CS230

A Comparison of Neural and Unsupervised Text Summarization Techniques

Devin Cintron*
Department of Computer Science
Stanford University
dcintron@stanford.edu

Abstract

There is an ever-growing abundance and long form documents available on the web. The ability to synthesize subsections of large volumes of texts into a concise, summarative format will enable experts and novices alike to quickly review large amounts of information in a reasonable time. In this vein, a variety of text summarization techniques have been advanced which effectively reduce long form texts into much shorter formats. The nature of the outputs of these models, however, is notably diverse.

We carry out an assessment of the differences between the summaries produced by two different techniques: the unsupervised *TextRank* algorithm and a neural Pointer-Generator Network. We evaluate the performance of these methods using the commonly used ROUGE score as well as through a surveying of human preferences. Interestingly, we find that, while the Pointer-Generator Network performs better as measured by ROUGE score (average ROUGE-1 F-score of 0.44 vs 0.35), that human evaluation found TextRank summaries to be superior.

1 Introduction

The number of pages on the web surpassed a count of 1 billion as early as 2014. As emerging markets continue to gain web access and as IoT devices continue to increase time spent on the web, growth will surely continue. Due to its obvious value, text summarization has been a long standing and continuous focus of much of Natural Language Processing research.

Modern summarization techniques can be considered in two particular classes: extractive and abstractive techniques. Extractive summarization techniques follow a process in which the most valuable elements of a portion of text are selected and *extracted* to form a summary shorter than the original portion. The *TextRank* algorithm as well as a number of many other lightweight unsupervised techniques have been created for extractive text summarization. Abstractive text summarization, on the other hand, is a more complex procedure in which new language and terms can be introduced which are not drawn directly from the text itself.

On the surface, abstractive text summarization is arguably much closer to human performed summarization. This assertion is drawn from the fact that it can bring out of vocabulary terms whereas extractive techniques cannot. Naturally, the more complex task of abstractive summarization has constituted the majority of recent research into the task, though extractive techniques continue to be visited for their robustness and effectiveness in particular applications.

*Use footnote for providing further information about author (webpage, alternative address)—*not* for acknowledging funding agencies.

CS230: Deep Learning, Winter 2018, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

2 Related work

Because text summarization has been a longstanding focus for natural language processing, there is an extensive existing body of research. Formal techniques for automatic summarization were advanced as early as the 1950s. In (P. Luhn, 1958), for instance, an early technique is outlined for summarizing technical papers and magazine articles based off of a *relevance scoring* derived from term frequency. The resulting summaries which Luhn ironically calls "auto-abstracts", can clearly be thought of as an early form of an extractive summarization technique.

2.1 Extractive Summarization

Modern extractive techniques have advanced quite far beyond those early works of the mid 20^{th} century. In (García-Hernández and Ledeneva, 2013), for instance, a 'genetic' reinforcement algorithm is advanced which attempts to extract elements of a long form text in a manner close to human summarization. Particularly nuanced extractive techniques have been advanced for highly specific applications; in (Bokaei et al., 2016), a method is outlined for the purpose of multi-party summarization (i.e. summarizing the content of two or more different parties in a conversation) that combines the two tasks of speaker resolution and extractive summarization. The actual task of extraction itself has also been approached by a variety of methods, and neural approaches to extraction have been visited. Such an approach is found in (Xu and Durrett, 2019) in which a bidirectional LSTM is used to encode sentences and a CNN is used to encode candidate compressions of those sentences which are then kept or discarded by evaluation of a binary classifier. A similar approach is taken in (Nallapati et al., 2016) which introduces the SummaRuNNer architecture in which a two-layer GRU-RNN is used to determine whether portions of texts should be excluded from a source text while summarizing. Even an approach as complex as this still qualifies as extractive, as the text which remains is still drawn from the original source.

