Using Pre-Trained Transformer for Better Lay Summarization

Seungwon Kim
Incheon Airport Corporation, Georgia Institute of Technology
skim3222@gatech.edu

Abstract

In this paper, we tack lay summarization tasks, which aim to automatically produce lay summaries for scientific papers, to participate in the first CL-LaySumm 2020 in SDP workshop at EMNLP 2020. We present our approach of using Pre-training with Extracted Gap-sentences for Abstractive Summarization (PEGASUS; Zhang et al., 2019b) to produce the lay summary and combining those with the extractive summarization model using Bidirectional Encoder Representations from Transformers (BERT; Devlin et al., 2019) and readability metrics that measure the readability of the sentence to further improve the quality of the summary. Our model achieves a remarkable performance on ROUGE metrics, demonstrating the produced summary is more readable while it summarizes the main points of the document.

1 Introduction

Recent summarization techniques have greatly benefitted from the advancement of language models and successfully produced plausible summaries for both general news articles in real-life and technical scholarly documents in the expert domain. An informative but concise summary can help people to reduce the search time and boost the decision making by expeditiously providing more relevant documents (Mani et al., 2002; Roussinov and Chen, 2001; Maña-López et al., 2004; McKeown et al., 2005). For processing scholarly documents, automatic summarization is promising since it can benefit researchers to cope with the pace of the exponentially growing number of publications (Bornmann and Mutz, 2015).

Despite the recent advancement in automatic summarization literature, summarization for scholarly documents has been less explored compared to the works regarding summarization for ordinary news articles due to the absence of large-scale datasets. Developing human-written lay summaries for scholarly documents is challenging since it involves expert knowledge to understand the technical jargon and the complex structure of scientific documents. Because of these inherent challenges, existing summarization techniques for scientific documents is limited in a sense, which the produced summary is either too concise to provide important information (Vasilyev et al., 2019; Cachola et al., 2020) or aiming to directly extract the content from abstract or citation sentences (Yasunaga et al., 2019), which mostly resembles the abstract, making it hard for the public and researchers from outside of the particular domain to understand the main points of the scientific papers. Although the readability of the abstracts in scientific papers had continuously decreased due to the increase in the use of technical jargon (Plavén-Sigray et al., 2017), the summarization of scientific papers for the public and researchers from outside of the certain field has been remained elusive.

To provide a better summary for the public and researchers, we participated in the first Computational Linguistics Lay Summary Challenge (CL-LaySumm 2020) Shared task (Chandrasekaran et al., (Forthcoming) and developed a summarization system that automatically produces lay summaries for scholarly documents. The main task of CL-LaySumm is producing a corresponding lay summary given the full-text and abstract of the research paper. We employed the dataset from the CL-LaySumm 2020 committee and performed experiments using recent summarization models including Pre-training with Extracted Gap-sentences for Abstractive Summarization (PEGASUS; Zhang et al., 2019b), extractive summarization with Bidirectional Encoder Representations from Transformers (BERT; Devlin et al., 2019), and a new evaluation protocol that measures the readability of the sentence in the summary. We showcase how PEGASUS, BERT, and readability metric improve the summarization system and demonstrate that

Proceedings of the First Workshop on Scholarly Document Processing, pages 328–335

https://doi.org/10.18653/v1/P17

the produced summary is more readable while it summarizes the main ideas of the documents.

2 Related Work

The type of recent benchmark datasets that are widely utilized for evaluating the performance of the summarization system can be categorized into two themes: news articles and scientific documents.

Summarization of news articles have been more actively explored since it is relatively easy to develop human-written summaries. Woodsend and Lapata (2010) and Cheng and Lapata (2016) created a large-scale dataset that contains 200K news articles with manually written gold summaries. Owing to the large-scale dataset and relatively simple structure of the articles, neural abstractive summarization using sequence models such as Long Term Short Memory (LSTM, Hochreiter and Schmidhuber, 1997) with attention mechanism (Bahdanau et al., 2014) has been actively used in abstractive summarization for news articles. The attention-based encoder-decoder network has been improved by others. See et al. (2017) used LSTM with two different networks: pointer-generator network that produces accurate expression by pointing each word in the source and coverage network that avoids repetition. Paulus et al. (2017) incorporated reinforcement learning (RL) into sequence models for summarization tasks and Celikyilmaz et al. (2018) developed multi-agent encoders that communicate with each other by sharing outputs for each layer in the encoder network.

