


# CRAFT : Cross-domain Abstractive Summarization through Incremental Fine-tuning

Marko Veljanovski, Ryan Newkirk, Lingji Kong, Cesar Villalta Reyna  
Northwestern University

## Abstract

General pre-trained language models perform best on downstream NLP tasks when fine-tuned on domain-specific datasets. However, this remarkable performance tends to be limited within their designated domains, making it challenging to generalize their capabilities across diverse domains. We propose *Cross-domain Abstractive summarization through incremental Fine-Tuning (CRAFT *), a general model capable of summarizing texts from multiple domains. Our model incrementally fine-tunes using the  $k$  most similar documents from a collection of datasets, containing a diverse range of cross-domain text summarizations. From a short sample test on CNN/DailyMail, we were able to achieve better performance than a baseline PEGASUS<sub>LARGE</sub> which was fully fine-tuned on the dataset. Our model’s flexible design suggests it may be applicable to a wide range of downstream NLP tasks.

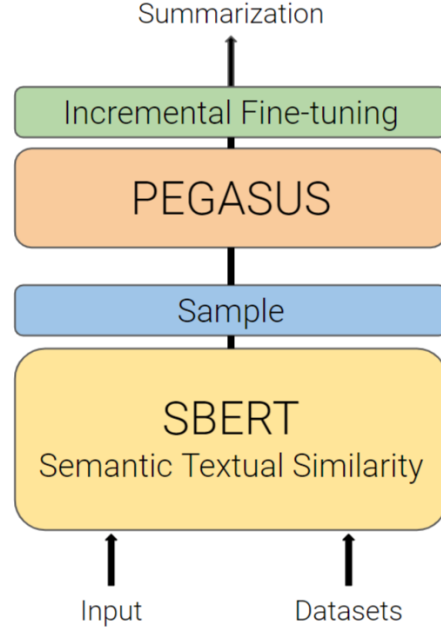


Figure 1: Model Pipeline Visualization

## 1 Introduction

Advanced pre-trained language models have demonstrated their prowess in downstream Natural Language Processing (NLP) tasks by achieving remarkable performance when fine-tuned on specific datasets (Talmor et al., 2019) (Zhang et al., 2020). However, it is important to note that these models excel primarily within their designated domains. Even when performing the same task, such as text summarization, the model’s performance can be influenced by subtle variations in the domain, making it challenging to generalize their capabilities across diverse domains with differences in language use, style, purpose, etc. We propose CRAFT, which alleviates the constraints of domain-specific fine-tuning and pushes the boundaries of cross-domain text summarization.

By first gathering the most similar documents from a diverse corpus of cross-domain documents and incrementally fine-tuning a general pre-trained language model on this new dataset, we now have a general method to fine-tune a general pre-trained model based on how each input is written.

We propose a novel dataset for the NLP field: a cross-domain text summarization dataset, to test our model on. The inclusion of diverse domains in the dataset, coupled with the high variance observed in the lengths of tokenized sequences makes it a challenging task.

## 2 Dataset

For our cross-domain text summarization (CDTS) research, we customized a challenging dataset by selecting diverse datasets from various domains to ensure comprehensive coverage. The CDTS dataset

includes news articles from two major sources, CNN and DailyMail, along with their corresponding headlines (Hermann et al., 2015; Nallapati et al., 2016), dialogues with summaries from the SAM-Sum Corpus (Gliwa et al., 2019), scientific papers with abstracts from Arvix and PubMed (Cohan et al., 2018), and legislature bills with summaries from the BillSum dataset (Kornilova and Eidelman, 2019).

## 2.1 Sampling Method

To create a balanced representation of our four diverse domains, we employed a sampling method as follows:

- **Balanced:** We randomly sampled (10000, 800, 800) entries from each domain for the training, validation, and test sets, respectively.

To explore the impact of unbalanced training sets, we provided additional options using the following sampling strategies:

- **All:** We included all text and summary pairs from the chosen datasets in the training set.
- **40k:** We randomly sampled 40,000 entries from all text and summary pairs.
- **100k:** We randomly sampled 100,000 entries from all pairs.

The comprehensive analysis of CDTs dataset and the starter code for loading it can be found at [https://github.com/nemonemonee/cdts\\_dataset.git](https://github.com/nemonemonee/cdts_dataset.git).

