability to extract meaningful relationships within the text. According to Fang (1991), the writing style of newspaper articles includes a lead paragraph with the main points of the article. This differs from current summarized metadata in an article and therefore, we believed that the use of the first paragraph as a "document" was justified given a newspaper article's unique writing style. According to the *New York Times*'s official API documentation, the abstract of the given article refers to the editorial summary of the article -- thus, we used the abstract as a reference summary.

3.2.2 Data Pre-processing

We processed our data by ensuring that all entries within our dataset contained both text for the first paragraph, which we used as our text

data, and text for the abstract, which we used as our reference summaries. We determined that the minimum threshold of characters to be deemed a "valid" text entry would be above 10 character string length. We find through EDA that our initial dataset contained insufficient entries with the following cases:

1. Insufficient first paragraph text (0.63% of corpus - 1097 articles)
2. Insufficient abstract text
3. (0.01% of corpus - 16 articles)
4. Insufficient data for both fields (0.001% of data - 2 articles)

Insufficient text from the abstract would be particularly troublesome as there would be little to no reference from which to perform our evaluation metrics. Insufficient text from the first paragraph could jeopardize the validity of our dataset. We defined inclusion criteria as those articles where both the first paragraph and abstract text is available and kept all qualifying examples in our dataset. Additionally, we manually reviewed the titles of the removed articles to ensure there was no resulting bias in the topics from removing these articles from our dataset. Our dataset had 197,572 articles after cleaning.

3.3 Model Details

3.3.1 Pipeline Architecture

Our baseline and two proposed pipeline architectures are:

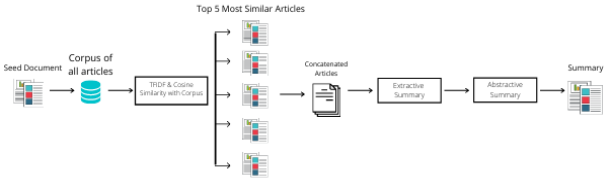

Figure 1: Baseline Architecture

Our baseline pipeline architecture (**Figure 1**) starts with an initial document and finishes with a final summary consisting of the top 5 most similar articles. We begin by performing TF-IDF and a subsequent cosine similarity between the seed document and the entire corpus. We then use the top 5 most similar documents (by cosine similarity scores) to our seed document. Next, the pipeline concatenates these 5 documents together and uses our extractive summarizer followed by our abstractive summarizer. We used an extractive summarizer first to shorten our concatenated document and retain the important facts without the risk of hallucinating. The pipeline then returns our final summary which is evaluated using a ROUGE scoring system.

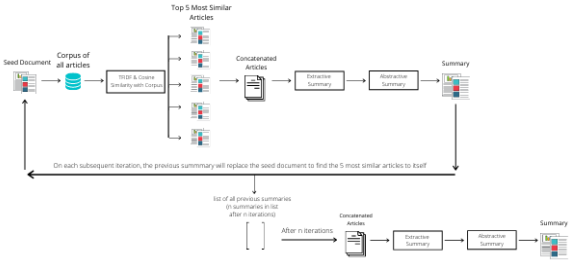

Figure 2: Iterative Stacked Architecture

Our iterative stacked pipeline architecture **(Figure 2)** utilizes the baseline and naively builds on top of it through multiple iterations. After each iteration of the baseline architecture, our model feeds the resulting summary back into

the front of the model to find 5 new similar documents. It takes each summary and feeds it back in as the seed document. At the same time these summaries are saved during each iteration and concatenating them together. Finally, the model performs a final summarization on the concatenated summaries. This architecture yields a more general contextual summary of events by abstracting away the bias from the first 5 articles by using 5 * *n*articles on the same topic.

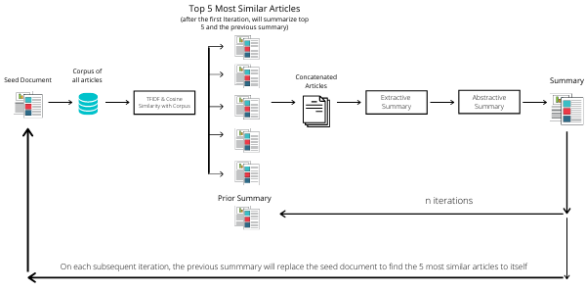

Figure 3: Iterative Recurrent Architecture

Our iterative recurrent architecture **(Figure 3)** applies logic from recurrent neural networks. After each iteration the resulting summary is used to find 5 similar articles and on top of this is also concatenated alongside these articles. This ensures that information from the previous iteration is captured in all future iterations.

