Semantic Self-Segmentation for Abstractive Summarization of Long Documents in Low-Resource Regimes

Gianluca Moro [1,2], **Luca Ragazzi** [1]

[1]Department of Computer Science and Engineering, University of Bologna, Cesena Campus
Via dell'Università 50, I-47522 Cesena, Italy
[2]CNIT
{gianluca.moro, l.ragazzi}@unibo.it

Abstract

The quadratic memory complexity of transformers prevents long document summarization in low computational resource scenarios. State-of-the-art models need to apply input truncation, thus discarding and ignoring potential summary-relevant contents, leading to a performance drop. Furthermore, this loss is generally destructive for semantic text analytics in high-impact domains such as the legal one. In this paper, we propose a novel semantic self-segmentation (Se3) approach for long document summarization to address the critical problems of low-resource regimes, namely to process inputs longer than the GPU memory capacity and produce accurate summaries despite the availability of only a few dozens of training instances. Se3 segments a long input into semantically coherent chunks, allowing transformers to summarize very long documents without truncation by summarizing each chunk and concatenating the results. Experimental outcomes show the approach significantly improves the performance of abstractive summarization transformers, even with just a dozen of labeled data, achieving new state-of-the-art results on two legal datasets of different domains and contents. Finally, we report ablation studies to evaluate each contribution of the components of our method to the performance gain.

Introduction

State-of-the-art solutions on abstractive summarization are built upon the transformer model (Vaswani et al. 2017) with quadratic time and memory complexities in the input size (Lewis et al. 2020; Zhang et al. 2020a; Raffel et al. 2020; Qi et al. 2020). Such models have been trained with short inputs, so they struggle to model long sequences accurately in downstream tasks. Thus, efficient transformers with linear complexity have been proposed to process longer sequences by reducing the attention mechanism calculation (Kitaev, Kaiser, and Levskaya 2020; Beltagy, Peters, and Cohan 2020; Zaheer et al. 2020; Huang et al. 2021; Choromanski et al. 2021; Xiong et al. 2021; Guo et al. 2021). However, training large transformers requires high-resource settings (Sharir, Peleg, and Shoham 2020; Ahmed and Wahed 2020), leaving long document summarization an open research problem in low-resource regimes with limited GPU memories and only dozens of labeled training instances.

One of the domains most affected by long documents and low-resource settings of labeled data is the legal one, where reading and evaluating legal cases are labor-intensive and time-consuming tasks for legal experts (Kornilova and Eidelman 2019). Legal texts are generally long with a complex and articulated structure, characterized by longer sentences than other domains that make up long reasonings, understandable only after reading the entire textual details (Kanapala, Pal, and Pamula 2019).

Input truncation, unavoidable for long sequences with a low-memory GPU, ignores valuable information, destroying the summary semantic, which is a critical problem also in multi-document summarization (Moro et al. 2022). To address this problem, particularly relevant in the legal domain, we propose a new approach for long document summarization: *Semantic Self-Segmentation* (Se3).[1] Se3 creates high-correlated source-target pairs by segmenting long texts into semantically coherent chunks with lengths modulated to fit into the GPU memory, and pairing them with the most similar summary part, enabling transformers to process documents without truncation. This approach works as a data augmentation strategy to cope with the lack of labeled instances, usually addressed with transfer learning methods (Domeniconi et al. 2014b,c, 2015a). As far as we know, this is the first study on text summarization with limited GPU memories and labeled data scarcity.

Given the complexity of summarizing long legal documents, we experiment on two legal datasets of different domain and content sizes, using Se3 combined with BART (Lewis et al. 2020) and LED (Beltagy, Peters, and Cohan 2020) on a single Titan Xp GPU of 12GB memory. Results show that Se3 significantly improves the performance of abstractive summarization transformers, even with just a few dozens of labeled training data. Moreover, to analyze where the performance gain comes from, we perform ablation studies and prove the importance of each module of Se3. Finally, we analyze the accuracy of the predicted summaries.

Our contributions are: i) the Se3 method to address long document summarization in low-resource regimes; ii) the research advancement on abstractive summarization in legal domains, whose documents are more challenging to analyze, achieving new state-of-the-art results on two datasets.

[1] Solution website: https://disi-unibo-nlp.github.io/projects/se3

Related Work

Long document summarization. Although most solutions focus on short inputs because of the quadratic complexity of transformers, several works presented new approaches to summarize long texts. Çelikyilmaz et al. (2018) introduced a hierarchical model that handles the encoding phase through collaborating agents responsible for processing each text subsection. Liu and Chen (2019) and Xu et al. (2020) proposed to exploit the discourse segmentation to extract the salient content for extractive summarization. Gidiotis and Tsoumakas (2020) introduced a divide-and-conquer approach that relies on structured documents to summarize each section independently. Bajaj et al. (2021) compressed long texts by extracting the sentences that best correlate with the summary, adopting an extract-then-abstract paradigm. Rohde, Wu, and Liu (2021) and Grail, Perez, and Gaussier (2021) modified the standard transformer by adding hierarchical attention layers. Manakul and Gales (2021) showed that applying local self-attention and an explicit content selection improves the performance of large pre-trained quadratic transformers. Cui and Hu (2021) proposed an extractive model that can summarize inputs of arbitrary size without truncation by using a memory network.

Legal document summarization. Most of the summarization solutions in the legal domain are extractive (Galgani, Compton, and Hoffmann 2015; Tran, Nguyen, and Satoh 2018; Anand and Wagh 2019; Jain, Borah, and Biswas 2021a,b), whereas few studies are abstractive. A first comparative analysis that shows the better performance of abstractive approaches than extractive ones has been proposed by de Vargas Feijó and Moreira (2019), summarizing Brazilian legal rulings. Afterward, Zhang et al. (2020a) achieved new state-of-the-art results on the legal dataset BillSum (Kornilova and Eidelman 2019) with PEGASUS, a transformer-based model with a self-supervised pre-training objective tailored for the abstractive summarization task. In contrast, Huang et al. (2020) extended a pointer-generator network with legal domain-specific knowledge to generate abstractive summaries in the legal public opinion domain.

Low-resource summarization. About low-resource studies, prior works have only focused on data scarcity. Parida and Motlícek (2019) and Magooda and Litman (2020) proved that augmenting training instances with synthetic data improves the summarization accuracy in low-resource conditions. Bajaj et al. (2021) applied long document summarization with few labeled data, proposing a new method to extract salient sentences from the source. Yu, Liu, and Fung (2021) introduced a new low-resource setting dataset to investigate several adaptive pre-training strategies to cope with the absence of data. Chen and Shuai (2021) proposed meta-transfer learning combined with multiple corpora to improve the accuracy after training models with few labeled data.

