Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), pages 5388–5392
Marseille, 20-25 June 2022

Surfer100: Generating Surveys From Web Resources, Wikipedia-style

Irene Li, Alexander Fabbri, Rina Kawamura, Yixin Liu, Xiangru Tang, Jaesung Tae, Chang Shen, Sally Ma, Tomoe Mizutani, Dragomir Radev
Yale University
{irene.li,alexander.fabbri,rina.kawamura,yixin.liu,xiangru.tang,jake.tae}@yale.edu
{chang.shen,sally.ma,tomoe.mizutani,dragomir.radev}@yale.edu

Abstract

Fast-developing fields such as Artificial Intelligence (AI) often outpace the efforts of encyclopedic sources such as Wikipedia, which either do not completely cover recently-introduced topics or lack such content entirely. As a result, methods for automatically producing content are valuable tools to address this information overload. We show that recent advances in pretrained language modeling can be combined for a two-stage extractive and abstractive approach for Wikipedia lead paragraph generation. We extend this approach to generate longer Wikipedia-style summaries with sections and examine how such methods struggle in this application through detailed studies with 100 reference human-collected surveys. This is the first study on utilizing web resources for long Wikipedia-style summaries to the best of our knowledge.

Keywords: web resources, survey generation, abstractive summarization

1. Introduction

Novel concepts are being introduced and evolving at a rate that makes creating high-quality, up-to-date Wikipedia pages for such topics challenging. A pipeline for automatically creating such Wikipedia pages is thus desirable. While there has been some work on generating full Wikipedia pages, these efforts are either domain-specific (Sauper and Barzilay, 2009), making strong assumptions about the topics being summarized (Banerjee and Mitra, 2016), or are purely extractive (Jha et al., 2015). In a related line of work, query-based summarization has been applied to specific sections of Wikipedia pages (Deutsch and Roth, 2019; Zhu et al., 2019), which can be viewed as a more self-contained version of Wikipedia page generation. Recent Wikipedia page generation work has focused on generating the initial leading paragraph of a Wikipedia page (Liu et al., 2018; Liu and Lapata, 2019; Perez-Beltrachini et al., 2019). These papers consist of a two-step framework by which an extractive method selects relevant content for a specific topic, and an abstractive method generates the final summary of the topic.

In this paper, we first examine how recently-introduced pretrained language models (Devlin et al., 2019; Liu et al., 2019; Lewis et al., 2019) improve upon both the extractive and abstractive steps of previous models for the task of lead paragraph generation. We further focus on analyzing the extension of such methods to full Wikipedia page generation on scientific topics related to AI and Natural Language Processing (NLP). We manually create summaries of 100 AI and NLP topics divided along sections, as on Wikipedia pages. We perform ablation studies on content selection and generation methods over selected topics, finding that current content selection methods are not precise and fail to differentiate content well among queries for subtopics of the main topic.

Our contributions are: 1) We demonstrate how recent advances in pretrained language models improve upon Wikipedia lead paragraph generation. 2) We then extend such a method to generate full Wikipedia-style pages of scientific topics; 3) For a testing purpose, we manually collected Surfer100, 100 SURveys From wEb Resources on scientific topics, filling the gap on human-written surveys using web resources in scientific topics. We provide a better understanding of current methods and their faults on a real-world application.

2. Wikipedia Lead Paragraph Generation

In this section, we show how combining recent methods for a two-staged approach of content selection and generation give improved results on the WikiSum dataset (Liu et al., 2018) as well as a newly curated set of Wikipedia articles.

2.1. Data

We make use of the **WikiSum** dataset (Liu et al., 2018), a collection of over 1.5 million Wikipedia pages and their references. Applying pretraining techniques such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and BART (Lewis et al., 2019); however, these models make use of Wikipedia during pretraining. To address this problem, we mirror the process of Liu et al. (2018) to collect an unbiased dataset of newly added Wikipedia pages[1] which did not appear in pretraining, (**NewPage WikiSum**). We collect 10,000 of the newest Wikipedia pages, scrape Wikipedia for their references and the top 10 Google Search results. We remove non-English results and any articles for which we could not scrape a single reference. Due to the sparsity of search results on specific topics, we were left with about 1,000

[1] https://en.wikipedia.org/wiki/Special:NewPages

Methods	L=5	L=10	L=20	L=40
TF-IDF	24.86	32.43	40.87	49.49
LSTM-Rank	39.38	46.74	53.84	60.42
WikiCite	**65.27**	69.77	73.54	76.51
Semantic Search	34.87	48.60	61.87	74.54
RoBERTa-Rank	64.12	**72.49**	**79.17**	**84.28**

Table 1: ROUGE-L-Recall scores for WikiSum content selection, varying the number of paragraphs returned.

articles we used as a test set. We name this dataset **NewPage**.