3.3.3 Rationale For Pipeline Modelling

Previous NLP work on summarization has pointed to a few flaws in the model-based approaches. Namely, in the extractive modeling process as outlined by Verma (2018), the coherence and fluency of the summaries are sacrificed in exchange for high accuracy when compared to the reference article. In prior abstractive modeling approaches by Nallapati (2016), Rush (2015), and Gunel (2020), abstractive models continued to place a greater emphasis on coherency and syntax while failing to respect facts included in the source. In fact, Krycinski (2019) finds that up to 30% of abstractive models present inconsistencies between the "source" and the summary. Additionally, abstractive models seemed to exhibit weaknesses in producing coherent summaries when the source documents were long (a problem we hoped to mitigate through the use of only first paragraph text).

As outlined in Hsu (2018), a unified modeling for extractive and abstractive summarization helps to provide more informative and readable summarizations on the CNN/Daily Mail dataset with solid human evaluation. We believe that building off of this framework, our proposed iterative unified modeling approaches as presented in both **Figure 2** and **Figure 3** allow for similarly informative and readable summarizations when evaluated through both machine-based metrics and human evaluation.

3.4 Similarity and Evaluation Metrics

3.4.1 TF-IDF and Cosine Similarity

Throughout our project, we used a TF-IDF approach combined with cosine similarity to find similarity scores of our news articles. By multiplying the word frequency in the document by the inverse document frequency of the word across a set of documents, TF-IDF tells you how relevant a word is in a document with respect to your entire collection of documents. We then computed the cosine similarity between documents by representing documents as a set of vectors in vector space. Each term has its own axis and using the formula below we can find out the similarity score between any two documents.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}},$$

3.4.2 ROUGE Evaluation Metric Score

We used ROUGE scoring to help validate our summaries. ROUGE is a commonly used summarization scoring tool that compares machine-generated summaries reference summaries. ROUGE scores precision and recall and comes in a few different forms. We mainly used ROUGE-1, which focuses on unigrams, and ROUGE-L, which focuses on the longest common subsequence since this gives us a mixture of both individual word overlap and sentence structure. One main issue with summary validation is that oftentimes we do not have the reference summary readily available. We got around this issue by fine-tuning our BERT algorithm on news articles with data that did include reference summaries. During the testing phase of the project, in which we summarized multiple similar articles, we used concatenated articles as the reference.

4 Results and Discussion

4.1 Result Variation

Our results varied widely across different model architectures; however, our most successful results were produced by our Recurrent Pipeline Model. At a high level, we found that the Stacked and Baseline Models performed best when applied to a single iteration. The pitfall of these earlier models is due to their inability to persist long range information across multiple iterations. Consequently, each iteration of the Baseline and Stacked Pipelines resulted in summaries that contained little information from the articles of the previous iterations and diverged significantly from the context of the seed document. In contrast, the Recurrent Pipeline was able to persist long range information across iterations rather well.

4.2 Iterative Model Semantic Correction

In addition to mitigating information loss at each iteration, we found evidence to suggest that the Recurrent Model was able to correct semantic errors, coherency, and minor hallucinations in some cases. By the end of our experimentation, we found that the primary limitation of the Recurrent Model is BERT's underlying maximum token length of 512. This is a limiting factor when trying to combine more than three articles per iteration, with greater than five iterations, although this does depend somewhat on the length of the articles being summarized.

4.3 Manual Model Evaluation

With regard to summary coherence, syntactic structure, named entity recognition, and chronological information selection, all models performed satisfactorily and rarely produced content that was completely incoherent. Overall, with regard to task accomplishment, our Recurrent Pipeline, as well as our more simplistic models, demonstrated that they were capable of producing a generalized overview of a topic pertaining to the content of a given seed document. Albeit infrequent, we also found evidence during various trial runs, to suggest that the Recurrent Pipeline architecture can induce summary error-correction over multiple iterations and actually create a more informative summary, while maintaining a reasonably compact length (less than 512 tokens).

