Attention to COVID-19: Abstractive Summarization of COVID-19 Research with State-Of-The-Art Transformers

Jan Apolline D. Estrella
Department of Computer Science
University of the Philippines, Diliman
Quezon City, Philippines
jdestrella1@up.edu.ph

Christian S. Quinzon
Department of Computer Science
University of the Philippines, Diliman
Quezon City, Philippines
csquinzon@up.edu.ph

Francis George C. Cabarle
Department of Computer Science
University of the Philippines, Diliman
Quezon City, Philippines
fccabarle@up.edu.ph

Jhoirene B. Clemente
Department of Computer Science
University of the Philippines, Diliman
Quezon City, Philippines
jbclemente@up.edu.ph

***Abstract*—The COVID-19 pandemic has led to an overwhelming volume of scientific publications as researchers strive to address the crisis. To alleviate information overload, the COVID-19 Open Research Dataset (CORD-19) was released to help in analyzing large amounts of data and facilitate faster response. Most existing tools based on CORD-19 use extractive summarizers, which suffer from poor coherence and readability. Thus more abstractive summarizers for COVID-19 are needed. Specifically, using state-of-the-art (SOTA) transformers has shown to be successful in summarizing biomedical datasets like arXiv and PubMed.**

In this study, we finetune two checkpoints of SOTA transformer PEGASUS-X on the CORD-19 dataset: PEGASUS-X$_{BASE}$-CORD19 and PEGASUS-X$_{BASE}$-arXiv-CORD19. Our results highlight the importance of finetuning summarizers on domain-specific datasets in the abstractive summarization of COVID-19 research: checkpoints finetuned on CORD-19 outperform other existing checkpoints and transformers finetuned on more general research datasets (e.g., arXiv and PubMed). After stopword removal in evaluation, we observe that PEGASUS-X$_{BASE}$-arXiv-CORD19 surpasses PEGASUS-X$_{BASE}$-CORD19 by a small margin. Our checkpoints still fall behind earlier sequence-to-sequence models; however, this limitation may be due to our constrained GPU resources. Future works, with access to more resources, can further improve our checkpoints for COVID-19 research summarization.

***Index Terms*—COVID-19, natural language processing, abstractive summarization, transformer**

I. Introduction

The COVID-19 pandemic has driven the global research community to publish increasing volumes of publicly available scientific papers to further understand and combat the coronavirus [1]. The accelerated release of publications on the disease leads to information overload, making it difficult for health experts and policymakers to catch up with the latest findings [2].

Artificial intelligence (AI)—through natural language processing (NLP)—has the potential to alleviate health emergencies, such as the current coronavirus crisis, by allowing health experts to efficiently analyze enormous amounts of data and ultimately make prompt responses [3]. Helpful AI-powered techniques include automated text-mining tasks such as summarization.

For the advancement of such AI tools in the fast-paced field of COVID-19, the COVID-19 Open Research Dataset (CORD-19) was released in 2020 [4]. This is a huge corpus of scientific papers on COVID-19 and other related coronaviruses. Some impactful tools that were based on CORD-19 involve tasks like summarization, search and discovery, question answering, paper recommendation, claim verification, and assistive literature review.

Most of these tools combine only elements of information retrieval and extraction of text snippets [4]. COVID-ASK is a question-answering system that only directly copies snippets from documents that might answer a user's question [5]. In the area of summarization, there exist some applications: GRETEL, which integrates transformer-based and graph-based methods [6]; and SAPGRAPH, which takes advantage of the research paper structure [7]. Other extractive summarizers are pre-trained transformer-based models like BERT, RoBERTa, BioBERT, and PubMedBert [6], and the state-of-the-art (SOTA) BertSumExt [8]. These tools are all extractive summarizers, which only copy and concatenate select sentences from a document to generate a summary.

Extractive summaries have drawbacks as they do not resemble human-generated summaries and lack sentence connectivity, readability, and cohesion. Additionally, these summaries can be lengthy, necessitating manual synthesis of information [9], [10].

On the other hand, abstractive summarization can create shorter and more readable human-like summaries—making reading papers more efficient for biomedical experts and more understandable for non-experts such as policymakers. Unfortunately, in recent years, the research community has been focused more on the extractive approach, rather than the more complex task of abstractive summarization [10]. However, the emergence of state-of-the-art transformers has made abstractive summarization more achievable. [11], [12].