After the advent of pre-trained language models such as Transformer, BERT, and Bidirectional and Auto-Regressive Transformers (BART), the summarization literature benefits from these pre-trained language models that provide more contextual word representation (Vaswani et al., 2017; Devlin et al., 2019; Lewis et al., 2020). Liu and Lapata (2019) used BERT model as an encoder, Zhang et al. (2019a) applied BERT to both encoder and decoder networks, Scialom et al. (2020) constructed generative adversarial networks using BERT models, and Yoon et al. (2020) appended semantic similarity layers on top of the pre-trained BART. While neural sequence models have been successfully applied to the summarization for news articles, applying the same techniques to scientific documents would be challenging since the scholarly documents are far longer than ordinary news articles and have a complicated structure. Our work is different from the described works as we tackle summarization for scientific documents.

Although summarization for scientific texts is less explored, Cohan et al. (2018) proposed hierarchical encoder-decoder network to address the long scholarly documents for constructing abstract summary, Yasunaga et al. (2019) suggested summarization using abstract and citation sentences with graph convolutional networks (Kipf and Welling, 2016) and LSTM, and released the medium-scale dataset that contains 1000 scientific papers in the computational linguistic domain with human-written summaries and citation sentences for each paper. Cachola et al. (2020) implemented an extreme summarization system, which is TLDR (Too Long; Don't Read) summarization, for scientific documents using multi-task learning with headline generation models (Vasilyev et al., 2019). Zhang et al. (2019b) proposed PEGASUS by masking important sentences in the input document with a Transformer-based encoder-decoder network to force the model to summarize main points of the contents given the remainder of the text. PEGASUS tackled summarization for both news articles and scholarly documents but it only aimed to produce the abstract. In contrast, our work is distinct from the previous approaches as we aim to produce lay summaries for scientific documents rather than generating extremely short sentences or summaries that contain technical words which makes it difficult for lay audiences to understand.

To facilitate scholarly document processing, there have been annual workshops regarding data mining, natural language processing (NLP), information retrieval for scientific publications: BIRNDL (Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries), WOSP (Workshop on Mining Scientific Publications), TAC (Text Analytics Conference). In particular, the annual CL-SciSumm (Jaidka et al., 2016, 2018; Chandrasekaran et al., 2019) encouraged participants to research on scientific documents summarization. Our work is closely related to the CL-LaySumm 2020, which is the first lay summary challenge shared task. We employed the LaySumm dataset provided by the workshop organizing committee and performed experiments using a variety of recent summarization models to develop the lay summarization system.

3 Data Analysis

3.1 Overview

Laysumm dataset consists of around 600 scientific papers in epilepsy, archeology, and materials engineering domain, including full-text, abstract and corresponding lay summaries written by authors and journalists. The task for CL-LaySumm 2020 is creating a lay summary with less than 150 words given the full-text and the abstract of the paper. For evaluation, a test set which contains 37 scientific papers without ground truth lay summary is given. The below table shows the average number of words and sentences for each document. Here Spacy (Honnibal and Johnson, 2015) is used for word tokenization.

-	Train		Test	
	words	sentences	words	sentences
Full-text	4915.31	254.41	5696.57	306.36
Abstract	271.96	13.27	264.28	12.51
Laysum	109.07	3.82	—	—

Table 1: Average word, sentence length of dataset.

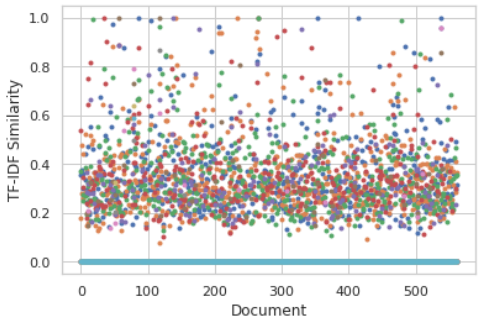
3.2 Sentence similarity

Before developing a specific summarization model, we measured the sentence similarity to determine which type of summarization is suitable for lay summarization. There are two types of summarization: extractive summarization and abstractive summarization. The extractive summarization scores the importance of sentences in the source and directly extracts the sentences based on the score. In contrast, the abstractive summarization generates the summary from scratch while it maintains the representative content of the source. We assumed that this resembles the way humans summarize the contents and the lay summarization can be categorized into the abstractive summarization. However, if the sentences in the lay summary exist in the abstract or full-text of the paper, extractive summarization is more promising. Table 2 shows the average number of overlapping sentences between the sentences in the lay summary and the abstract and full-text for the training set.