2.2 Abstractive Summarization

Abstractive summarization techniques do vary significantly in implementation details and performance, but the majority are built upon sequence-to-sequence models. As discussed in (Shi et al., 2018), the differences between models, on the whole, fall under the categories of "network structure, parameter inference, and decoding/generation." A non-RNN structure which is often used is a graph-based architecture, as in (Bhargava et al., 2016). In this approach, the long form text is represented in a directed graph where multiple occurrences of the same word are mapped to one vertex to reduce redundancies. Then, sentiment scores are incorporated such that sentences may be fused together with the goal of maintaining the same sentiment is used as a proxy for correctness. As for parameter inference, an interesting work is (Sahoo et al., 2018) which infers significance and salience through a clustering based approach. Via the Markov clustering principle, sentences are grouped by close semantic relationship and new compressed sentences are synthesized via a linguistically-detailed combination of multiple sentences that share very close meaning. Finally, more flexible decoding procedures exist so that models may be better fir to the various preferences of different summarization scenarios. One flexible model is (Fan et al., 2017) which permits users to specify attributes like "desired length, style, [and] the entities that the user might be interested in" in order to create more useful summaries.

3 Dataset

We use the *CNN/ Daily Mail* data set in order to compare these methods. The data set consists of articles drawn from the two news services with accompanying summaries for each article. The average length of the articles is approximately 800 words and the average length of the summaries is approximately 60 words. In order to compare against a state-of-the-art model, we compare the results of the Pointer-Generator model taken from the highest achieving parameters found by the author. This leaves us with a subset of 11,490 articles and summaries drawn from the larger total CNN/Daily Mail.

4 Methods

4.1 Extractive Baseline First-n

As a baseline for comparison of the two other models, we first design a naive "summarization" algorithm which simply takes the first n sentences of an article, where n is a random choice among integers such that the expected value of the length of the summary is consistent with our other extractive approach.

4.2 Extractive Approach: TextRank

In (Mihalcea and Tarau, 2004), the authors define *TextRank*, a graph-based extractive summarization approach adapted from Larry Page's *PageRank* algorithm. A directed graph is created from a longform text in which summaritive text units represent vertices and edges represent relationships between text elements. Thereafter a ranking algorithm is applied and the best scoring elements are kept. This algorithm is similar to PageRank as scores are reflective of the number of incoming connections and the scores of the source-verticies of those incoming connections, the difference being that, in TextRank, edges values are weighted on a basis of the strength of the relationship. The full process of TextRank is then:

1. *Identify text units that best define the task at hand, and add them as vertices in the graph.*
2. *Identify relations that connect such text units, and use these relations to draw edges between vertices in the graph. Edges can be directed or undirected, weighted or unweighted.*
3. *Iterate the graph-based ranking algorithm until convergence.*
4. *Sort vertices based on their final score. Use the values attached to each vertex for ranking/selection decisions.*

Our implementation uses a variation of this scoring function found in (Barrios et al., 2016) and taken from the (GenSim) python library.

4.3 Abstractive Approach: Pointer Generator with Coverage

For our abstractive summarization example, we chose the pointer-generator network approach as advanced by (See et al., 2017), in particular the authors' pointer-generator with coverage technique. Token-wise encoding is performed by a single layer bidirectional LSTM and decoding by a unidirectional LSTM. Bahdanau attention is used to compute an attention distribution and context vector. A probability for *generating* a new term is calculated at each time step via a sigmoid of the weighted product of the context vector, decoded state, and decoded input, as well as a bias. Coverage is an additional component which functions to prevent any particular section of the document from receiving an imbalanced amount of attention – it is the ongoing sum attention distributions of all previous time steps. The coverage value is then included in future calculations of attention to inform about the past distributions. In our training, we were unable to reach a performance exceeding that of the authors' so we opt to use their test results directly when comparing against the output of our extractive and naive approaches.

5 Results

5.1 Automatically-Produced Results

	Average Rouge F-Scores		
	Rouge-1	Rouge-2	Rouge-L
PG-Cov	0.4367	0.2039	0.4098
GenSim	0.3506	0.1461	0.2749
Naïve	0.39816	0.17351	0.35995

For an automated approach to comparison, we used the standard ROUGE score. Specifically we use ROUGE-1, ROUGE-2, and ROUGE-L metrics – these metrics are calculated via overlap of unigrams, overlap of bigrams, and longest common subsequence, respectively. The pointer generator with coverage approach outperforms the TextRank approach by all measures.

Our calculation of ROUGE is performed via National School of Computer Science and Applied Mathematics of Grenoble PhD student Paul Tardy's File2Rouge implementation. We believe that this implementation may be somewhat inflationary as we note that the scores appear to be inflated relative to other ROUGE implementation measurements – however, since our objective is comparison between the abstractive and extractive techniques, we find this to be permissable for our application.