Our work. Unlike previous works, we propose a new approach for abstractive long document summarization to address both issues of low-resource regimes, i.e., limited GPU memories and labeled data scarcity, by semantically segmenting long inputs into GPU memory-adaptable chunks.

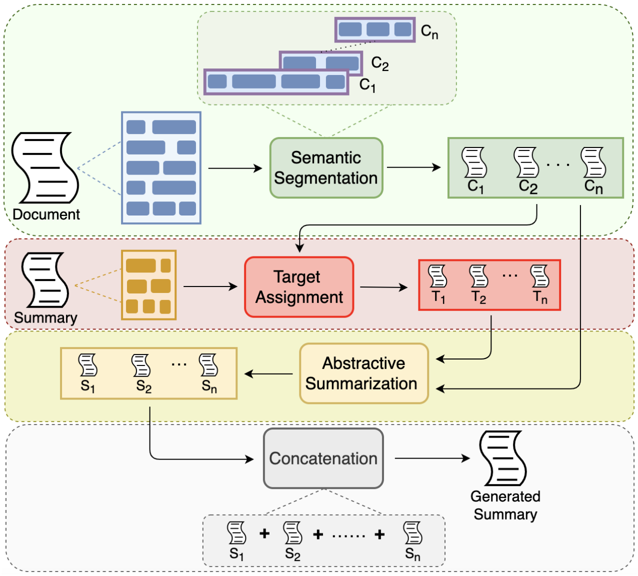


Figure 1: The overview of Se3 for the abstractive summarization of a long input. First, a document composed of many sentences, i.e., blue rectangles, is segmented into content-wise chunks (green phase). Afterward, each summary sentence, i.e., orange rectangles, is assigned to the most similar chunk, creating new high-correlated source-target pairs (red phase) used to train summarization models (yellow phase). At inference time, the final summary is obtained from concatenating the chunk summaries (gray phase).

Preliminary

We provide a better definition of *low-resource regimes*.

Hardware memory scarcity: small and medium-sized organizations can generally afford low-budget GPUs (e.g., 12GB of memory). These memories limit the training of medium neural models, mostly with long documents, causing "out of memory" exceptions.

Data scarcity: in real scenarios, datasets might consist of only a few dozens of labeled instances. This data shortage limits the learning chances and produces underfit models.

Method

Our semantic self-segmentation (Se3) approach for abstractive long document summarization allows fine-tuning transformers on entire long inputs without truncation with limited GPUs (Fig. 1). Concretely, Se3 segments a long document D into N small chunks, *resolving the hardware memory scarcity issue* since we avoid processing a long text in a single input. Afterward, each sentence of the summary of D is assigned to the most similar input chunk, *resolving the data scarcity issue* since we obtain N chunk-target pairs that augment the training instances.

Two observations motivate this solution: i) truncating input to a fixed length may discard valuable information; ii) in a low-resource scenario, there may also be a lack of labeled data to fine-tune effectively pre-trained models.

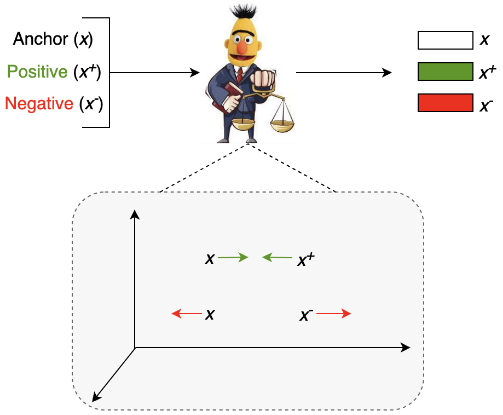


Figure 2: The metric learning of LEGAL-BERT with the triplet loss. The aim is to create meaningful sentence embeddings by projecting topic-related sentences closer in the vector space and the different ones farther.

Semantic Self-Segmentation

We need three elements to train transformers to summarize long inputs without truncation with a limited GPU memory.

The *chunk size* is needed to standardize the chunk content within a range since pre-trained transformers have been trained on fixed sizes, so they struggle to process chunks of very different sizes. Moreover, this range lets users change input size to adapt chunks to the GPU memory available and best leverage the capability of transformers.

A *pre-trained language model* (PLM) is needed to represent the sentences semantically. Thus, as we test Se3 on legal documents, we use LEGAL-BERT (Chalkidis et al. 2020), a BERT model pre-trained on legal corpora.[2] Further, we fine-tune LEGAL-BERT with metric learning to learn whether two sentences belong to the same section. This learning trains the model to enrich the sentence representation with the thematic meaning, essential for our text segmentation to split sentences. For metric learning, we use a public dataset created in a self-supervised manner (Ein-Dor et al. 2018), as done with paper citations in Moro and Valgimigli (2021), to train models to project sentences of the same section closer in the vector space and the different ones farther (Fig. 2). We consider two ranking losses in our experiments, i.e., the triplet and the contrastive loss. The triplet loss takes as input a triplet composed of a sentence from a section (anchor, x), a sentence from the same section (positive, x^+), and a sentence from a different section (negative, x^-). The function minimizes the distance between x and x^+ and maximizes the distance between x and x^-, considering a margin m:

$$\mathcal{L}_{\text{triplet}} = \max(||x - x^+|| - ||x - x^-|| + m, 0) \quad (1)$$

The contrastive loss takes as input a triplet composed of a sentence from a section x, a second sentence y, and a label l, meaning whether the two sentences belong to the same section (1 if true, 0 otherwise). The loss is as follows:

$$\mathcal{L}_{\text{contrastive}} = l \cdot ||x-y|| + (1-l) \cdot \max(m - ||x-y||, 0) \quad (2)$$

Algorithm 1: Semantic Self-Segmentation

Input: $model$; $doc\_sentences$; $summary\_sentences$
Parameters: $L_s \leftarrow$ lower size; $U_s \leftarrow$ upper size
Output: The chunk-target pairs

```
Let $chunks = []$
Let $current\_chunk = []$
for $s_d$ in $doc\_sentences$ do
    if $len(current\_chunk) + len(s_d) < L_s$ then
        $current\_chunk.append(s_d)$
    else if $len(current\_chunk) + len(s_d) > U_s$ then
        $chunks.append(current\_chunk)$
        $current\_chunk \leftarrow []$
    else
        Perform the Semantic Similarity (Alg. 2)
    end if
end for
$targets \leftarrow$ Perform the Target Assignment (Alg. 3)
return $(chunks, targets)$
```

Therefore, our text segmentation algorithm uses the trained language model to produce semantically meaningful sentence embeddings to create the chunks.