2.2. Step One: Content Selection

We experiment with five approaches for our initial content-selection step. **TF-IDF**: a simple approach to extract relevant content is to use term frequency–inverse document frequency (Liu et al., 2018; Fan et al., 2019). **LSTM-Rank**: Liu and Lapata (2019) approach query-based content selection as a regression problem of predicting the ROUGE-2 recall of a given paragraph-topic pair. **WikiCite**: Deutsch and Roth (2019) approach query-based summarization via an extractive classification approach with attention (Bahdanau et al., 2014) over the topic and context.

We apply two additional methods to the task of content selection. **Semantic Search**: Reimers and Gurevych (2019) fine-tune BERT and Roberta using siamese and triplet networks to produce fixed-length vectors which can be compared using cosine similarity to find semantically similar input. We embed the title of each Wikipedia page, and each candidate paragraph, using this method, and choose the paragraphs with the most similar vectors to the title as selected content. **RoBERTa-Rank**: we train RoBERTa similar to the approach of (Liu and Lapata, 2019), treating the title and paragraph to be ranked as sentence pairs and use predicted relevance scores as a ranking function for determining the most relevant paragraphs. We show the results in Table 1. WikiCite performs well despite not including extensive pretraining and without fine-tuning on the WikiSum data, perhaps because the model is trained for the task of fine-grained selection (for section titles within a given page). RoBERTa-Rank is the highest-scoring content selector except for the 5-paragraph case, so then we choose this as the content selection method for abstractive summarization input on WikiSum data.

2.3. Step Two: Abstractive Summarization

We use the RoBERTa-Rank content selection component to select paragraphs up to 1,024 total tokens as input to our abstractive summarization step. As the abstractive model in our two-step approach, we experiment with **BART** (Lewis et al., 2019), which has achieved state-of-the-art performance in both natural language understanding and generation tasks. We compare BART fine-tuned on the WikiSum data with the previous state of the art **HierSumm** model (Liu and Lapata, 2019).

Dataset	Hiersumm	BART
WikiSum	41.53/26.52/35.76	**46.61/26.82/43.25**
NewPage	31.64/15.06/27.13	**39.29/18.56/36.03**

Table 2: ROUGE scores for intro paragraph generation on WikiSum and NewPage WikiSum.

We show improved results on generating the introduction paragraph on WikiSum and on our NewPage WikiSum data in Table 2. We use the same RoBERTa-Rank for both models on NewPage WikiSum. BART generation still outperforms HierSumm. We note that the large difference in scores between that of the WikiSum data and on our collected subset is likely due to the widespread nature of topics in WikiSum; WikiSum includes many well-established topics for which finding reference documents is simple, while the newly introduced topics may not contain enough reference information for higher-quality generation. So far, we have shown that applying RoBERTa-Rank and BART as a two-step pipeline gives promising results in generating lead Wikipedia sections.

3. Application of Pipeline to Full Wikipedia Generation

We follow Banerjee and Mitra (2016) in extending a two-step pipeline to full Wikipedia-style summaries (section by section content selection and summarization) to study the applicability of recent methods in this real-world setting.

3.1. Data

Testing our models on full Wikipedia-page data would again face the problem of pretraining bias, and large-scale collection of full-size Wikipedia pages for novel topics is not infeasible. Furthermore, we focus on generating Wikipedia-style pages for AI-related topics. We picked a mixture of NLP and broader AI-related topics to include topics with existing Wikipedia pages as well as those without pages or stub articles, with 100 topics in total.