With the limitations of extractive summarizers and the lack

of abstractive summarizers, there is a need to push for the progress of abstractive summarization in the specific domain of COVID-19 and other related coronaviruses. There are various purposes for working on abstractive summarization for COVID-19. This task can automatically provide researchers with reliable draft abstracts as they finish writing their paper as demonstrated by COVIDSum [1]. Moreover, abstractive summarizers can be helpful for other NLP tools for COVID-19 papers, like search engines and question-answering (QA) systems. For instance, search engines like Vespa [2] and CO-Search [13] use abstractive summarization to provide a summary for papers they recommend. QA systems such as CAiRE-COVID and CAVIDOQS also take advantage of abstractive multi-document summarization. This can also improve question-answering systems such as CAiRE-COVID [14] and CAVIDOQS [15], which generate an answer to a question by creating a summary from multiple documents. These instruments help clinicians and clinical researchers conduct systematic reviews faster [4].

This study aims to train a top SOTA transformer (i.e., PEGASUS-X) on CORD-19, to improve abstractive summarization for COVID-19 and to leverage other NLP tools, like question-answering and multi-document summarization. The following are the main contributions of our present work:

1) We finetune the SOTA PEGASUS-X—which was found to be successful on other well-known biomedical datasets arXiv and PubMed—on CORD-19, and produce two new checkpoints: PEGASUS-X_{BASE}-CORD19 and PEGASUS-X_{BASE}-arXiv-CORD19.
2) We quantitatively compare the two PEGASUS-X checkpoints finetuned on CORD-19 with previous works via the ROUGE metrics.

Our research also highlights the following insights:

1) Finetuning on domain-specific datasets may enhance the performance of a transformer in the abstractive summarization of COVID-19 research.
2) Our PEGASUS-X checkpoints finetuned on CORD-19 outperform other checkpoints and transformers finetuned on more generic research datasets.

II. Related Literature

A. Summarization

Automated text summarization is the task of compressing text to reduce its length while conserving key information [16]. By generating a summary, a summarizer helps its users capture the main ideas of a paper without reading the entirety of a document, and thus save time and effort [10].

Summarizers can be classified based on various aspects [10].

1) Summarization Domain: A summarizer can be categorized as general or domain-specific [10]. General summarizers can process documents from any domain, while domain-specific summarizers are designed to summarize text from a particular domain, such as COVID-19.

2) Summarization Approach: The approach by which a summary is generated can differ: it can be extractive, abstractive, or hybrid [10]. Extractive summarization is the concatenation of the most important sentences in a document. Abstractive summarization first understands the semantics of the source by converting it into some intermediate representation and then generates a summary with novel sentences that are not in the original document [17]. Summarizers have also been made with hybrid approaches, which are a mix of both extracting and abstractive models.

Extractive summarization is easier to implement and faster to execute. But its summaries tend to have high redundancy, low coherence, and conflicting information. On the contrary, abstractive summarization can produce better and shorter human-like summaries via paraphrasing, compressing, and fusing sentences. In earlier years, it has been found to be difficult to develop as it requires natural language generation, which was still a growing field. Fortunately, abstractive summarization has become more possible with transformers.

3) Input Length: Summarizers can be classified as either short-input or long-input [10]. Short-input summarization generates summaries from short documents like news articles, while long-input summarization focuses on summarizing longer documents such as research papers. Traditional summarizers have mainly been designed for short-input scenarios and struggle to process long sequences effectively, resulting in less accurate summaries. Addressing this challenge, recent works like PEGASUS-X have aimed to extend summarizers to handle long inputs more effectively and efficiently [18].

B. Transformers

Recent text-to-text systems, such as summarizers, are sequence-to-sequence (seq2seq) models that follow an encoder-decoder architecture. The encoder acts as a language model, processing input text and passing it to the decoder, which generates output text one element at a time.

Traditional seq2seq models based on recurrent neural networks (RNNs), have been successful but suffer from two limitations due to their sequential nature: the inability to capture long-range dependencies and the lack of parallelization [12].

Hence, the transformer was developed: this can overcome the aforementioned two limitations by completely relying on *self-attention* [12]. Self-attention maps dependencies not just between the encoder and decoder but also within tokens of an encoder or decoder themselves. This enables transformers to capture global dependencies through traditional attention and learn various contexts for tokens through self-attention. Moreover, unlike sequential RNNs, self-attention can attend to multiple parts of an input sequence simultaneously, allowing for parallelization.