A *chunk target* is needed to train abstractive summarization models since we are in a supervised machine learning scenario. For this reason, we assign the most similar part of the summary to the chunks, creating high-correlated source-target pairs. In detail, we apply a syntactic assignment where we pair each sentence of the target summary to the chunk that maximizes the ROUGE-1 precision (Lin 2004). Unlike recall, f-measure, or ROUGE-L, we choose such a metric to guarantee a more proper matching for abstractive summaries. The motivations are the following: i) ROUGE-1 checks for uni-gram matching between the summary sentences and the source document, searching the chunk where a summary sentence can be better summarized; ii) the precision metric scores how much content of a summary sentence is within a chunk, searching for the best content coverage.

Algorithm

Let $s_{d1}, s_{d2}, ..., s_{dn}$ be the sentences of a document D obtained using the state-of-the-art tokenizer PySBD (Sadvilkar and Neumann 2020). Let $s_{s1}, s_{s2}, ..., s_{sm}$ be the sentences of the actual summary of D. Let L_s, U_s be the chunk's lower and upper size, respectively. To create the chunk c_i, along with its target t_i, Se3 performs the following steps (Alg. 1):

1. Given s_{dj}, if the size of c_i is less than L_s, then add s_{dj} to c_i. This first step does not consider the semantic representation of sentences. However, it is necessary to standardize each chunk to a minimum size to best leverage the capability of transformers since they have been trained on fixed size sequences.
2. Given s_{dj}, if the size of c_i is greater than L_s, and the addition of s_{dj} to c_i does not exceed U_s, we compute the semantic similarity between sentences (Alg. 2). Otherwise, we create a new chunk c_{i+1} and add s_{dj} to it.
3. To compute the similarity, Se3 first creates the sentence

[2] We could use domain-specific PLMs for other domains, e.g., SciBERT (Beltagy, Lo, and Cohan 2019) for scientific texts.

Algorithm 2: Semantic Similarity

```
Input: $model$; $s_{dj} \leftarrow$ current sentence; $c_i \leftarrow$ current chunk
Output: Put $s_{dj}$ in the best chunk
1: Let $c_i \leftarrow [s_{dj-x}, ..., s_{dj-1}]$
2: Let $c_{i+1} \leftarrow [s_{dj+1}, ..., s_{dj+y}]$
3: $enc\_c_i \leftarrow model.encode(c_i)$
4: $enc\_c_{i+1} \leftarrow model.encode(c\_i + 1)$
5: $score\_c_i \leftarrow mean(cosine\_sim(enc\_c_i, s_{dj}))$
6: $score\_c_{i+1} \leftarrow mean(cosine\_sim(enc\_c_{i+1}, s_{dj}))$
7: if $score\_c_i > score\_c_{i+1}$ then
8:     Put $s_j$ into $c_i$
9: else
10:    Put $s_j$ into $c_{i+1}$
11: end if
```

embeddings using the fine-tuned LEGAL-BERT. Afterward, the semantic similarity is calculated between s_{dj} and each sentence within c_i and c_{i+1}. Finally, the similarities are averaged per chunk and compared. In detail, c_{i+1} is created through a look-ahead. More precisely, we perform step 1 until the size of c_{i+1} is at least L_s. Thanks to such a look-ahead, the algorithm does not rely on any hyperparameter similarity threshold. For example, a sentence could be put into the chunk c_i if its semantic similarity with respect to c_i is greater than a fixed value. Instead, we compare the similarity score of the previous chunk with respect to the next one, obtaining an algorithm free from further hyperparameters.

4. Once the chunks have been created, we perform the target assignment (Alg. 3). Concretely, given s_{sk}, we compare it with each chunk and assign it to the chunk that maximizes the ROUGE-1 precision metric. We then discard chunks without targets at training time.

Abstractive Summarization

For experimental purposes, we use both a state-of-the-art quadratic and linear transformer. Their comparison is helpful to analyze how much an efficient transformer can be decisive to improve the summarization accuracy with a limited GPU memory. About the linear transformer, we choose Longformer-Encoder-Decoder (Beltagy, Peters, and Cohan 2020), namely LED, because it is the only efficient transformer with a base version public checkpoint. LED replaces the quadratic encoder self-attention using local window attention and global attention. Each token attends to itself and its neighbors in local attention, whereas the first token is connected to everything else in global attention, like in the full attention. About the quadratic transformer, we choose BART (Lewis et al. 2020) for the following reasons: i) there is a public checkpoint of the base version; ii) it is used as a checkpoint to initialize LED parameters because the latter follows the exact architecture of BART in terms of the number of layers and hidden sizes. The difference is that LED can read more tokens thanks to the linear attention mechanism, making it suitable for processing long documents. We choose the base versions for both models because the large ones do not fit into our GPU memory. For this reason, we make comparisons only with base models.

Algorithm 3: Target Assignment

```
Input: $summary\_sentences$; $chunks$
Output: The targets of the chunks
Let $targets = [t_1 = [], ..., t_w = []]$.
for $s_s$ in $summary\_sentences$ do
    Let $scores = []$.
    for $c$ in $chunks$ do
        $chunk\_score \leftarrow rouge\_precision(c, s_s)$
        $scores.append(chunk\_score)$
    end for
    $idx \leftarrow argmax(scores)$
    $targets[idx].append(s_s)$
end for
return $targets$
```

	AustLII		**BillSum**	
Statistic	Document	Summary	Document	Summary
# sentences	222	14	65	6
# words	7362	667	1592	197
# tokens	7983	722	1673	214
# docs	1754		22218	

Table 1: The dataset statistics. All values are mean over the dataset except for the "# docs" row. We used the LED tokenizer for tokens count and NLTK for words and sentences.

Experiments

Datasets

We used a dataset comprised of sentence triplets from Wikipedia articles (Ein-Dor et al. 2018) for metric learning. The 1.78M triplets are composed of a sentence pivot, one from the same section, and one from a different section.

We used two legal datasets of different countries (i.e., Australia and the United States) for abstractive summarization. *Australian Legal Case Reports*, referenced as AustLII and publicly downloadable from the UCI archive,[3] is a corpus of around 4000 legal cases from the Federal Court of Australia. We created a target for each document by using the catchphrases provided (i.e., the crucial statements of documents). In detail, we extracted every sentence containing the catchphrase, and we concatenated them to create the actual summary. Since not all documents have catchphrases, we collected 1754 documents, split into 1578 (90%) for training and 176 (10%) for testing. *BillSum* (Kornilova and Eidelman 2019), downloadable from the Hugging Face library and already split into 18,949 (≈85%) documents for training and 3,269 (≈15%) for testing,[4] consists of 22218 US Congressional Bills with human-written references.