## 3 Model

We propose Abstractive summarization through incremental **Fine-Tuning (CRAFT<sub>⚙️</sub>)**, a two-step pipeline utilizing PEGASUS<sub>LARGE</sub> and a distilled SROBERTa model. The pipeline process can be visualized in Figure 1. To start, we use SROBERTa, a sentence transformer, for computing all of the sentence embeddings (Reimers and Gurevych, 2019). In particular, we begin by computing the sentence embeddings for all documents in our fine-tuning dataset  $X$ . Then, for every document in our testing dataset  $T$ , we compute its sentence embedding and proceed by computing the *cosine-similarity* between this current document and all fine-tuning documents, calculated as

$$\text{cosine\_similarity}(\mathbf{x}, \mathbf{t}) = \frac{\mathbf{x} \cdot \mathbf{t}}{\|\mathbf{x}\| \|\mathbf{t}\|}$$

We then sort the documents by the calculated *cosine-similarity* to select the top  $k$  most similar documents. Following, we incrementally fine-tune PEGASUS<sub>LARGE</sub> on this small dataset. Note that depending on the testing dataset, our baseline PEGASUS<sub>LARGE</sub> is one that is fully fine-tuned on that dataset. Thus, we explore how any additional small or "incremental" further fine-tuning on a selection of different datasets will impact performance.

## 4 Experiment

### 4.1 Evaluation Matrices

To test the validity of our model, we implement ROUGE metric (Lin, 2004) to compare the summary generated by a standard PEGASUS<sub>LARGE</sub> model and the summary generated by our fine-tuned model with the ground truth. Each ROUGE F1 score such as R1, R2 and RL were computed.

### 4.2 Incremental Finetuning on Original Train Set

To save time and computation power, we used various machines from the Wilkinson Lab to run our experiments. We fine-tuned a PEGASUS<sub>LARGE</sub> pre-trained on the CNN/DailyMail dataset. To compute similarity scores, we used a distilled SROBERTa to encode the sentences from the train and test sets. For a given article in test, we computed the  $k = 16$  most similar documents from the train set using a cosine similarity metric. The similar documents picked were then used to incrementally fine-tune our PEGASUS model. It took approximately 1.5 hours to tokenize and compute the sentence embedding. In the fine-tuning process, we trained using 1 step and in total, took approximately half an hour to finish the first 50 samples from the test. The most similar 16 documents with batch size of 1 was fed to our model. For our optimizer, we use Adam (Kingma and Ba, 2017) with a learning rate of  $5e - 4$ . Regularization using weight decay was also employed.

### 4.3 Incremental Finetuning on Additional Datasets

To encode our sentences, we used a distilled SROBERTa on the XSum dataset to serve as our similarity comparison between an example from CNN/DailyMail and a subsample of XSum dataset. We used the same fine-tuned model as section 4.2. For a given article in CNN/DailyMail, we com-

Model	Rogue Score(R-1/R-2/R-L)
PEGASUS <sub>LARGE</sub> fine-tuned on CNN/DailyMail	33.66/14.26/24.02
Incremental fine-tuning on CNN/DailyMail	35.01/16.55/27.44

Table 1: Comparison of a baseline PEGASUS<sub>LARGE</sub> fine-tuned on CNN/DailyMail with the same model incrementally fine-tuned on  $k = 16$  most similar subsets of CNN/DailyMail. The ROUGE Scores were computed on a 50 testing sample size from CNN/DailyMail

Model	Rogue Score(R-1/R-2/R-L)
PEGASUS <sub>LARGE</sub> fine-tuned on CNN/DailyMail	31.93/12.23/22.88
Incremental fine-tuning on XSum	33.16/13.09/23.61

Table 2: Comparison of a baseline PEGASUS<sub>LARGE</sub> fine-tuned on CNN/DailyMail with the same model incrementally fine-tuned on  $k = 100$  most similar subsets of XSum. The ROUGE Scores were computed on a 250 testing sample size from CNN/Dailymail

puted the  $k = 100$  most similar documents from XSum using a cosine similarity metric. The similar documents picked were then used to incrementally fine-tune our PEGASUS model.

In the fine-tuning process, we trained using 1 step and in total, took approximately 6 hours to finish.