Despite its success, self-attention is not fully understood by researchers and demands significant computational power, particularly for larger models dealing with longer texts. However, advancements in hardware have enabled transformers to showcase their potential. As a result, most state-of-the-art deep-learning models for abstractive text summarization now incorporate components of the original transformer.

Table I shows the top-performing models for the arXiv and PubMed datasets.

To our knowledge, Top-Down and PEGASUS-X are the leading models for abstractive summarization in the biomedical datasets arXiv and PubMed, as shown in Table I. Out

TABLE I
TOP TRANSFORMER-BASED AND RNN-BASED TEXT SUMMARIZERS FOR ARXIV AND PUBMED

Model	arXiv			PubMed		
	R-1	*R-2*	*R-L*	*R-1*	*R-2*	*R-L*
Top-Down (transformer) [19]	51.0	21.9	45.6	51.1	23.3	46.5
PEGASUS-X (transformer) [18]	50.0	21.8	44.6	51.0	24.7	46.6
BART-LS (transformer) [20]	50.2	22.1	45.4	50.3	24.3	45.4
LongT5 (transformer) [21]	48.4	21.9	44.3	50.2	24.8	46.7
DANCER LSTM (RNN) [22]	42.7	16.5	38.4	44.1	17.7	40.3
DANCER RUM (RNN) [22]	41.9	16.0	37.6	44.0	17.7	40.3

of these two, only PEGASUS-X is publicly accessible for training. Hence, we chose to work with PEGASUS-X.

1) Pre-training: Pre-training a transformer involves learning language understanding via training on unlabeled data. This helps the model adapt to specific tasks by learning syntactic, semantic, and linguistic patterns between words and sentences on [23], [24]. Large models are pre-trained from scratch, like Google's PEGASUS, which was trained on 1.5 million mixed samples from C4 and HugeNews.

2) Finetuning: Finetuning a transformer involves learning language usage via training on labeled data. This process consists of updating the pre-trained model's parameters for it to adapt to specific tasks like question-answering, translation, or summarization. For example, $\text{PEGASUS}_{\text{CNN/DailyMail}}$ is created by finetuning the pre-trained PEGASUS model on the DailyMail dataset, which contains new articles and their summaries [25]. Therefore, the finetuned $\text{PEGASUS}_{\text{CNN/DailyMail}}$ model is capable of news summarization. One can also further finetune an already-finetuned model (i.e., checkpoint).

C. PEGASUS-X

Early transformers have been performant on short-input NLP tasks, such as the abstractive summarization of news articles and social media posts [18].

These models however still struggle with long texts [26], especially those found in larger long-document datasets (e.g. PubMed, arXiv, CORD-19). Training transformers to handle lengthy sequences incurs high computation and memory costs [27].

PEGASUS-X extends transformers for efficient long input summarization, achieving SOTA performance on the PubMed dataset [18]. It has two model sizes available for use: $\text{PEGASUS-X}_{\text{BASE}}$, which has 272M parameters; and the memory-heavy $\text{PEGASUS-X}_{\text{LARGE}}$, which has 568M parameters.

PEGASUS-X's success can be attributed to its global-local architecture and novel pre-training objectives [18]. It employs staggered local attention with global tokens, preventing quadratic memory growth of self-attention. As an extension of PEGASUS, it uses Gap Sentences Generation, a novel training objective imitating abstractive summarization.

D. CORD-19 Abstractive Summarizers

1) HITS-Based Attentional Neural Model: The HITS-based attentional neural model modifies the traditional attention mechanism, which treats all sentences in a document as equally important [28]. This study argues that the same words in different sentences carry varying levels of importance.

Achieving higher ROUGE scores than the SOTA BERT-SumAbs when evaluated on CORD-19, this model has proven itself to be effective in synthesizing papers in the biomedical field. However, its novel attention mechanism is yet to be integrated into other pre-trained models, such as the SOTA transformer-based summarizers for biomedical documents.

2) UGDAS: The unsupervised graph-network-based denoiser for abstractive summarization in biomedical domain (UGDAS) consists of a denoiser and a generator [29]. The denoiser encodes sentences using a pre-trained language model with domain knowledge, converting the document into a graph network. Sentences are scored based on importance, considering their position and node weight in the graph. The top-scoring sentences are used by the auto-regressive generator to create the final summary.