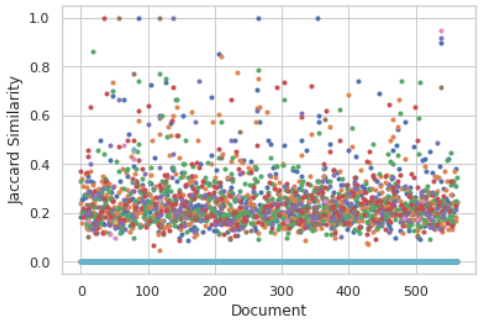
—	# overlapping sentences
Full-text	0.01
Abstract	0.12

Table 2: Average number of overlapping sentences.

As shown in Table 2, the lay summaries were written from scratch rather than directly using the sentences from the abstract or full-text of the paper. We observed that overlapping occurs only 7% of the training set (40 of 572) and if any sentence in the lay summary exists in the abstract, there are 1.73 overlapping sentences in the abstract on average.



(a) TF-IDF similarity



(b) Jaccard similarity

Figure 1: Sentence similarity between the sentences in the lay summary and sentences in the full-text.

We also measured the similarity between the sentences in the lay summary and the sentences in the full-text. For this task, Term Frequency–Inverse Document Frequency (TF-IDF) and Jaccard similarity are used (Sammut and Webb, 2010; Hamers et al., 1989). Figure 1 shows the maximum value of similarity for each sentence in the lay summary in terms of TF-IDF and Jaccard similarity. As shown in the figures, the similarity is below 0.4 on average for TF-IDF and it becomes lower for Jaccard similarity. From the results of the analysis, we excluded full-text and aimed to produce the lay summary solely with the abstract.

4 Method

There are two main summarization models used in our system to generate the lay summary. We tried to use PEGASUS which is an abstractive summarization model and Presumm (Liu and Lapata, 2019) for extractive summarization to produce the summary. We trained the summarization model on the

lay summary dataset in a supervised way by pairing the abstract and the corresponding lay summary of the paper. After producing the lay summary using PEGASUS, we improved the quality of the produced summary by appending important sentences to the summary of which the number of words is under a certain threshold. For example, if the number of the lay summary generated by the abstractive model was under 90, we added the sentences from the corresponding summry geneated by extractive model up to this threshold. When appending the sentences to the produced summary, we prioritized the sentences in the abstract based on the score predicted by Presumm model and readability metric and applied Tri-gram blocking to avoid repetition (Paulus et al., 2017). Detailed descriptions of each summarization model and the readability metric are presented in the following sections.

4.1 PEGASUS

We used PEGASUS that is trained on large text corpus of news text from the web pages to produce abstractive summaries (Zhang et al., 2019b). The architecture of PEGASUS model is Transformer-based encoder-decoder network and the model targets to output the important sentences by masking principal sentences or greedily selected sentences based on the ROUGE (Nallapati et al., 2016) in the input text during the training process. We used the official implementation and the checkpoint of the pre-trained PEGASUS model without any modification and trained this model directly on the lay summary dataset.

4.2 PreSumm

For extractive summarization, we used the Presumm model (Liu and Lapata, 2019) which uses BERT, a pre-trained language model, for news article summarization without any modification. Presumm model uses BERT as a pre-trained encoder. The authors added [CLS] token between the sentences as the input of BERT to obtain sentence representation. This token is used to calculate the score to determine whether each sentence is included in the lay summary.

For training this summarization model, we assumed that the model needs a large-scale dataset that contains thousands of instances to train over one hundred million of parameters. Since the lay summary dataset only consists of 600 documents, we used the CNN/DM dataset that consists of 300K news articles for the pre-training stage before training the lay summary dataset. CNN/DM dataset is a common benchmark used in the summarization literature and the target summary for this dataset is somewhat extractive rather than abstractive, thus we considered this dataset seemed suitable for the extractive summarization. After training the model on the CNN/DM dataset for a few iterations, we switched the dataset with the lay summary dataset.