4.4 - Machine-Based Metric Evaluation

Model evaluation proved to be one of our most difficult challenges. For each model we capture

the BLEU, ROUGE, and Saturation scores at each iteration. BLEU and ROUGE scoring is helpful for acquiring a frame of reference, although these standard scoring metrics are not particularly insightful since our models are primarily tested on an unlabeled dataset, where there is no reference summary to compare to. Despite not having a labeled reference summary, we tested our models on the first paragraph of each article to serve as a proxy reference. Thus, the Bleu and Rouge scores are still valuable for identifying significant contextual divergences from the seed document. Since we could not rely on these metrics, we resorted to manual inspection and evaluation of our Saturation Score. In order to determine how much new information was injected into the pipeline at each iteration, we created a new metric which we refer to as a Saturation Score. The saturation score is the number of new documents that are present at each iteration. When the saturation is zero, this indicates that no new documents were introduced for summarization between the previous and current iterations, at which point the pipeline is terminated since there exist no unseen articles in the dataset that are more similar to the current summary than the selected articles that comprise it.

4.5 Illustrative Example

While seems to be an abstract concept, below is an illustrative example of our model outputs and the respective scoring using our stated metrics.

Our stacked and recurrent model is an improvement on the baseline. While the pipelines still suffer from problems in coherency and the problem of conflicting factual evidence, the recurrent pipeline model ended up correcting some factual errors in the summary.

Seed Document	"Los Angeles County could see 'catastrophic suffering and death' in the coming weeks, public health officials warn, as the nation's most populous county reported another record day of new coronavirus cases."
Model & Metrics	**Resulting Summary**
Baseline *Rouge-2:* 0.307 *BLEU:* 2.098 *Saturation:* 0.666	"los angeles county is one of the hardest - hit areas in the u. s. the number of people with the coronavirus in the united states has passed 300, 000 on monday. the total number of coronavirus cases reached a quarter - million on monday, less than four weeks after the nation's death toll reached 24 million."
Iterative Stacked *Rouge-2:* 0.305 *BLEU:* 1.917 *Saturation:* 1	"los angeles county is one of the hardest - hit areas in the u. s. the number of people with the coronavirus in the united states has passed 300, 000 on monday. the total number of coronavirus cases reached a quarter - million on monday, less than four weeks after the nation's death toll reached 24 million."
Iterative Recurrent *Rouge-2:* 0.306 *BLEU:* 1.985 *Saturation:* 0.666	"los angeles county is one of the hardest - hit areas in the u. s. the number of people with the coronavirus in the united states has passed 300, 000 on monday. as the total number of coronavirus cases reaches 24 million on monday, as the number continues to rise."

5 Conclusion

This paper explores inverse hierarchical MDS that gives a contextual summary on events. To create the most factually accurate and human readable contextual summaries, we proposed two variations from a baseline pipeline architecture to accomplish this task. We propose two iterative pipeline architectures to find semantically similar documents from a corpus of *NYT* articles. Our iterative recurrent pipeline performed the best in terms of traditional metrics such as ROUGE and BLEU scores. Additionally, the iterative recurrent model allowed for the greatest factual accuracy and human readability from our 3 proposals.

Acknowledgements

We would like to acknowledge our professor Peter Grabowski for teaching us natural language processing. Additionally, we would like to acknowledge the use of the extractive BERT Summarizer as presented by Miller (2019). We would also like to acknowledge Patrick von Platen for his Seq-2-Seq BERT-based model which was used in our pipeline architecture.

References

Christensen et al. 2014, “Hierarchical Summarization: Scaling Up Multi-Document Summarization” June, 2014. in *Association for Computational Linguistics p.902-912*

Fabbri, Alexander R., et al. 2020. “SummEval: Re-evaluating Summarization Evaluation” arXiv:2007.12626 [cs.CL]

Fang, Irving, 1991. “Writing Style Differences in Newspaper, Radio, and Television News.” *Center for Interdisciplinary Studies in Writing, University of Minnesota*

Gunel, Beliz, et. al. 2020 “Mind The Facts: Knowledge-Boosted Coherent Abstractive Text Summarization.” arXiv:2006.15435 [cs.CL]

Gunes Erkan and Dragomir R Radev. 2004. “Lexrank: Graph-based lexical centrality as salience in text summarization.” in *Journal of artificial intelligence research, 22:457–479*.