Since the denoiser utilizes domain-specific information, the model is able to represent biomedical information more effectively. However, it has only been used to summarize a CORD-19 abstract to generate a paper title. Although UGDAS has surpassed the baseline model BART on CORD-19, its performance on the long-document summarization of the actual content of a research paper is still unknown.

3) COVIDSum: COVIDSum is a model based on BioBERT, which is specifically designed for biomedical text. It extracts important sentences using sentence position heuristics and creates word co-occurrence graphs. The model also employs linguistic knowledge from BioBERT and contextual graph embeddings to generate abstractive summaries with a transformer decoder.

With these techniques, COVIDSum outperformed other CORD-19 summarization models like BERTSumAbs, PEGASUS-LARGE, and BART. However, it has some drawbacks, such as repetition and the usage of extractive summarization as its first module.

While there have been outstanding technologies for CORD-19, there is still a lack of works that build upon other SOTA transformer-based models—especially those that were already trained and evaluated on well-known biomedical datasets (i.e., arXiv and PubMed)—to introduce long-document abstractive summarizers in the specific field of COVID-19.

III. EXPERIMENTAL SETUP

A. Preparing the Dataset

To train a transformer for the task of summarizing COVID-19 papers, we chose the CORD-19 dataset. CORD-19 was loaded via the Hugging Face datasets library. Labeled `allenai/cord19` on Hugging Face, this is the most recent version of the dataset, containing 368K samples. We filtered out samples with no full text or abstract. As done in [1], we divided the remaining 105,097 samples into a 90/5/5 training-validation-test split. The training, validation, and test split then contain 94,587/5,255/5,255 samples.

B. Choosing a Pre-trained Model

We chose two pre-trained PEGASUS-X models publicly available on Hugging Face to finetune: $\text{PEGASUS-X}_{\text{BASE}}$

and PEGASUS-X_{BASE}-arXiv. PEGASUS-X_{BASE}-arXiv is a checkpoint of PEGASUS-X_{BASE} that was already finetuned on arXiv. Both of these models were loaded using the Hugging Face `transformers` library. We used the `PegasusXForConditionalGeneration` to load the pre-trained weights and architecture of a pre-trained model, and the `AutoTokenizer` class to load the `google/pegasus-x-large` tokenizer, which converts raw text into numerical tokens before feeding them for finetuning or summarizing.

C. Finetuning

We used a single NVIDIA A100 80 GB Tensor Core GPU, running on the Ubuntu operating system. We finetuned the two aforementioned pre-trained models on the CORD-19 dataset to generate two new models: PEGASUS-X_{BASE}-CORD19 and PEGASUS-X_{BASE}-arXiv-CORD19.

To work with limited GPU memory resources, we set the batch size to 8 and the number of gradient accumulation steps to 1. To decrease training time, we set the number of epochs to 8. Each finetuning period took around two days.

D. Evaluating

We compared the performance of our finetuned transformers, alongside other previous models, via the Recall-Oriented Understudy for Gisting Evaluation (ROUGE) metrics [30]. It is the most commonly used evaluation method and was thus accepted as a standard in measuring the performance of summarizers [10], especially those that are based on transformers.

ROUGE counts the number of overlapping units between the computer-produced summaries and the ground truth summaries written by humans [30]. There are various kinds of ROUGE metrics. ROUGE-N (R-N) is an n-gram overlap measure between machine-generated and ground truth summaries. For instance, R-1 measures unigram overlaps. While R-2 measures bigram overlaps. ROUGE-L (R-L) is the overlap measure of the longest common subsequences (LCS) between a machine-generated summary and a ground truth summary. ROUGE-L-Sum (R-LSum) is an extension and the summary-level version of R-L such that newlines in a text are interpreted as sentence boundaries, and the LCS overlap between each pair of machine-generated and ground truth sentences is computed [31].

We used the `load_metric` function from the Hugging Face `datasets` library.

IV. Results

We have the following hypotheses that we test in the succeeding sections:

1) Finetuning on a specific dataset, such as CORD-19, improves the performance of a transformer in the abstractive summarization of COVID-19 research.
2) When it comes to abstractive summarization of COVID-19 papers, our PEGASUS-X checkpoints that have been finetuned specifically on the CORD-19 dataset perform better than other transformers finetuned on more generic research datasets.
3) Transformers, like our PEGASUS-X checkpoints, outperform earlier sequence-to-sequence works on COVID-19 abstractive summarization.