The statistics of the datasets show that the AustLII documents are much longer than the BillSum ones (Table 1).

[3] https://archive.ics.uci.edu/ml/datasets/Legal+Case+Reports

[4] https://huggingface.co/datasets/billsum

System (*MaxLen*)	**AustLII** R1 / R2 / RL	**BillSum** R1 / R2 / RL
Baselines		
$\text{PEGASUS}_{\text{BASE}}$	-	51.42/29.68/37.78
$\text{BART}_{\text{BASE}}$ (*1024*)	33.51/23.92/27.88	54.42/35.81/41.98
$\text{BART}_{\text{BASE}}$ (*512*)	26.61/17.67/21.79	49.84/30.67/37.73
$\text{BART}_{\text{BASE}}$ (*256*)	23.87/13.98/18.80	45.99/26.36/34.12
$\text{BART}_{\text{BASE}}$ (*128*)	22.11/12.36/17.19	42.32/22.78/31.48
Baselines w/ Se3 - triplet		
$\text{BART}_{\text{BASE}}$ (*1024*)	**59.04/52.46/53.67**	57.31/37.85/43.78
$\text{BART}_{\text{BASE}}$ (*512*)	53.14/46.44/47.38	55.65/35.73/40.99
$\text{BART}_{\text{BASE}}$ (*256*)	44.55/36.50/37.05	51.99/32.63/37.11
$\text{BART}_{\text{BASE}}$ (*128*)	37.28/31.42/31.83	44.06/28.69/32.00
Baselines w/ Se3 - contrastive		
$\text{BART}_{\text{BASE}}$ (*1024*)	57.96/50.92/52.49	**57.66/38.20/44.11**
$\text{BART}_{\text{BASE}}$ (*512*)	52.66/45.71/46.66	55.96/35.82/41.27
$\text{BART}_{\text{BASE}}$ (*256*)	45.18/36.82/37.52	52.54/33.00/37.61
$\text{BART}_{\text{BASE}}$ (*128*)	37.54/31.89/32.27	44.29/28.90/32.27

Table 2: The results of BART with different chunk sizes. Best ROUGE scores are underlined for each max size, i.e., *1024*, *512*, *256*, *128*. The highest are bolded.

Experimental Settings

In order to thoroughly evaluate the performance of Se3 in low-resource regimes, the experiments were twofold.

Limited GPU memory issue. We experimented with six chunk size ranges, expressed in the number of tokens, by segmenting input documents based on the following sizes: *64-128*, *128-256*, *256-512*, *512-1024*, *1024-2048*, and *2048-4096*. About BART, we could not experiment with *1024-2048* and *2048-4096* since it was trained on short documents because of the quadratic memory complexity, so it truncates inputs longer than *1024* tokens. Further, to experiment with two versions of our method, we fine-tuned LEGAL-BERT with both losses, i.e., the triplet and the contrastive loss. To assess whether Se3 allows existing models to achieve a performance gain in low-resource regimes, we used BART and LED as baselines as they were designed, i.e., truncating the input according to each chunk max size without text segmentation. Thus, input sizes and memory requirements are the same, but the solutions with Se3 read the complete document details without truncation.

Labeled data scarcity issue. We fine-tuned both models combined with Se3 with 10 and 100 labeled training instances. We experimented only on the BillSum dataset to compare our results with recent works on the same low-resource summarization task.

Training Details

We trained LEGAL-BERT for 1 epoch for metric learning using a batch size of 8 and a learning rate set to 2×10^{-5}. About abstractive summarization, we trained BART and LED for all experiments using the Hugging Face library. All models are fine-tuned for 5 epochs using a batch size of 1 and a learning rate with a linear schedule set to 5×10^{-5}. At inference time, we used 2 as beam size and length penalty.

System (*MaxLen*)	**AustLII** R1 / R2 / RL	**BillSum** R1 / R2 / RL
Baselines		
$\text{PEGASUS}_{\text{BASE}}$	-	51.42/29.68/37.78
LED_{BASE} (*4096*)	50.27/39.85/42.04	58.83/39.83/45.71
LED_{BASE} (*2048*)	42.76/32.20/35.71	58.38/39.37/45.09
LED_{BASE} (*1024*)	35.20/24.62/28.38	55.32/36.48/42.67
LED_{BASE} (*512*)	30.47/18.90/23.56	49.96/30.76/37.68
LED_{BASE} (*256*)	26.77/15.37/20.39	46.76/26.54/34.44
LED_{BASE} (*128*)	23.78/12.58/18.12	42.75/22.97/31.70
Baselines w/ Se3 - triplet		
LED_{BASE} (*4096*)	57.89/48.96/50.28	58.51/39.71/45.66
LED_{BASE} (*2048*)	**60.03/53.03/54.57**	58.38/39.53/45.48
LED_{BASE} (*1024*)	58.48/52.17/53.48	57.88/38.38/44.15
LED_{BASE} (*512*)	54.25/47.33/48.32	55.61/35.87/41.04
LED_{BASE} (*256*)	45.27/36.88/37.68	51.79/32.74/37.09
LED_{BASE} (*128*)	37.36/31.60/32.09	43.72/28.72/31.88
Baselines w/ Se3 - contrastive		
LED_{BASE} (*4096*)	57.82/49.06/50.50	**59.18/40.18/46.04**
LED_{BASE} (*2048*)	60.20/52.40/53.79	58.63/39.77/45.60
LED_{BASE} (*1024*)	58.75/52.28/53.71	58.11/38.61/44.52
LED_{BASE} (*512*)	52.37/45.63/46.54	55.99/36.09/41.40
LED_{BASE} (*256*)	45.35/36.80/37.51	52.28/33.00/37.44
LED_{BASE} (*128*)	38.07/32.20/32.67	43.74/28.78/31.95

Table 3: The results of LED with several chunk sizes. Best ROUGE scores are underlined for each size, i.e., *4096*, *2048*, *1024*, *512*, *256*, *128*. The highest are bolded.