We define a template for the surveys consisting of five sections: **Introduction**, **History**, **Key Ideas**, **Variations** (similar topics or topics with similar goals) and **Applications**. We arrived at these section titles by an examination of sample Wikipedia pages in NLP. First, we searched Google for the given topic, retrieving all HTML page links for the first two search result pages. We then have the annotator read each page, extract relevant content into the corresponding section, and paraphrase and summarize the relevant content for each section to between 50 and 150 words per section. We split the job to eight annotators, and each survey requires 45 to 60 minutes. Given that the data collection is time-consuming, we focus on a testing purpose rather than

Methods	k=10	k=20	k=50
SS-BERT	0.5360	0.5370	0.4242
SS-Wiki	0.5050	0.5125	0.4110
SS-SciBERT	0.5780	0.5555	0.4232
WikiCite	**0.7460**	0.6605	0.4722
RoBERTa-Rank	0.7240	**0.6925**	**0.5024**

Table 3: Evaluation on Content Selection: comparison of AvgP@k scores.

training. We make all data public. [2]

3.2. Content Selection

We first tested the quality of the content selection methods for generic retrieval of content relevant to a topic on our data. We choose the Semantic Search, WikiCite, and RoBERTa-Rank methods from Table 1 for analysis. For Semantic Search, we experiment with three types of sentence embeddings, the original sentence-transformer BERT embeddings (**SS-BERT**), embeddings fine-tuned with SciBERT (**SS-SciBERT**), and a version fine-tuned to differentiate whether two paragraphs belong to the same Wikipedia section (**SS-Wiki**). Surprisingly, we found such content was often returned during retrieval despite the poor grammaticality and relevance. We hypothesize that the tendency to return short sentences, often with odd punctuation may relate to the extension of these methods to paragraph levels while inherently being developed for sentence-level tasks.

We then remove sentences shorter than 6 tokenized words, as well as apply heuristics for removing sentences based on the number of parentheses, brackets, and other tokens such as equal signs. We required that each paragraph returned consist of at least two sentences and require that the topic word (or one word within the topic, for multi-word topics) appear in the paragraph. About 85 paragraphs per topic remain after this filtering. The comparison of results before and after preprocessing and filtering is found in Table 3. Notably, the WikiCite method performs much better than semantic search and close to RoBERTa. We believe this is because the method is trained for content selection based on a topic and not simply trained for returning content with high recall. A potential problem with current methods in this two-step approach is that content selection is trained and evaluated with recall in mind, to capture as large a range of the topic, which produces models without the precision necessary in a real-world application. This aligns with previous work in extractive summarization suggesting that optimizing for recall gives suboptimal results (Zopf et al., 2018).

Section-Specific Content Selection: We investigated the ability of our content selection models to retrieve content specific for each chosen section, for example, querying "History of BERT"rather than "BERT."We observed large overlaps between the returned results, between 5 and 9 paragraph overlap between the top 10 results for each section. Among all methods, Wikicite has the least overlap. As an alternative method to select distinct content for each section, we investigate clustering methods, using out-of-the-box Agglomerative (Müllner, 2011) clustering provided by scikit-learn [3]. We cluster the embeddings obtained before the final output layer from the WikiCite and RoBERTa methods, and the Search-Wiki embeddings. We annotated the coherence of each cluster. Clusters obtained using embeddings from RoBERTa, Search-Wiki and WikiCite had a corresponding average coherence of 3.07, 3.40, and 3.52 on a 1-5 scale, signaling slightly above-average coherence for each clustering. Again, the poor performance of RoBERTa in clustering may be due to the more general topic training method. As suggested by Deutsch and Roth (2019), the WikiCite method may dilute topic information in the final layer despite topic attention in previous layers and thus benefit from using embeddings before the final layer as clustering.

Method	R-1	R-2	R-L
WikiCite + First3	31.37	9.74	<u>20.51</u>
WikiCite + First5	32.53	<u>9.96</u>	20.30
WikiCite + TextRank	<u>32.57</u>	9.79	19.03
WikiCite + MMR	29.79	6.66	16.82
RoBERTa-Rank + First3	29.68	7.87	18.93
RoBERTa-Rank + First5	30.64	7.95	18.71
RoBERTa-Rank + TextRank	29.37	7.25	16.81
RoBERTa-Rank + MMR	28.78	4.71	15.33
WikiCite + BART	29.00	6.86	18.57
RoBERTa-Rank + BART	**32.23**	**10.12**	**21.78**

Table 4: Summarization performance: ROUGE scores.