A. Comparison of Training Datasets

To test our first hypothesis, we compare various checkpoints of PEGASUS-X_{BASE}, each of which was finetuned on a different dataset: one was finetuned on the general research dataset arXiv, another on the domain-specific dataset that is CORD-19, and the other on both arXiv and CORD-19. The results from this experiment show that PEGASUS-X checkpoints finetuned on CORD-19 perform better than the checkpoint finetuned on arXiv only, thus supporting our first hypothesis.

To perform this experiment, we loaded the arXiv checkpoint (i.e., PEGASUS-X_{BASE}-arXiv) from Hugging Face. We then finetuned PEGASUS-X_{BASE} on CORD-19 to obtain the CORD-19 checkpoint (i.e., PEGASUS-X_{BASE}-CORD19). We also finetuned the aforementioned arXiv checkpoint on CORD-19 to derive the arXiv & CORD-19 checkpoint (i.e., PEGASUS-X_{BASE}-arXiv-CORD19).

Results on the performance of these three checkpoints, when evaluated on the CORD-19 dataset, are shown in Table II. The two checkpoints finetuned on CORD-19 both outperform the arXiv checkpoint. Additionally, the CORD-19 checkpoint achieves the highest scores in terms of R-1, R-L, and R-LSum scores. It surpasses the existing Hugging Face arXiv checkpoint by 3.032 in R-1, 4.309 in R-L, and 5.109 R-LSum. It also performs 1.608/0.672/0.661 better than the arXiv & CORD-19 in R-1/R-L/R-LSum. However, the arXiv & CORD-19 checkpoint outperforms the other models in terms of R-2 scores. We suspect that the presence of stopwords is the reason why the ROUGE scores of the CORD-19 checkpoint are not consistently higher than those of the arXiv & CORD-19 checkpoint.

TABLE II
COMPARISON OF PEGASUS-X CHECKPOINTS AND OTHER TRANSFORMERS ON CORD-19 SUMMARIZATION

Model	Checkpoint	CORD-19			
		R-1	***R-2***	***R-L***	***R-LSum***
PEGASUS-X (base)	arXiv	36.65	12.84	21.14	21.13
	CORD-19	**39.68**	17.15	**26.25**	**26.24**
	arXiv & CORD-19	38.07	**18.22**	25.58	25.58
PEGASUS	arXiv	18.03	2.38	13.26	13.26
PEGASUS	PubMed	25.66	4.22	15.52	15.51
Big Bird-Pegasus (large)	arXiv	18.69	2.41	12.81	12.81
Big Bird-Pegasus (large)	PubMed	26.21	4.20	14.88	14.88

TABLE III
COMPARISON OF PEGASUS-X CHECKPOINTS ON CORD-19 SUMMARIZATION (WITHOUT STOPWORDS)

Model	Checkpoint	CORD-19			
		R-1	***R-2***	***R-L***	***R-LSum***
PEGASUS-X (base)	arXiv	26.01	9.86	16.72	16.72
	CORD-19	30.57	14.86	22.53	22.51
	arXiv & CORD-19	**30.81**	**16.19**	**22.88**	**22.90**

Results on the performance of the three checkpoints when stopwords are removed are shown in Table III. We find that

the arXiv & CORD-19 checkpoint now overtakes the CORD-19 checkpoint. This supports our suspicion that stopwords may have greatly contributed to the scores of the CORD-19 checkpoint. The summaries generated by the arXiv & CORD-19 checkpoint seem to have more matches that are not stopwords, implying that it may have been able to learn COVID-19 terms and contexts better.

Since the difference in ROUGE scores between the CORD-19 checkpoint and the arXiv & CORD-19 checkpoint are relatively small in both Table II and III, it may be more computationally efficient to finetune a model immediately on a specific dataset, rather than finetuning it on a general dataset first. Finetuning more datasets increase training time when there are no publicly available checkpoints to continue training from. Finetuning on a larger general dataset may also require more GPU memory usage when batch size is increased to decrease training time. Hence, when one has limited time and memory resources, they may opt to focus their efforts on finetuning a model on a smaller but more specific dataset. If the lack of resources is not an obstacle, performance may be improved by finetuning on a broad dataset before a particular dataset, but this is not a top priority.