4.3 Readability of the Sentence

The evaluation metric that is widely used in the summarization literature is ROUGE, which reflects the ratio of overlapping vocabulary between the produced summary and the ground-truth summary. However, ROUGE only focuses on counting the overlapping words and it is unable to determine whether the sentence is difficult or not to understand. We believe the produced lay summary has to be more readable for the lay audience, thus we adopted the readability of the sentence as an additional metric and we combine this metric with extractive summarization. Specifically, we combine this metric with extractive summarization. When we produced the extractive summary based on the important score predicted by Presumm model, we pruned the sentence of which readability score is under a certain threshold.

The readability of the sentence is measured by considering the ratio of jargon. We used the corpus of words developed by Rakedzon et al. (2017). The authors collected 900 million words published on the BBC site and classifying the word as easy, medium, and rare (jargon) based on the frequency of words used on the BBC site. The dictionary contains around 500K words which were the most frequently used. To measure the readability of the sentence, we followed the authors as shown in equation (1) with different constant factors (c_1, c_2, c_3) in front of each ratio (r_1 : medium, r_2 : rare, r_3 : out of dictionary). We used 10, 20, 30 as constant factors in front of each ratio.

$$Score = 100 - (c_1 r_1 + c_2 r_2 + c_3 r_3) \quad (1)$$

Using this metric, we measured the average sentence readability of the abstract and the lay summary in the Laysumm dataset. As shown in table 3, the lay summary achieves high readability than the abstract since it avoids using technical words. In the next section, we present the readability of the produced summary with ROUGE metric to show whether the summarization model can achieve a high score in both ROUGE and readability metrics.

—	# Average Sentence Readability
Abstract	92.25
Laysumm	96.18

Table 3: Average sentence readability.

5 Experiments

5.1 Dataset and Evaluation

We evaluated the performance of the model on the lay summary dataset. The lay summary dataset is divided into the train, validation, and test set (8/1/1 split). Evaluation metrics are ROUGE recall and F1 score in terms of unigram, bigram, and the longest common subsequence overlap.

5.2 Implementation Details

We mainly used the official implementation of the PEGASUS, Presumm, and pre-trained checkpoints provided by the authors. We did not modify any network architecture and for Presumm model, the dataset was switched from CNN/DM to lay summary data after sufficient training steps. After switching the dataset, all the trainable parameters are gradually fine-tuned with a lower learning rate.

Presumm extractive models were trained on dual GPUs (NVIDIA RTX 2080ti) with gradient accumulating every 4 steps. The model was trained for 50,000 steps for the pre-training stage and 10,000 steps after switching the data into the Laysumm dataset. We saved the checkpoints of the model every 200 steps after switching the dataset and performed validation by choosing the top three checkpoints, which have the lowest validation loss, to evaluate the model on the test set. To generate the extractive summary, we selected the sentence from the highest score only if the readability score is over 85 until the number of words in the produced summary is over 150. Trigram Blocking (Paulus et al., 2017) is applied when generating the summary to reduce the redundancy.

PEGASUS model was trained for 20,000 steps on a single GPU (NVIDIA RTX 2080ti) with hyperparameters provided by the authors except for batch size and learning rate. Due to the memory constraints, we decreased batch size to 1 with a decreased learning rate at 0.0001. We saved the checkpoints of the model every 1000 steps and performed the same validation done in the extractive summarization and chose beam search at size 10 to encourage the model not to generate short sentences.

5.3 Results

The best results were achieved by submitting different checkpoints from the validation and test stage for each model. The performance of extractive and abstractive models are summarized in table 4. EXT and ABS indicate Presumm extractive model and PEGASUS abstractive model respectively. ABSEXT means sentences produced by the extractive model are appended to the abstractive summary until the number of words is over 90. As reported in the table 4, the hybrid approach outperforms a solely extractive or abstractive model. Hybrid model benefits from high recall in the extractive model and high precision in the abstractive model.

Model	ROUGE-F1(1/2/L)			ROUGE-R(1/2/L)		
EXT	42.96	17.85	23.38	45.85	18.92	25.02
ABS	43.61	20.51	28.98	43.16	20.35	28.59
ABSEXT	**45.96**	**21.46**	**29.77**	**48.10**	**22.37**	**31.05**

Table 4: Best ROUGE F1, Recall results on test set.