#### 4.4 The CDTs Dataset

This will be discussed in the future work section. We do not have the time and computing units to finish this in time.

### 5 Results and Analysis

Table 1 and 2 show the performance of our fine-tuned models. We see that incremental fine-tuning on CNN/DailyMail showed considerable improvement in the ROUGE score. This improvement could be attributed with the fact that there would be more similar documents in CNN/DailyMail and therefore produce a better fine-tuned dataset.

A similar improvement can be seen when we finetune using XSum, as shown in Table 2. With a much bigger finetune dataset with  $k = 100$ , our model has more examples to use to improve upon and therefore outputs a better ROUGE score. It could also be that the improvement is due to XSum also being a news dataset. Here is a sample result (prediction 1 is from the original PEGASUS and prediction 2 is from the incrementally finetuned PEGASUS, the label is the actual summarization given):

**prediction 1 :** A student has admitted to hanging a noose from a tree near a student union,

Duke says. The prestigious private school didn't identify the student, citing federal privacy laws.

**prediction 2 :** A student admitted to hanging a noose made of rope from a tree near a student union. The student was no longer on campus and will face student conduct review.

**label :** Student is no longer on Duke University campus and will face disciplinary review. School officials identified student during investigation and the person admitted to hanging the noose, Duke says. The noose, made of rope, was discovered on campus about 2 a.m.

We can see that the 2nd prediction is more similar to the actual label.

### 6 Limitations

Our primary limitation throughout the experiments done was without compute power. Many of the original experiments planned could not be accomplished with the limited amount of compute power we had coupled with the remaining time.

In addition, the datasets that we tested on do not come from different numbers of domains, only primarily focusing on news dataset.

Also, since we are using a different finetuning dataset for each example from the test set, our CRAFT model can only handle small test sets.

### 7 Future Work

#### 7.1 More Experiments


We want to test on our custom datasets. In particular, we plan on testing the performance of 6

PEGASUS<sub>LARGE</sub> models, where 4 are fine-tuned on the respective 4 datasets: CNN/DailyMail, SAM-Sum Corpus, Arxiv/PubMed, BillSum dataset, one on all 4 i.e. on our custom CDTs dataset, and a final model fine-tuned on CDTs with the addition of incremental fine-tuning.

## 7.2 Ablation Study

We want to experiment with different  $k$  and the incremental finetune arguments. We want to test if using TFIDF will demonstrate a better results than using SBERT. We also want to use different summarization models other than PEGASUS, to see how our incremental fine-tuning method applies to other general pre-trained language models.

## 8 Conclusion

In conclusion, we have introduced CRAFT : Cross-domain Abstractive Summarization through Incremental Fine-tuning, a model capable of summarizing texts from multiple domains. Our empirical results on incrementally fine-tuning a pre-trained language model on a diverse collection of datasets demonstrate that our model surpasses a baseline PEGASUS<sub>LARGE</sub> model fine-tuned on the CNN/DailyMail dataset. This highlights the potential of our model to generalize across domains and tasks. However, limited computational resources hindered our ability to experiment with more datasets. Future work involves exploring our custom CDTs dataset and conducting an ablation study to further improve our model.

## References

- 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](#). *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*.
- Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. [SAMSum corpus: A human-annotated dialogue dataset for abstractive summarization](#). In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79, Hong Kong, China. Association for Computational Linguistics.
- Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. [Teaching machines to read](#)

- [and comprehend](#). In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.
- Diederik P. Kingma and Jimmy Ba. 2017. [Adam: A method for stochastic optimization](#).
- Anastassia Kornilova and Vlad Eidelman. 2019. [Billsum: A corpus for automatic summarization of us legislation](#).
- Chin-Yew Lin. 2004. [ROUGE: A package for automatic evaluation of summaries](#). In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.
- Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. 2016. [Abstractive text summarization using sequence-to-sequence RNNs and beyond](#). In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.
- Nils Reimers and Iryna Gurevych. 2019. [Sentence-bert: Sentence embeddings using siamese bert-networks](#).
- Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. [CommonsenseQA: A question answering challenge targeting commonsense knowledge](#). 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 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.
- Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter J. Liu. 2020. [Pegasus: Pre-training with extracted gap-sentences for abstractive summarization](#).