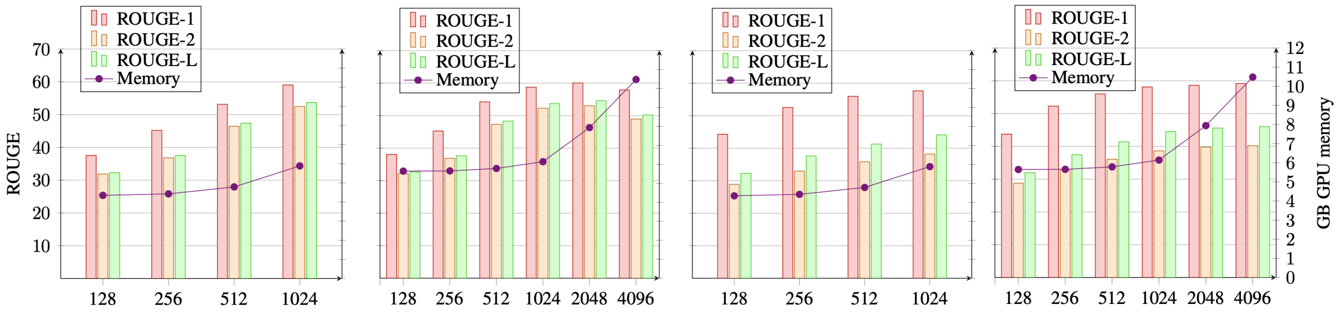
Results with Input Longer Than the GPU Memory

Table 2 and Table 3 summarize BART and LED evaluation results with different chunk sizes on both datasets.

Model performance comparisons. Solutions with Se3 significantly perform the best. In particular, our solution is more effective for the AustLII documents because they are very long, leading to a consistent boost in performance. Indeed, the baselines truncate the input, discarding valuable information in the final summary. Comparison of models shows no performance differences for short inputs. Instead, LED can process longer sequences thanks to the linear complexity of its encoder self-attention, obtaining better results than BART, which can read inputs up to *1024* tokens in size.

Ranking loss comparisons. The contrastive loss is better than the triplet loss when used for the BillSum dataset, differently from the AustLII documents. These results prove that performance mainly depends on the legal content.

Chunk memory requirement comparisons. The bigger the chunks, the higher the scores. This result is motivated by the better capability of transformers to process longer sequences. Further, to visualize the scalability of Se3, Fig. 3 shows the trade-off between the GPU memory used and the model accuracy. The results point out that the best trade-off for both models is *1024* as the max chunk size. LED is trained with a local attention window of *1024* tokens, so it pads inputs if shorter. Therefore, the memory requirements no longer decrease proportionally below this threshold.



(a) AustLII Se3-BART (b) AustLII Se3-LED (c) BillSum Se3-BART (d) BillSum Se3-LED

Figure 3: The trade-off between performance and memory requirements of Se3 for each max chunk size on both datasets.

System (*MaxLen*)	**BillSum (10)** R1 / R2 / RL	**BillSum (100)** R1 / R2 / RL
Baselines		
MTL-ABS	41.22/18.61/26.33	45.29/22.74/29.56
PEGASUS$_{\text{LARGE}}$	40.48/18.49/27.27	44.78/26.40/**34.40**
BART$_{\text{BASE}}$	39.58/18.94/26.63	44.66/24.87/31.09
LED$_{\text{BASE}}$	41.10/21.15/27.93	47.68/26.98/32.43
Solutions w/ Se3		
BART$_{\text{BASE}}$ (*1024*)	44.37/21.17/27.57	47.85/26.67/33.36
BART$_{\text{BASE}}$ (*512*)	46.58/22.03/28.23	49.88/26.84/33.33
BART$_{\text{BASE}}$ (*256*)	46.50/23.24/28.54	48.17/26.55/31.51
BART$_{\text{BASE}}$ (*128*)	41.48/22.73/26.37	42.42/25.42/28.98
LED$_{\text{BASE}}$ (*4096*)	38.48/19.26/26.36	48.11/26.44/31.91
LED$_{\text{BASE}}$ (*2048*)	42.35/20.70/27.12	47.71/26.33/32.12
LED$_{\text{BASE}}$ (*1024*)	45.32/22.67/29.12	48.28/26.97/33.46
LED$_{\text{BASE}}$ (*512*)	**46.94**/23.04/**29.29**	**50.45/27.73**/33.74
LED$_{\text{BASE}}$ (*256*)	46.22/**24.32**/29.16	48.13/27.16/31.89
LED$_{\text{BASE}}$ (*128*)	40.14/22.76/26.05	40.93/25.29/28.55

Table 4: Labeled data scarcity summarization on BillSum with 10 and 100 training instances. Best values are bolded.

Results on Labeled Data Scarcity

Table 4 shows the performance of labeled data scarcity summarization. We used the first 10 and 100 labeled instances of BillSum as done by Zhang et al. (2020a) and Chen and Shuai (2021) with PEGASUS and MTL-ABS, respectively. Our method significantly improves the performance, proving the importance of creating high-correlated source-target pairs in low-resource settings. In detail, the smaller the chunks, the greater the labeled data, allowing transformers to train on more instances. Indeed, we achieved baseline-like results even with models trained on chunk sizes of *64-128* tokens.

Ablation Studies

We conducted additional experiments with LED on chunks of *512-1024* tokens, reporting the performance of Se3 after removing 1) metric learning, 2) legal language modeling, and 3) sentence semantic representation (Table 5). We notice that excluding either thematic metric learning, legal language modeling, or sentence semantic representation leads to a performance drop. Training the model without considering the semantic meaning of the sentences in the text segmentation (i.e., ignoring Alg. 2) leads to the most significant decrease in performance, showing the importance of our semantic self-segmentation algorithm. Indeed, using the sentence representation from BERT improves the accuracy because, without Se3, sentences semantically closer could be split into different chunks, worsening the final summarization. Removing the legal language modeling and the thematic metric learning also decreases the model performance, proving that a domain-specific language model trained on thematic similarity helps create better aligned chunk-target pairs, improving the summarization. In particular, we report the content coverage between chunk-target pairs at training and test time, computed with the average ROUGE-1 precision (R1-P). Results show better alignments using Se3, confirming our method contribution for creating new small high-correlated instances essential for training abstractive summarization transformers in low-resource regimes.