Generally, we see that both the checkpoints finetuned on the CORD-19 have higher scores than the checkpoint finetuned on arXiv only. We then observe that when finetuning a transformer for the task of summarizing papers in a specific field like COVID-19, it is better to finetune on a specific dataset such as CORD-19, rather than relying on other models that have been finetuned only on a general dataset like arXiv.

B. Comparison with Other Transformer-based Summarizers

To test our second hypothesis, we compare our two checkpoints, PEGASUS-X_{BASE}-CORD19 and PEGASUS-X_{BASE}-arXiv-CORD19, with other transformer-based summarizers. The results from this experiment demonstrate that our two PEGASUS-X CORD-19 checkpoints surpass previous transformers that have been finetuned on more generic datasets, thus validating our second hypothesis.

Table II shows the ROUGE scores of the two PEGASUS-X_{BASE} checkpoints and other transformer-based summarizers on CORD-19. It can be seen that the models finetuned on CORD-19 perform better on all ROUGE measurements. This further displays the need for finetuning models on CORD-19. Moreover, we observed that PEGASUS and Big Bird-$Pegasus_{LARGE}$ finetuned on PubMed have higher scores than if these were finetuned on arXiv. This may be an effect of PubMed consisting of scientific text in the biomedical field, which covers the domain of COVID-19 and is more specific than the general scientific domain of arXiv.

We predict that for domain-specific tasks, finetuning transformer models on smaller and more specific domains will yield better results than on general domain datasets. Future work can further finetune a PubMed checkpoint on CORD-19 if it is available. Otherwise, we recommend finetuning PEGASUS-X_{BASE} on PubMed and then on CORD-19 to see if there would be any improvements.

TABLE IV
COMPARISON OF PEGASUS-X CHECKPOINTS AND PREVIOUS SEQUENCE-TO-SEQUENCE WORKS ON CORD-19 SUMMARIZATION

Model	CORD-19	
	R-1	***R-2***
COVIDSum [1]	44.56	18.89
HITS-based Attentional Neural Model [28]	42.79	15.86
PEGASUS-X_{BASE}-CORD19	39.68	17.15
PEGASUS-X_{BASE}-arXiv-CORD19	38.07	18.22
PEGASUS-X_{BASE}-arXiv	36.65	12.84

C. Comparison With Other Sequence-To-Sequence Summarizers

To test our third hypothesis, we compare our two checkpoints, PEGASUS-X_{BASE}-CORD19 and PEGASUS-X_{BASE}-arXiv-CORD19, with other sequence-to-sequence summarizers. The results of this experiment indicate that earlier non-transformer-based summarizers still outperform our transformer-based COVID-19 summarizers, thus denying our third hypothesis.

Results on the performance of the two checkpoints and the other seq2seq summarizers are shown in Table IV. (Note that these earlier works do not indicate their R-L and R-LSums scores.) COVIDSum [1] achieves higher R-1 and R-2 scores than the PEGASUS-X checkpoints. The HITS-based Attentional Neural Model [28] has a higher R-1 score than the best-performing PEGASUS-X checkpoint.

We observe that previous works still generally score higher than our two checkpoints. Moreover, the HITS-based Attentional Neural Model has a lower R-2 score possibly due to its RNN-based decoder.

It is important to note that due to our lack of time and memory resources, we stopped the finetuning of the PEGASUS-X_{BASE}-CORD19 and PEGASUS-X_{BASE}-arXiv-CORD19 checkpoints at a training loss of approximately 0.559 and 0.451 respectively, and the models were still learning as implied by their continuously decreasing training loss. We recommend for future works to finetune PEGASUS-X on the CORD-19 dataset for more epochs to achieve a lower final training loss and thus higher performance.