5.4 Analysis of Threshold

In this section, we investigate how the number of words in the produced lay summary affects the performance of the summarization model. We first produced lay summaries using PEGASUS(ABS) and measured the number of words for each summary. Then, we set a standard threshold and appended sentences from the extractive summary produced by PRESUMM(EXT) if the number of words in the abstractive summary is below that limit. Table 5 shows the ROUGE F1 score with respect to different threshold values and the average number of words of lay summaries after appending sentences. ABSEXT with a threshold at 90 performs best and it shows appending sentences from the extractive model to the abstractive summary consistently improves the performance. This makes sense as the abstractive model(ABS) tends to produce short summaries: the average number of words in abstractive summary is 82, whereas the average number of words in the ground-truth lay summaries in the Laysumm dataset is around 110.

5.5 Readability of Summary

We provide ROUGE-F1 and readability scores for each model. As shown in Table 6, for the extractive summary, EXT performs better than EXT W/O R, demonstrating excluding hard sentences improves the performance on both ROUGE and readability metrics. When the extractive summary is combined

Threshold	# of words	ROUGE-F1 (1/2/L)		
—	82.54	43.61	20.51	28.98
70	85.92	44.12	20.33	29.12
90	95.19	**45.96**	**21.46**	**29.77**
115	105.78	45.19	21.00	29.02
135	114.54	45.17	20.95	28.22

Table 5: F1 with different threshold on test set. The first row indicates the abstractive model without being embedded with sentences from the extractive summary.

with abstractive summary (ABSEXT W/O R, ABSEXT), readability constraints slightly improves the performance on both ROUGE-F1 and readability metric. Overall, we observed that our models successfully produce lay summaries that are more readable than the abstract.

Model(dataset)	Readability	ROUGE-F1 (1/2/L)		
EXT W/O R	93.09	42.35	17.66	23.37
EXT	93.39	42.69	17.85	23.38
ABSEXT W/O R	93.83	45.86	21.46	29.76
ABSEXT	**93.85**	**45.96**	**21.46**	**29.77**
(Abstract)	92.25	—	—	—
(LaySumm)	96.18	—	—	—

Table 6: ROUGE-F1 and readability score on test set. EXT W/O R means PRESUMM extractive summarization model without readability constraints, whereas EXT model involves pruning sentences whose readability is under 85 from the produced summary. ABS + EXT W/O R and ABS + EXT indicate sentences from EXT W/O R and EXT W/O R are embedded to abstractive summary respectively. (Abstract) and (Lay summary) are the abstract and the lay summary of Laysumm dataset.

6 Discussion and Future work

We applied transfer learning to mitigate the absence of large-scale datasets to tackle the lay summarization task. While we demonstrated that transfer learning can result in a good performance, it can create a bottleneck for the model due to the discrepancy between the distributions of datasets, resulting in sub-optimal solutions. Our summarization model also excludes the full-text of the paper and tries to produce the summary solely based on the abstract. Although the model achieves good performance, there might exist important points in the body of the paper. It is obvious for humans to utilize the full-text of the paper to write a better lay summary. Creating a large-scale lay summary dataset that handles scholarly documents and considering important sentences from the body text can be a promising direction to address these issues. The readability score might be usefully utilized for constructing the large-scale dataset since it is necessary to pair the difficult sentences and a more readable lay summary.

Secondly, in the optimization process during training the model, we only focused on predicting the only ground truth lay summary. This might limit the capability of the summarization model. Applying the readability score as an additional feature in the training stage would make the model more creative and help the system to summarize the contents while it selectively chooses easier words.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Lutz Bornmann and Rüdiger Mutz. 2015. Growth rates of modern science: A bibliometric analysis based on the number of publications and cited references. *Journal of the Association for Information Science and Technology*, 66(11):2215–2222.

Isabel Cachola, Kyle Lo, Arman Cohan, and Daniel S Weld. 2020. Tldr: Extreme summarization of scientific documents. *arXiv preprint arXiv:2004.15011*.

Asli Celikyilmaz, Antoine Bosselut, Xiaodong He, and Yejin Choi. 2018. Deep communicating agents for abstractive 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 1662–1675, New Orleans, Louisiana. Association for Computational Linguistics.

M. K. Chandrasekaran, G. Feigenblat, E. Hovy, A. Ravichander, M. Shmueli-Scheuer, and A. De Waard. (Forthcoming). Overview and insights from scientific document summarization shared tasks 2020: Cl-scisumm, laysumm and longsumm. *In Proceedings of the First Workshop on Scholarly Document Processing (SDP 2020)*.