Summaries Accuracy

To evaluate the accuracy of the predicted summaries to not rely only on ROUGE, we first used BERTScore (Zhang et al. 2020b) for semantic assessment. Second, we investigated the redundancy due to the independent chunk processing and the final concatenation. To this end, we used the same approaches in Xiao and Carenini (2020). We first used a *Unique n-gram ratio* to measure n-grams uniqueness. The lower the score, the more redundant the document.

$$Uniq\_ngram\_ratio = \frac{count(uniq\_n\_gram)}{count(n\_gram)} \quad (3)$$

Second, we used the *Normalized Inverse of Diversity (NID)* to capture redundancy by normalizing the unigrams entropy in the document with the maximum possible entropy. The higher the score, the more redundant the document.

$$NID = 1 - \frac{entropy(D)}{log(|D|)} \quad (4)$$

Table 6 shows the results using LED. BERTScore reports higher results of Se3 on AustLII and similar scores on BillSum. Differently, we notice a decrease of n-gram uniqueness with our solution, which is a symbol of more redundancy. Instead, NID scores do not capture such differences.

	AustLII		BillSum	
Approach	R1 / R2 / RL	R1-P (Train/Test)	R1 / R2 / RL	R1-P (Train/Test)
Se3 (Full)	**58.75/52.28/53.71**	**98.39/98.12**	**58.11/38.61/44.52**	**88.61/88.88**
w/o metric learning	57.11/50.18/51.26	92.87/92.45	57.94/38.44/44.36	88.27/88.44
w/o legal language modeling	56.66/48.95/50.20	98.39/97.97	57.85/38.32/44.03	88.47/88.68
w/o sentence semantics	55.38/47.66/49.05	93.15/93.02	56.65/37.27/43.16	85.62/85.96

Table 5: The ablations to study how each module of our method contributes to the performance gain. We gradually removed each component of our solution to show the performance drop. Best values are bolded.

	AustLII					BillSum				
System (*MaxLen*)	BERTScore	Uni%	Bi%	Tri%	NID	BERTScore	Uni%	Bi%	Tri%	NID
Reference										
Source	-	22.50	62.21	82.90	28.09	-	25.83	57.82	73.98	30.04
Target	-	51.20	82.37	91.53	23.78	-	57.37	88.33	94.97	21.20
Baselines										
LED_{BASE} (*4096*)	88.59	58.58	91.21	99.68	21.54	90.26	58.76	92.84	99.88	20.77
LED_{BASE} (*2048*)	87.53	60.96	92.23	99.77	21.71	90.20	59.21	92.94	99.87	**20.76**
LED_{BASE} (*1024*)	86.29	60.92	91.75	99.69	22.49	89.82	61.01	93.46	99.88	20.92
LED_{BASE} (*512*)	84.92	59.88	90.41	99.72	23.47	88.93	62.34	93.70	**99.90**	21.38
LED_{BASE} (*256*)	84.26	63.14	92.28	99.79	22.96	88.21	62.36	93.39	99.89	22.08
LED_{BASE} (*128*)	83.46	**67.29**	**94.18**	**99.81**	22.43	87.49	**64.42**	**94.27**	99.88	22.15
Baselines w/ Se3										
LED_{BASE} (*4096*)	89.45	51.59	88.26	97.86	**21.54**	**90.30**	59.00	92.89	99.86	20.77
LED_{BASE} (*2048*)	**89.75**	48.68	86.33	96.78	21.74	90.16	58.87	92.09	98.94	20.84
LED_{BASE} (*1024*)	89.42	44.68	84.00	95.58	21.94	89.79	55.49	89.10	96.35	21.51
LED_{BASE} (*512*)	88.04	41.47	81.20	93.90	22.61	89.04	50.86	85.03	93.47	22.49
LED_{BASE} (*256*)	86.10	39.39	79.62	92.87	23.63	88.11	45.77	80.64	90.54	23.62
LED_{BASE} (*128*)	85.00	33.19	74.16	89.99	24.25	87.12	38.44	74.29	86.86	25.11

Table 6: The evaluation of the predicted summaries with BERTScore, uni-gram, bi-gram, and trigram uniqueness, and NID. We also provide the values of the reference documents. Best scores are bolded.

Conclusion

In this work, we introduced Se3[5] to address the abstractive long document summarization under low-resource regimes, namely with low-memory GPUs and labeled data scarcity, where the accuracy of existing approaches drops. Thanks to Se3, summarization transformers can process all document details without truncation, achieving also new state-of-the-art results in low-resource scenarios with base models. Moreover, we proved that the method generates semantically accurate summaries in legal datasets, hence it can be successfully applied to other less complex domains.

We envisage further directions to deal with inputs longer than the GPU memory allows: i) training models to self-annotate cross-chunk salient information through memory-based neural layers (Moro et al. 2018; Cui and Hu 2021); ii) extracting relevant texts with term weighting techniques (Domeniconi et al. 2015b) and inter-chunk semantic relations with unsupervised methods (Domeniconi et al. 2014a, 2016b,c) to model interpretable representations with knowledge graph (Frisoni and Moro 2021; Frisoni, Moro, and Carbonaro 2020) or relation and event extraction (Domeniconi et al. 2016a; Frisoni, Moro, and Carbonaro 2021).

Broader Impact and Ethical Statement

Summarizing long documents can benefit from our solution, even in small organizations with minimal resources. However, because of the high societal impact of legislation and biases in PLMs (Dey et al. 2020; Liang et al. 2021), domain experts should guide the usage of our method to validate the quality of the inferred summaries. Finally, if the method will be applied to sensitive data, users should also deal with privacy-preserving policies (da Silva et al. 2006).

Acknowledgments

We thank the Maggioli Group[6] for granting the Ph.D. scholarship to Luca Ragazzi. We also thank the anonymous reviewers and Giacomo Frisoni and Lorenzo Valgimigli (DISI, University of Bologna) for suggesting corrections.

[5] https://www.unibo.it/sitoweb/gianluca.moro/en

[6] In particular, Manlio Maggioli, Paolo Maggioli, Cristina Maggioli, Amalia Maggioli, Nicoletta Belardinelli, and Andrea Montefiori. https://www.maggioli.com/who-we-are/company-profile

References

Ahmed, N.; and Wahed, M. 2020. The De-democratization of AI: Deep Learning and the Compute Divide in Artificial Intelligence Research. *CoRR*, abs/2010.15581.

Anand, D.; and Wagh, R. 2019. Effective deep learning approaches for summarization of legal texts. *Journal of King Saud University - Computer and Information Sciences*.

Bajaj, A.; Dangati, P.; Krishna, K.; Kumar, P. A.; Uppaal, R.; Windsor, B.; Brenner, E.; Dotterrer, D.; Das, R.; and McCallum, A. 2021. Long Document Summarization in a Low Resource Setting using Pretrained Language Models. In *ACL-IJCNLP 2021, July 5-10*, 71–80. ACL.

Beltagy, I.; Lo, K.; and Cohan, A. 2019. SciBERT: A Pretrained Language Model for Scientific Text. In *EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, 3613–3618. Association for Computational Linguistics.