3.3. Abstractive Summarization

Generation Model Choice: To perform study on the choice of generation model, we took the best performing WikiCite and RoBERTa-Rank content selection methods for the introduction paragraph as input to BART. We also compared with classic baselines: FirstK (K=3,5) and TextRank (Mihalcea and Tarau, 2004) and MMR (Goldstein and Carbonell, 1998). We show the ROUGE (Lin, 2004) score on 100 topics in Table 4. One can notice that, among the baselines, WikiCite has better performance among RoBERTa-Rank, marked with underlines. However, in the baselines, none of the summarization methods are robust. The best performance can be found when applying our pipeline (RoBERTa-Rank with BART) and this method surpasses other selected baselines in all cases.

We further conduct human evaluation on WikiCite + BART and RoBERTa-Rank + BART. We randomly select 20 concepts and ask two human judges to give scores (range 1-5) on the following four perspectives:

[2]https://github.com/IreneZihuiLi/Surfer100

[3]https://scikit-learn.org/stable/index.html

Introduction
Text summarization is an interesting machine learning field that is increasingly gaining traction. As research in this area continues , we can expect to see breakthroughs that will assist in fluently and accurately shortening long text documents. In this article, we look at how machine learning can be used to help shorten text.
History
Summarization has been and continues to be a hot research topic in the data science arena. While text summarization algorithms have existed for a while , major advances in natural language processing and deep learning have been made in recent years. Google has reportedly worked on projects that attempt to understand novels. Summarization can help consumers quickly understand what a book is about.
Key Ideas
Automatic summarization aims to produce a shorter version of an input text, preserving only the essential information. There are two main types of summarization : extractive summarization selects important sentences from the input and abstractive summarizing generates content without explicitly re-using whole sentences. In our new paper , we constructed two novel , large-scale summarization datasets from scientific journal articles.
Variations
Multi-document summarization can be a powerful tool to quickly analyze dozens of search results. MeaningCloud 's Summarization API locates the most relevant phrases in a document and builds a synopsis with them. More specific summarization systems could be developed to analyze legal documents.
Applications
Summarization can be a crucial component in the tele-health supply chain when it comes to analyzing medical cases. The Spreading Activation approach does not allow to improve our results. Tables 8 and 9 show the high recall obtained with these methods, which may be a very interesting feature in some cases.

Table 5: Sample survey (part) of the topic `Text Summarization` created using our pipeline.

Introduction
Dropout is a technique where randomly selected neurons are ignored during training. This means that their contribution to the activation of downstream neurons is removed. Dropout alone does not have any way to prevent parameter values from becoming too large during this update phase. In the example below we add a new Dropout layer between the input (or visible layer) and the first hidden layer. The dropout rate is set to 20%, meaning one in 5 inputs will be randomly excluded from each update cycle.
History
Classical generalization theory suggests that to close the gap between train and test performance , we should aim for a simple model. Christopher Bishop formalized this idea when he proved that training with input noise is equivalent to Tikhonov regularization. In 2014, Srivastava et al. developed a clever idea for how to apply Bishop 's idea to the internal layers of the network. They proposed to inject noise into each layer of the Network before calculating the subsequent layer.
Key Ideas
Additionally , as recommended in the original paper on Dropout , a constraint is imposed on the weights for each hidden layer. This is done by setting the kernel˙constraint argument on the Dense class when constructing the layers. In the example below Dropout is applied between the two hidden layers and between the last hidden layer and the output layer.
Variations
With a Gaussian-Dropout , the expected value of the activation remains unchanged. Unlike the regular Dropout , no weight scaling is required during inferencing. Dropout is only used during the training of a model and is not used when evaluating the skill of the model. The main problem hindering dropout in NLP has been that it could not be applied to recurrent connections.
Applications
During training time , dropout randomly sets node values to zero. During inference time, dropout does not kill node values, but all the weights in the layer were multiplied. This multiplier could be placed on the input values rather than the weights. TensorFlow has its own implementation of dropout which only does work during training time.

Table 6: Sample survey of the topic of `Dropout`. Some stylistic problems such as references to examples described in the original document are present, although key concepts of the topic are addressed.

readability, relevancy, redundancy and hallucination. For readability and relevancy, higher score is better; but for redundancy and hallucination, higher score is not preferred, as we want the survey to be less redundant and hallucinate on the content. Results are shown in Table 7. As seen in the Table, RoBERTa+BART performs better in the most cases, which is consistent with the ROUGE evaluation. Both models have a high hallucination score, we hypothesize that the content selection step keeps too many information that should not be included in the leading paragraph, for example, model technical details.