Hermann, Karl M. et al. 2015. “Teaching Machines to Read and Comprehend” arXiv:1506.03340 [cs.CL]

Hsu, Wan-Ting, et. al. 2018. “A Unified Model for Extractive and Abstractive Summarization using Inconsistency Loss” arXiv:1805.0626 [cs.CL]

Krycinski, W. et al. 2019. “Evaluating the Factual Consistency of Abstractive Text Summarization” arXiv:1910.12840 [cs.CL]

Liu, Yang and Mirella Lapata. 2019. “Text Summarization and Pretrained Encoders” arXiv:1908.08345 [cs.CL]

Ma, CongBo, et al. 2020, “Multi-document Summarization via Deep Learning Techniques: A Survey” arXiv:201104843 [cs.CL]

Miller, Derek (2019). “Leveraging BERT for Extractive Text Summarization on Lectures” arXiv:1906.04165 [cs.CL]

Nallapati, Ramesh, et al. 2016. “Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond.” arXiv:1602.06023 [cs.CL]

Rush, Alexander M, et al. 2015. “A Neural Attention Model for Abstractive Sentence Summarization.” arXiv:1509.00685 [cs.CL]

Verma, Sukriti and Vagisha Nidhi. 2018. “Extractive Summarization using Deep Learning.” *Res. Comput. Sci.* 147 (2018): 107-117.

Wang et. al (2019) “A Text Abstraction Summary Model: Model Based on BERT Word Embedding and Reinforcement Learning.” 4 November 2019 in *MDPI*.

Yang, et. al. (2019) “Hierarchical Summarization Of Text Documents” July 2019 in *Association for Computational Linguistics* DOI:10.18653/v1/P19-1500.