Beltagy, I.; Peters, M. E.; and Cohan, A. 2020. Longformer: The Long-Document Transformer. *CoRR*, abs/2004.05150.

Çelikyilmaz, A.; Bosselut, A.; He, X.; and Choi, Y. 2018. Deep Communicating Agents for Abstractive Summarization. In *NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6*, 1662–1675. ACL.

Chalkidis, I.; Fergadiotis, M.; Malakasiotis, P.; Aletras, N.; and Androutsopoulos, I. 2020. LEGAL-BERT: The Muppets straight out of Law School. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2898–2904. Online: Association for Computational Linguistics.

Chen, Y.; and Shuai, H. 2021. Meta-Transfer Learning for Low-Resource Abstractive Summarization. In *AAAI-IAAI-EAAI 2021, February 2-9*, 12692–12700. AAAI Press.

Choromanski, K. M.; Likhosherstov, V.; Dohan, D.; Song, X.; Gane, A.; Sarlós, T.; Hawkins, P.; Davis, J. Q.; Mohiuddin, A.; Kaiser, L.; Belanger, D. B.; Colwell, L. J.; and Weller, A. 2021. Rethinking Attention with Performers. In *ICLR 2021, Austria, May 3-7, 2021*. OpenReview.net.

Cui, P.; and Hu, L. 2021. Sliding Selector Network with Dynamic Memory for Extractive Summarization of Long Documents. In *NAACL-HLT 2021, June 6-11*, 5881–5891. ACL.

da Silva, J. C.; Klusch, M.; Lodi, S.; and Moro, G. 2006. Privacy-preserving agent-based distributed data clustering. *Web Intell. Agent Syst.*, 4(2): 221–238.

de Vargas Feijó, D.; and Moreira, V. P. 2019. Summarizing Legal Rulings: Comparative Experiments. In *RANLP 2019, Varna, Bulgaria, September 2-4*, 313–322. INCOMA Ltd.

Dey, A.; Chowdhury, T.; Atri, Y. K.; and Chakraborty, T. 2020. Corpora Evaluation and System Bias detection in Multi Document Summarization. In *EMNLP 2020, Online Event, 16-20 November 2020*, 2830–2840. ACL.

Domeniconi, G.; Masseroli, M.; Moro, G.; and Pinoli, P. 2014a. Discovering New Gene Functionalities from Random Perturbations of Known Gene Ontological Annotations. In *KDIR 2014, Rome, Italy, 21-24 October*, 107–116. SciTePress.

Domeniconi, G.; Masseroli, M.; Moro, G.; and Pinoli, P. 2016a. Cross-organism learning method to discover new gene functionalities. *Computer Methods and Programs in Biomedicine*, 126: 20–34.

Domeniconi, G.; Moro, G.; Pagliarani, A.; Pasini, K.; and Pasolini, R. 2016b. Job Recommendation from Semantic Similarity of LinkedIn Users' Skills. In *ICPRAM 2016*, 270–277. SciTePress.

Domeniconi, G.; Moro, G.; Pagliarani, A.; and Pasolini, R. 2015a. Markov Chain based Method for In-Domain and Cross-Domain Sentiment Classification. In *KDIR 2015 - Proc. of IC3K, Volume 1, Lisbon, Portugal, November 12-14, 2015*, 127–137. SciTePress.

Domeniconi, G.; Moro, G.; Pasolini, R.; and Sartori, C. 2014b. Cross-domain Text Classification through Iterative Refining of Target Categories Representations. In *KDIR 2014, Rome, Italy, 21 - 24 October, 2014*, 31–42. SciTePress.

Domeniconi, G.; Moro, G.; Pasolini, R.; and Sartori, C. 2014c. Iterative Refining of Category Profiles for Nearest Centroid Cross-Domain Text Classification. In *IC3K 2014, Rome, Italy, October 21-24*, volume 553, 50–67. Springer.

Domeniconi, G.; Moro, G.; Pasolini, R.; and Sartori, C. 2015b. A Comparison of Term Weighting Schemes for Text Classification and Sentiment Analysis with a Supervised Variant of tf.idf. In *DATA*, volume 584, 39–58. Springer.

Domeniconi, G.; Semertzidis, K.; López, V.; Daly, E. M.; Kotoulas, S.; and Moro, G. 2016c. A Novel Method for Unsupervised and Supervised Conversational Message Thread Detection. In *DATA 2016, Lisbon, Portugal, 24-26 July, 2016*, 43–54. SciTePress.

Ein-Dor, L.; Mass, Y.; Halfon, A.; Venezian, E.; Shnayderman, I.; Aharonov, R.; and Slonim, N. 2018. Learning Thematic Similarity Metric from Article Sections Using Triplet Networks. In *ACL 18, Melbourne, July 15-20*, 49–54. ACL.

Frisoni, G.; and Moro, G. 2021. Phenomena Explanation from Text: Unsupervised Learning of Interpretable and Statistically Significant Knowledge. In *Data Management Technologies and Applications*, 293–318. Cham: Springer International Publishing.

Frisoni, G.; Moro, G.; and Carbonaro, A. 2020. Unsupervised Descriptive Text Mining for Knowledge Graph Learning. In *IC3K 2020, Volume 1: KDIR, Budapest, Hungary, November 2-4, 2020*, 316–324. SCITEPRESS.

Frisoni, G.; Moro, G.; and Carbonaro, A. 2021. A Survey on Event Extraction for Natural Language Understanding: Riding the Biomedical Literature Wave. *IEEE Access*, 9: 160721–160757.

Galgani, F.; Compton, P.; and Hoffmann, A. G. 2015. Summarization based on bi-directional citation analysis. *Inf. Process. Manag.*, 51(1): 1–24.

Gidiotis, A.; and Tsoumakas, G. 2020. A Divide-and-Conquer Approach to the Summarization of Long Documents. *IEEE ACM*, 28: 3029–3040.

Grail, Q.; Perez, J.; and Gaussier, É. 2021. Globalizing BERT-based Transformer Architectures for Long Document Summarization. In *EACL 21, April 19-23*, 1792–1810. ACL.

Guo, M.; Ainslie, J.; Uthus, D. C.; Ontañón, S.; Ni, J.; Sung, Y.; and Yang, Y. 2021. LongT5: Efficient Text-To-Text Transformer for Long Sequences. *CoRR*, abs/2112.07916.

Huang, L.; Cao, S.; Parulian, N. N.; Ji, H.; and Wang, L. 2021. Efficient Attentions for Long Document Summarization. In *NAACL-HLT 2021, June 6-11*, 1419–1436. ACL.