Generation of Full Summaries: We take the clustering output for the three embedding methods in the previous section (**Cluster Search-Wiki**,**Cluster WikiCite**, and **Cluster RoBERTa**) as well as the Search-

Evaluation	WikiCite+BART	RoBERTa+BART
Readability	3.70	**4.15**
Relevancy	3.28	**3.58**
Redundancy*	**1.40**	1.43
Hallucination*	2.65	**2.55**

Table 7: Human evaluation on the leading paragraph generation, average scores on 20 random selected topics. * means lower score is better. RoBERTa is a short form of RoBERTa-Rank.

Wiki retrieval output (**Retrieval Search-Wiki**) as input to our generation component to create full sectioned summaries. We did not conduct similar evaluation as we think the trend would be similar to the evaluation for the leading paragraph. Instead, we show two case studies in Table 5 and 6. In Table 5, we show the generated summary of the topic `text summarization`. We could see there are descriptions about this topic: "Text summarization is an interesting machine learning field ", "Automatic summarization aims to ... ". We find certain stylistic features present in the surveys do not match Wikipedia pages. For example, some content is stated in the first person: "In our new paper, we...". This is an artifact of the generation model and the content extracted and can likely be remedied by fine-tuning BART in a different setting.

4. Conclusion

In this paper we show improvements in individual components of Wikipedia summarization through an application of recently-introduced embedding and summarization techniques, but largely focus on the failures of these methods when extended in a real-world scenario of full-page Wikipedia-styled summarization. We believe that a focus on high-precision and fine-grained query-based summarization in future work will help make this pipeline viable.

5. Bibliographical References

Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural Machine Translation by Jointly Learning to Align and Translate. *arXiv preprint arXiv:1409.0473*.

Banerjee, S. and Mitra, P. (2016). Wikiwrite: Generating Wikipedia Articles Automatically. In *IJCAI*.

Deutsch, D. and Roth, D. (2019). Summary Cloze: A New Task for Content Selection in Topic-Focused Summarization. In *EMNLP*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Uderstanding. *NAACL*.

Fan, A., Jernite, Y., Perez, E., Grangier, D., Weston, J., and Auli, M. (2019). Eli5: Long Form Question Answering. *ACL*.

Goldstein, J. and Carbonell, J. (1998). Summarization: (1) using MMR for diversity- based reranking and (2) evaluating summaries. In *TIPSTER TEXT PROGRAM PHASE III: Proceedings of a Workshop held at Baltimore, Maryland, October 13-15, 1998*, pages 181–195, Baltimore, Maryland, USA, October. Association for Computational Linguistics.

Jha, R., Coke, R., and Radev, D. (2015). Surveyor: A System for Generating Coherent Survey Articles for Scientific Topics. In *Twenty-Ninth AAAI conference on artificial intelligence*.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2019). Bart: Denoising Sequence-to-sequence Pre-training for Natural Language Generation, Translation, and Comprehension. *arXiv preprint arXiv:1910.13461*.

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

Liu, Y. and Lapata, M. (2019). Hierarchical Transformers for Multi-document Summarization. *ACL*.

Liu, P. J., Saleh, M., Pot, E., Goodrich, B., Sepassi, R., Kaiser, L., and Shazeer, N. (2018). Generating Wikipedia by Summarizing Long Sequences. *ICLR*.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A Robustly Optimized Bert Pretraining Approach. *arXiv preprint arXiv:1907.11692*.

Mihalcea, R. and Tarau, P. (2004). TextRank: Bringing order into text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411, Barcelona, Spain, July. Association for Computational Linguistics.

Müllner, D. (2011). Modern hierarchical, agglomerative clustering algorithms. *arXiv preprint arXiv:1109.2378*.

Perez-Beltrachini, L., Liu, Y., and Lapata, M. (2019). Generating Summaries with Topic Templates and Structured Convolutional Decoders. *ACL*.

Reimers, N. and Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *ACL*.

Sauper, C. and Barzilay, R. (2009). Automatically Generating Wikipedia Articles: A Structure-aware Approach. In *ACL 2009*.

Zhu, H., Dong, L., Wei, F., Qin, B., and Liu, T. (2019). Transforming Wikipedia into Augmented Data for Query-Focused Summarization. *arXiv preprint arXiv:1911.03324*.

Zopf, M., Loza Mencía, E., and Fürnkranz, J. (2018). Which scores to predict in sentence regression for text summarization? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1782–1791, New Orleans, Louisiana, June. Association for Computational Linguistics.