5.2 Human-Produced Results

In order to provide a human measurement of evaluation, we surveyed 22 individuals on their preferences between the methods. Each individual was asked to read a selection of 3 articles, then to select which summary they preferred between that computed by the TextRank approach and the Pointer Generator approach. The order of the summaries was randomized when read to decrease the effect of any sequential bias.

Interestingly, the TextRank approach was rated as more preferred in 43 of the 66 assessments.

5.3 Discussion

The greater performance of the Pointer Generator approach by comparison of ROUGE measure is consistent with our expectations. We were surprised, however, to discover this mismatch between human evaluations and ROUGE scoring. This problem has been explicitly addressed in prior research – in (Liu and Liu, 2008), in fact, the authors find that the correlation between human evaluation and ROUGE scores is rather low. As has been suggested previously, we argue that an efficient and low cost pipeline for human evaluation – such as that provided by MTurk – may be a more valuable metric for evaluation

6 Conclusion/Future Work

We carry out a comparison of an abstractive text summarization technique – the pointer generator model – and an extractive, unsupervised technique – the TextRank approach. We compare the models on a subsection of the CNN/Daily Mail data set. Interestingly, we find that, while evaluation via ROUGE scoring prefers the pointer generator approach, human evaluation scores find TextRank to provide more preferred summary.

Valuable future work would include an assessment of a wider range of models for summarization. Additionally, it would be valuable to perform a greater investigation into comparisons of human evaluation versus ROUGE scoring across a diverse range of summarization techniques and contexts. While research comparing ROUGE and human scoring has been performed previously, it is important to compare these methods across different summarization contexts and methods as there is a great diversity in the nature of summaries depending upon the application.

7 Contributions

The entirety of the work within this project, apart from explicit citations, was performed alone by Devin Cintron.

References

Federico Barrios, Federico López, Luis Argerich, and Rosa Wachenchauzer. 2016. Variations of the similarity function of textrank for automated summarization. *CoRR*, abs/1602.03606.

Rupal Bhargava, Yashvardhan Sharma, and Gargi Sharma. 2016. Atssi: Abstractive text summarization using sentiment infusion. *Procedia Computer Science*, 89:404 – 411. Twelfth International Conference on Communication Networks, ICCN 2016, August 19â“ 21, 2016, Bangalore, India Twelfth International Conference on Data Mining and Warehousing, ICDMW 2016, August 19-21, 2016, Bangalore, India Twelfth International Conference on Image and Signal Processing, ICISP 2016, August 19-21, 2016, Bangalore, India.

Mohammad Hadi Bokaei, Hossein Sameti, and Yang Liu. 2016. Extractive summarization of multi-party meetings through discourse segmentation. *Natural Language Engineering*, 22:41–72.

Angela Fan, David Grangier, and Michael Auli. 2017. Controllable abstractive summarization. *CoRR*, abs/1711.05217.

René Arnulfo García-Hernández and Yulia Ledeneva. 2013. Single extractive text summarization based on a genetic algorithm. In *Pattern Recognition*, pages 374–383, Berlin, Heidelberg. Springer Berlin Heidelberg.

GenSim. Python library.

Feifan Liu and Yang Liu. 2008. Correlation between rouge and human evaluation of extractive meeting summaries. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, HLT-Short '08, pages 201–204, Stroudsburg, PA, USA. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2016. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *AAAI*.

H P. Luhn. 1958. Luhn, h.p.: The automatic creation of literature abstracts. ibm journal of research and development 2(2), 157-165. *IBM Journal of Research and Development*, 2:159 – 165.

Deepak Sahoo, Ashutosh Bhoi, and Rakesh Chandra Balabantaray. 2018. Hybrid approach to abstractive summarization. *Procedia Computer Science*, 132:1228 – 1237. International Conference on Computational Intelligence and Data Science.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *ACL*.

Tian Shi, Yaser Keneshloo, Naren Ramakrishnan, and Chandan K. Reddy. 2018. Neural abstractive text summarization with sequence-to-sequence models. *CoRR*, abs/1812.02303.

Jiacheng Xu and Greg Durrett. 2019. Neural extractive text summarization with syntactic compression. *CoRR*, abs/1902.00863.