Appendix - Figures and Diagrams

FIGURE 1: Model 1 - Baseline Architecture

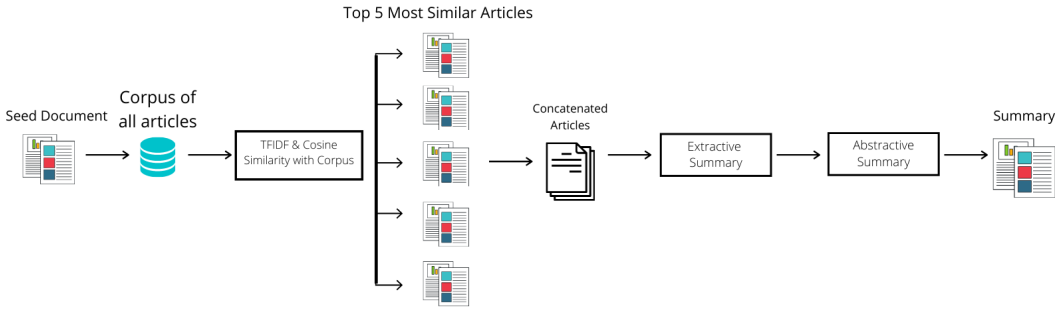

FIGURE 2: Model 2 - Iterative Stacked Architecture

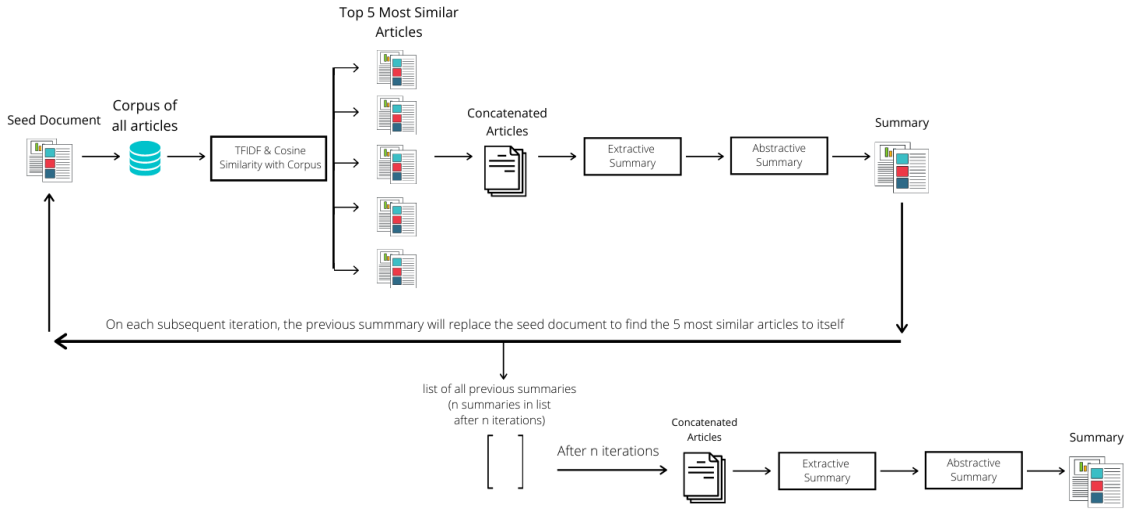

FIGURE 3: Model 3 - Iterative Recurrent Architecture

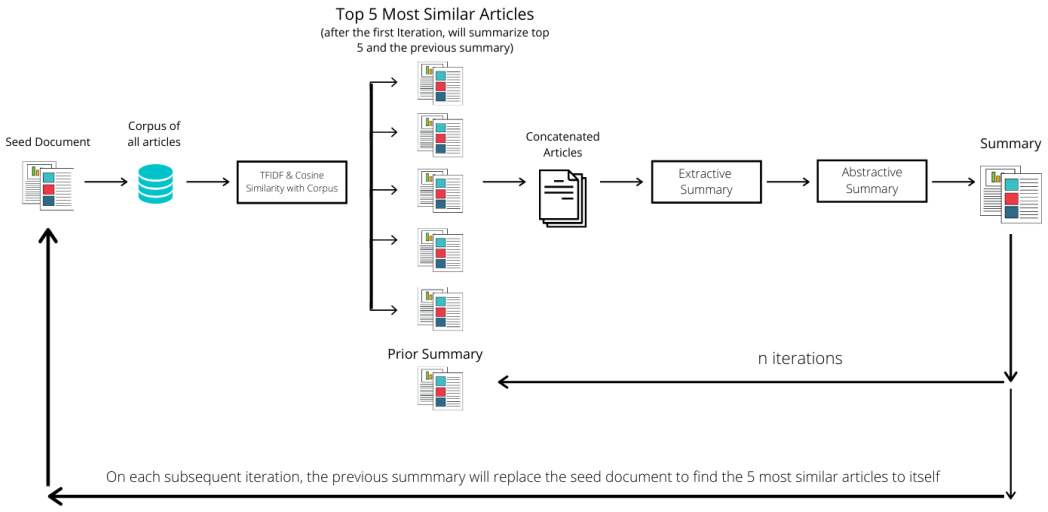

FIGURE 4: ILLUSTRATIVE EXAMPLE

Seed Document	"Los Angeles County could see 'catastrophic suffering and death' in the coming weeks, public health officials warn, as the nation's most populous county reported another record day of new coronavirus cases."
Model & Metrics	**Resulting Summary**
Baseline *Rouge-2:* 0.307 *BLEU:* 2.098 *Saturation:* 0.666	"los angeles county is one of the hardest - hit areas in the u. s. the number of people with the coronavirus in the united states has passed 300, 000 on monday. the total number of coronavirus cases reached a quarter - million on monday, less than four weeks after the nation's death toll reached 24 million."
Iterative Stacked *Rouge-2:* 0.305 *BLEU:* 1.917 *Saturation:* 1	"los angeles county is one of the hardest - hit areas in the u. s. the number of people with the coronavirus in the united states has passed 300, 000 on monday. the total number of coronavirus cases reached a quarter - million on monday, less than four weeks after the nation's death toll reached 24 million."
Iterative Recurrent *Rouge-2:* 0.306 *BLEU:* 1.985 *Saturation:* 0.666	"los angeles county is one of the hardest - hit areas in the u. s. the number of people with the coronavirus in the united states has passed 300, 000 on monday. as the total number of coronavirus cases reaches 24 million on monday, as the number continues to rise."