V. CONCLUSION

To contribute to the advancement of abstractive summarization for COVID-19 research, we finetuned a state-of-the-art transformer, PEGASUS-X, on CORD-19. More specifically, our contributions are two new checkpoints: PEGASUS-X_{BASE}-CORD19 and PEGASUS-X_{BASE}-arXiv-CORD19. Results show that these checkpoints perform better than other transformers finetuned on more general research datasets such as arXiv and PubMed. Our experiments demonstrate the following insights. First, finetuning on a smaller yet more specific dataset like CORD-19 may enhance the performance of a transformer in domain-specific abstractive summarization. Additionally, because the difference between the performance of PEGASUS-X_{BASE}-CORD19 and PEGASUS-X_{BASE}-arXiv-CORD19 is relatively small, we find that it may be more computationally efficient to finetune a model immediately on a more specific dataset rather than on a general dataset first when there are no publicly available checkpoints to train. Second, our PEGASUS-X CORD-19

checkpoints perform better than already existing transformers finetuned on generic research datasets, implying the need for abstractive summarizers specifically trained on COVID-19 datasets. Lastly, earlier seq2seq summarizers still generally perform better than our PEGASUS-X checkpoints. However, it is important to note that the performance of our checkpoints in the third experiment is limited due to our lack of GPU resources. Hence, future works can improve our PEGASUS-X checkpoints by finetuning these models for more epochs when resources permit.

ACKNOWLEDGMENT

We would like to express our sincere gratitude to the College of Engineering of the University of the Philippines for providing us with access to their High-Performance Computing resources.

REFERENCES

[1] X. Cai, S. Liu, L. Yang, Y. Lu, J. Zhao, D. Shen, and T. Liu, "Covidsum: A linguistically enriched scibert-based summarization model for covid-19 scientific papers," *Journal of Biomedical Informatics*, vol. 127, p. 103999, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1532046422000156

[2] L. L. Wang and K. Lo, "Text mining approaches for dealing with the rapidly expanding literature on COVID-19," *Briefings in Bioinformatics*, vol. 22, no. 2, pp. 781–799, 12 2020. [Online]. Available: https://doi.org/10.1093/bib/bbaa296

[3] M. Ciotti, M. Ciccozzi, A. Terrinoni, W.-C. Jiang, C.-B. Wang, and S. Bernardini, "The covid-19 pandemic," *Critical Reviews in Clinical Laboratory Sciences*, vol. 57, no. 6, pp. 365–388, 2020, pMID: 32645276. [Online]. Available: https://doi.org/10.1080/10408363.2020.1783198

[4] L. L. Wang, K. Lo, Y. Chandrasekhar, R. Reas, J. Yang, D. Burdick, D. Eide, K. Funk, Y. Katsis, R. Kinney, Y. Li, Z. Liu, W. Merrill, P. Mooney, D. Murdick, D. Rishi, J. Sheehan, Z. Shen, B. Stilson, A. Wade, K. Wang, N. X. R. Wang, C. Wilhelm, B. Xie, D. Raymond, D. S. Weld, O. Etzioni, and S. Kohlmeier, "Cord-19: The covid-19 open research dataset," 2020. [Online]. Available: https://arxiv.org/abs/2004.10706

[5] J. Lee, S. S. Yi, M. Jeong, M. Sung, W. Yoon, Y. Choi, M. Ko, and J. Kang, "Answering questions on covid-19 in real-time," 2020. [Online]. Available: https://arxiv.org/abs/2006.15830

[6] Q. Xie, J. Huang, T. Saha, and S. Ananiadou, "Gretel: Graph contrastive topic enhanced language model for long document extractive summarization," 2022. [Online]. Available: https://arxiv.org/abs/2208.09982

[7] S. Qi, L. Li, Y. Li, J. Jiang, D. Hu, Y. Li, Y. Zhu, Y. Zhou, M. Litvak, and N. Vanetik, "SAPGraph: Structure-aware extractive summarization for scientific papers with heterogeneous graph," in *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Nov. 2022, pp. 575–586. [Online]. Available: https://aclanthology.org/2022.aacl-main.44

[8] Y. Liu and M. Lapata, "Hierarchical transformers for multi-document summarization," 2019. [Online]. Available: https://arxiv.org/abs/1905.13164

[9] G. Sharma and D. Sharma, "Automatic text summarization methods: A comprehensive review," *SN Computer Science*, vol. 4, no. 1, p. 33, Oct 2022. [Online]. Available: https://doi.org/10.1007/s42979-022-01446-w

[10] W. S. El-Kassas, C. R. Salama, A. A. Rafea, and H. K. Mohamed, "Automatic text summarization: A comprehensive survey," *Expert Systems with Applications*, vol. 165, p. 113679, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417420305030

[11] A. A. Syed, F. L. Gaol, and T. Matsuo, "A survey of the state-of-the-art models in neural abstractive text summarization," *IEEE Access*, vol. 9, pp. 13 248–13 265, 2021.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[13] A. Esteva, A. Kale, R. Paulus, K. Hashimoto, W. Yin, D. Radev, and R. Socher, "Covid-19 information retrieval with deep-learning based semantic search, question answering, and abstractive summarization," *NPJ digital medicine*, vol. 4, no. 1, pp. 1–9, 2021.