Muthu Kumar Chandrasekaran, Michihiro Yasunaga, Dragomir Radev, Dayne Freitag, and Min-Yen Kan. 2019. Overview and results: Cl-scisumm shared task 2019. *arXiv preprint arXiv:1907.09854*.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–494, Berlin, Germany. Association for Computational Linguistics.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention

model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Lieve Hamers et al. 1989. Similarity measures in scientometric research: The jaccard index versus salton's cosine formula. *Information Processing and Management*, 25(3):315–18.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Matthew Honnibal and Mark Johnson. 2015. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1373–1378, Lisbon, Portugal. Association for Computational Linguistics.

Kokil Jaidka, Niyati Chhaya, and Lyle Ungar. 2018. Diachronic degradation of language models: Insights from social media. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 195–200, Melbourne, Australia. Association for Computational Linguistics.

Kokil Jaidka, Muthu Kumar Chandrasekaran, Sajal Rustagi, and Min-Yen Kan. 2016. Overview of the CL-SciSumm 2016 shared task. In *Proceedings of the Joint Workshop on Bibliometric-enhanced Information Retrieval and Natural Language Processing for Digital Libraries (BIRNDL)*, pages 93–102.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.

Manuel J Maña-López, Manuel De Buenaga, and José M Gómez-Hidalgo. 2004. Multidocument summarization: An added value to clustering in interactive retrieval. *ACM Transactions on Information Systems (TOIS)*, 22(2):215–241.

Inderjeet Mani, Gary Klein, David House, Lynette Hirschman, Therese Firmin, and Beth Sundheim. 2002. Summac: a text summarization evaluation. *Natural Language Engineering*, 8(1):43–68.

Kathleen McKeown, Rebecca J Passonneau, David K Elson, Ani Nenkova, and Julia Hirschberg. 2005. Do summaries help? In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 210–217.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2016. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. *arXiv preprint arXiv:1611.04230*.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Pontus Plavén-Sigray, Granville James Matheson, Björn Christian Schiffler, and William Hedley Thompson. 2017. The readability of scientific texts is decreasing over time. *Elife*, 6:e27725.

Tzipora Rakedzon, Elad Segev, Noam Chapnik, Roy Yosef, and Ayelet Baram-Tsabari. 2017. Automatic jargon identifier for scientists engaging with the public and science communication educators. *PloS one*, 12(8):e0181742.

Dmitri G Roussinov and Hsinchun Chen. 2001. Information navigation on the web by clustering and summarizing query results. *Information Processing & Management*, 37(6):789–816.

Claude Sammut and Geoffrey I Webb. 2010. Tf–idf.

Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2020. Discriminative adversarial search for abstractive summarization. *arXiv preprint arXiv:2002.10375*.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada. Association for Computational Linguistics.

Oleg Vasilyev, Tom Grek, and John Bohannon. 2019. Headline generation: Learning from decomposable document titles. *arXiv preprint arXiv:1904.08455*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Kristian Woodsend and Mirella Lapata. 2010. Automatic generation of story highlights. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 565–574, Uppsala, Sweden. Association for Computational Linguistics.

Michihiro Yasunaga, Jungo Kasai, Rui Zhang, Alexander R Fabbri, Irene Li, Dan Friedman, and Dragomir R Radev. 2019. Scisummnet: A large annotated corpus and content-impact models for scientific paper summarization with citation networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7386–7393.

Wonjin Yoon, Yoon Sun Yeo, Minbyul Jeong, Bong-Jun Yi, and Jaewoo Kang. 2020. Learning by semantic similarity makes abstractive summarization better. *arXiv preprint arXiv:2002.07767*.

Haoyu Zhang, Jingjing Cai, Jianjun Xu, and Ji Wang. 2019a. Pretraining-based natural language generation for text summarization. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 789–797, Hong Kong, China. Association for Computational Linguistics.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J Liu. 2019b. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. *arXiv preprint arXiv:1912.08777*.

A Lay Summary Example

We present our approach of using Pre-training with Extracted Gap-sentences for Abstractive Summarization (PEGASUS; Zhang et al., 2019b) to produce the lay summary and combining those with the extractive summarization model using Bidirectional Encoder Representations from Transformers (BERT; Devlin et al., 2019) and readability metrics that measure the readability of the sentence to further improve the quality of the summary. Our model achieves a remarkable performance on ROUGE metrics, demonstrating the produced summary is more readable while it summarizes the main points of the document.

Table 7: An example of a lay summary generated by ABSEXT model. The model only considers the abstract to produce the lay summary.