Huang, Y.; Yu, Z.; Guo, J.; Yu, Z.; and Xian, Y. 2020. Legal public opinion news abstractive summarization by incorporating topic information. *Int. J. Mach. Learn. Cybern.*, 11(9): 2039–2050.

Jain, D.; Borah, M. D.; and Biswas, A. 2021a. Automatic Summarization of Legal Bills: A Comparative Analysis of Classical Extractive Approaches. In *ICCCIS 2021*, 394–400.

Jain, D.; Borah, M. D.; and Biswas, A. 2021b. Summarization of legal documents: Where are we now and the way forward. *Comput. Sci. Rev.*, 40: 100388.

Kanapala, A.; Pal, S.; and Pamula, R. 2019. Text summarization from legal documents: a survey. *Artif. Intell. Rev.*, 51(3): 371–402.

Kitaev, N.; Kaiser, L.; and Levskaya, A. 2020. Reformer: The Efficient Transformer. In *ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Kornilova, A.; and Eidelman, V. 2019. BillSum: A Corpus for Automatic Summarization of US Legislation. In *Proc. of the 2nd Workshop on New Frontiers in Summarization*, 48–56. Hong Kong, China: ACL.

Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; and Zettlemoyer, L. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *ACL 2020, July 5-10*, 7871–7880. ACL.

Liang, P. P.; Wu, C.; Morency, L.; and Salakhutdinov, R. 2021. Towards Understanding and Mitigating Social Biases in Language Models. In *Proc. of the 38th ICML 2021, 18-24 July 2021, Virtual Event*, volume 139, 6565–6576. PMLR.

Lin, C.-Y. 2004. ROUGE: a Package for Automatic Evaluation of Summaries. In *ACL 2004, Barcelona, Spain*.

Liu, Z.; and Chen, N. 2019. Exploiting discourse-level segmentation for extractive summarization. In *Proc. of the 2nd Workshop on New Frontiers in Summarization*, 116–121.

Magooda, A.; and Litman, D. J. 2020. Abstractive Summarization for Low Resource Data Using Domain Transfer and Data Synthesis. In *Proceedings of the Thirty-Third International Florida Artificial Intelligence Research Society Conference, May 17-20, 2020*, 240–245. AAAI Press.

Manakul, P.; and Gales, M. J. F. 2021. Long-Span Summarization via Local Attention and Content Selection. In *ACL/IJCNLP 2021, August 1-6, 2021*, 6026–6041. ACL.

Moro, G.; Pagliarani, A.; Pasolini, R.; and Sartori, C. 2018. Cross-domain & In-domain Sentiment Analysis with Memory-based Deep Neural Networks. In *IC3K 2018, Seville, Spain, September 18-20*, 125–136. SciTePress.

Moro, G.; Ragazzi, L.; Valgimigli, L.; and Freddi, D. 2022. Discriminative Marginalized Probabilistic Neural Method for Multi-Document Summarization of Medical Literature. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, ACL 2022*. ACL.

Moro, G.; and Valgimigli, L. 2021. Efficient Self-Supervised Metric Information Retrieval: A Bibliography Based Method Applied to COVID Literature. *Sensors*, 21(19).

Parida, S.; and Motlícek, P. 2019. Abstract Text Summarization: A Low Resource Challenge. In *EMNLP 2019, Hong Kong, China, November 3-7, 2019*, 5993–5997. ACL.

Qi, W.; Yan, Y.; Gong, Y.; Liu, D.; Duan, N.; Chen, J.; Zhang, R.; and Zhou, M. 2020. ProphetNet: Predicting Future N-gram for Sequence-to-Sequence Pre-training. In *EMNLP 2020, 16-20 November 2020*, 2401–2410. ACL.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *J. Mach. Learn. Res.*, 21: 140:1–140:67.

Rohde, T.; Wu, X.; and Liu, Y. 2021. Hierarchical Learning for Generation with Long Source Sequences. *CoRR*, abs/2104.07545.

Sadvilkar, N.; and Neumann, M. 2020. PySBD: Pragmatic Sentence Boundary Disambiguation. In *Proceedings of Second Workshop for (NLP-OSS)*, 110–114. Online: ACL.

Sharir, O.; Peleg, B.; and Shoham, Y. 2020. The Cost of Training NLP Models: A Concise Overview. *CoRR*, abs/2004.08900.

Tran, V. D.; Nguyen, M. L.; and Satoh, K. 2018. Automatic Catchphrase Extraction from Legal Case Documents via Scoring using Deep Neural Networks. *CoRR*, abs/1809.05219.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All you Need. In *NIPS 2017, 4-9 December 2017, Long Beach, CA, USA*, 5998–6008.

Xiao, W.; and Carenini, G. 2020. Systematically Exploring Redundancy Reduction in Summarizing Long Documents. In *AACL/IJCNLP 2020, Suzhou, China, December 4-7*, 516–528. Association for Computational Linguistics.

Xiong, Y.; Zeng, Z.; Chakraborty, R.; Tan, M.; Fung, G.; Li, Y.; and Singh, V. 2021. Nyströmformer: A Nyström-based Algorithm for Approximating Self-Attention. In *AAAI-IAAI-EAAI 2021, February 2-9*, 14138–14148. AAAI Press.

Xu, J.; Gan, Z.; Cheng, Y.; and Liu, J. 2020. Discourse-Aware Neural Extractive Text Summarization. In *ACL 2020, Online, July 5-10*, 5021–5031. ACL.

Yu, T.; Liu, Z.; and Fung, P. 2021. AdaptSum: Towards Low-Resource Domain Adaptation for Abstractive Summarization. In *NAACL-HLT 2021, Online, June 6-11, 2021*, 5892–5904. Association for Computational Linguistics.

Zaheer, M.; Guruganesh, G.; Dubey, K. A.; Ainslie, J.; Alberti, C.; Ontañón, S.; Pham, P.; Ravula, A.; Wang, Q.; Yang, L.; and Ahmed, A. 2020. Big Bird: Transformers for Longer Sequences. In *NeurIPS 2020, December 6-12, 2020, virtual*.

Zhang, J.; Zhao, Y.; Saleh, M.; and Liu, P. J. 2020a. PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization. In *ICML 2020, 13-18 July 2020, Virtual Event*, volume 119, 11328–11339. PMLR.

Zhang, T.; Kishore, V.; Wu, F.; Weinberger, K. Q.; and Artzi, Y. 2020b. BERTScore: Evaluating Text Generation with BERT. In *ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.