[14] D. Su, Y. Xu, T. Yu, F. B. Siddique, E. J. Barezi, and P. Fung, "Caire-covid: A question answering and query-focused multi-document summarization system for covid-19 scholarly information management," 2020. [Online]. Available: https://arxiv.org/abs/2005.03975

[15] N. Wang, H. Liu, and D. Klabjan, "Large-scale multi-document summarization with information extraction and compression," *arXiv preprint arXiv:2205.00548*, 2022.

[16] M. Allahyari, S. Pouriyeh, M. Assefi, S. Safaei, E. D. Trippe, J. B. Gutierrez, and K. Kochut, "Text summarization techniques: A brief survey," 2017. [Online]. Available: https://arxiv.org/abs/1707.02268

[17] H. Lin and V. Ng, "Abstractive summarization: A survey of the state of the art," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 9815–9822, Jul. 2019. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/5056

[18] J. Phang, Y. Zhao, and P. J. Liu, "Investigating efficiently extending transformers for long input summarization," 2022. [Online]. Available: https://arxiv.org/abs/2208.04347

[19] B. Pang, E. Nijkamp, W. Kryściński, S. Savarese, Y. Zhou, and C. Xiong, "Long document summarization with top-down and bottom-up inference," 2022. [Online]. Available: https://arxiv.org/abs/2203.07586

[20] W. Xiong, A. Gupta, S. Toshniwal, Y. Mehdad, and W.-t. Yih, "Adapting pretrained text-to-text models for long text sequences," 2022. [Online]. Available: https://arxiv.org/abs/2209.10052

[21] M. Guo, J. Ainslie, D. Uthus, S. Ontanon, J. Ni, Y.-H. Sung, and Y. Yang, "Longt5: Efficient text-to-text transformer for long sequences," 2021. [Online]. Available: https://arxiv.org/abs/2112.07916

[22] A. Gidiotis and G. Tsoumakas, "A divide-and-conquer approach to the summarization of long documents," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 3029–3040, 2020.

[23] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. [Online]. Available: https://aclanthology.org/2020.emnlp-demos.6

[24] K. S. Kalyan, A. Rajasekharan, and S. Sangeetha, "Ammus : A survey of transformer-based pretrained models in natural language processing," 2021. [Online]. Available: https://arxiv.org/abs/2108.05542

[25] X. Zhang, F. Wei, and M. Zhou, "Hibert: Document level pre-training of hierarchical bidirectional transformers for document summarization," 2019. [Online]. Available: https://arxiv.org/abs/1905.06566

[26] W. Guan, I. Smetannikov, and M. Tianxing, "Survey on automatic text summarization and transformer models applicability," in *Proceedings of the 2020 1st International Conference on Control, Robotics and Intelligent System*, ser. CCRIS '20. New York, NY, USA: Association for Computing Machinery, 2021, p. 176–184. [Online]. Available: https://doi.org/10.1145/3437802.3437832

[27] I. Beltagy, M. E. Peters, and A. Cohan, "Longformer: The long-document transformer," *CoRR*, vol. abs/2004.05150, 2020. [Online]. Available: https://arxiv.org/abs/2004.05150

[28] X. Cai, K. Shi, Y. Jiang, L. Yang, and S. Liu, "Hits-based attentional neural model for abstractive summarization," *Knowledge-Based Systems*, vol. 222, p. 106996, 2021.

[29] Y. Du, Y. Zhao, J. Yan, and Q. Li, "Ugdas: Unsupervised graph-network based denoiser for abstractive summarization in biomedical domain," *Methods*, vol. 203, pp. 160–166, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1046202322000792

[30] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in *Text summarization branches out*, 2004, pp. 74–81.

[31] A. See, P. J. Liu, and C. D. Manning, "Get to the point: Summarization with pointer-generator networks," *CoRR*, vol. abs/1704.04368, 2017. [Online]. Available: http://arxiv.org/abs/1704.04368