Semantic Self-segmentation for Abstractive Summarization of Long Legal Documents in Low-resource Regimes

Gianluca Moro, Luca Ragazzi
Department of Computer Science and Engineering, University of Bologna, Cesena Campus
Via dell’Università 50, I-47522 Cesena, Italy
{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, such loss is generally destructive for semantic text analytics in the legal domain. 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 longer inputs 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 that our 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. Finally, we perform ablation studies to assess how the different components of our method contribute to the performance gain.1

1 Introduction

State-of-the-art solutions on abstractive summarization are built upon Transformer (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; Zahed et al. 2020; Huang et al. 2021; Choromanski et al. 2021; Xiong et al. 2021). However, training large Transformers requires high-resource settings (Sharir, Peleg, and Shoham 2020; Ahmed and Wahed 2020), leaving the summarization of long documents an open research problem in low-resource regimes with limited GPU memories and only dozens of labeled training data.

Legal analytics typically tackles low-resource settings of labeled instances, 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 document details (Kanapala, Pal, and Pamula 2019).

Input truncation, unavoidable for long sequences with a low-memory GPU, ignores valuable information, destroying the final summary semantic. To address this problem, particularly relevant in the legal domain, we propose a semantic self-segmentation (Se3) approach for long document summarization. Se3 creates high-correlated source-target pairs by segmenting long texts into semantically coherent chunks that fit into the GPU memory and pairing them with the most similar summary part, enabling Transformers to process all document details without truncation. This approach also works as a data augmentation strategy to cope with the typical lack of labeled training instances in low-resource settings, usually addressed with transfer learning techniques (Domeniconi et al. 2014, 2017). As far as we know, this is the first study on long document summarization with both limited GPU memories and labeled data scarcity.

In order to evaluate our method, we experiment on two legal datasets of different sizes and content lengths. All studies have been performed with one Titan Xp GPU of 12GB memory, using Se3 combined with BART (Lewis et al. 2020) and LED (Beltagy, Peters, and Cohan 2020). Results show that our approach significantly improves the performance of abstractive summarization Transformers, even with as few as 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.

To sum up, the paper contributions are the following:

1. We propose Se3 to successfully address long document summarization in low-resource regimes, namely limited GPU memories and labeled data scarcity, allowing very long documents to be summarized without truncation by summarizing each chunk and concatenating the results.

2. We advance the research on abstractive summarization in the legal domain, achieving new state-of-the-art results on two datasets using a single GPU of 12GB memory.

Copyright © 2022, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

1The solution is available at https://se3-unibo.github.io
2 Related Work

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

Long document summarization. Although most solu- tions focus on short inputs because of the quadratic comp- lexity of Transformers, several works presented new ap- proaches 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 ex- tract the salient content for extractive summarization. Gidi- otis and Tsoumakas (2020) introduced a divide-and-conquer approach that relies on structured documents to summa- rize each section independently. Bajaj et al. (2021) com- pressed long texts by extracting the sentences that best cor- relates 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) pro- posed an extractive model that can summarize inputs of ar- bitrary size without truncation by using a memory network.

Low-resource summarization. About low-resource stud- ies, prior works have only focused on data scarcity. Parida and Motlicek (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 sum- marization 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 in- vestigate 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 the other works, we propose a new ap- proach for the abstractive summarization of long documents to address low-resource regimes issues, namely limited GPU memories and labeled data scarcity, by semantically seg- menting long inputs into GPU memory-adaptable chunks.

Figure 1: The overview of Se3 for the abstractive summa- rization 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 sen- tence, i.e., orange rectangles, is assigned to the most corre- lated chunk to create source-target pairs to train models (red phase). Then, each chunk is summarized independently (yellow phase). Finally, the intermediate predictions are com- bined to obtain the final summary (gray phase).

3 Method

Our semantic self-segmentation (Se3) approach for abstrac- tive long document summarization allows fine-tuning Trans- formers on entire long inputs without truncation with lim- ited GPUs. Concretely, our method segments long texts into content-wise chunks and assigns them the most corre- lated summary part (Fig. 1). Therefore, Se3 augments the training data since each chunk is treated as a training instance. Two observations motivate this solution: (1) Truncating input to a fixed length may discard valuable information. (2) In a low- resource scenario, there may also be a lack of labeled data to fine-tune pre-trained language models effectively.

To sum up, Se3 has the following features:

- **Structure-independent**: it can be applied to any long doc- ument because it does not rely on textual characteristics.
- **Thematic-focused**: each chunk represents a semantic unit expressed in the text where each sentence shares an infor- mative topic with the others. Moreover, the sources are highly related to the targets, allowing models to focus on a specific document theme during training.
- **Data and memory-adaptable**: it is possible to change the size of the chunks according to the available computa- tional resources, enabling models fine-tuning with a lim- ited GPU memory thanks to the short input sizes. Further, training data augment because each chunk is treated as an individual training instance.
Algorithm 1: Semantic Self-segmentation

Input: $model \leftarrow$ LEGAL-BERT; $doc\_sent \leftarrow [s_{d0}, \ldots, s_{dn}]$; $summary\_sent \leftarrow [s_{0}, \ldots, s_{m}]$

Parameters: $L_s \leftarrow$ lower size; $U_s \leftarrow$ upper size

Output: Return the chunk-target pairs

1: Let $chunks = []$
2: Let $current\_chunk = []$
3: for $s_d$ in $doc\_sent$ do
4:   if $\text{len}(current\_chunk) + \text{len}(s_d) < L_s$ then
5:     $current\_chunk.append(s_d)$
6:   else if $\text{len}(current\_chunk) + \text{len}(s_d) > U_s$ then
7:     $chunks.append(current\_chunk)$
8:     $current\_chunk \leftarrow []$
9:   else
10:     Perform the Semantic Similarity (Alg. 2)
11:     end if
12: end for
13: targets $\leftarrow$ Perform the Target Assignment (Alg. 3)
14: return $(chunks, targets)$

Semantic Self-segmentation

In order to train Transformers to summarize very long inputs without truncation with a limited GPU memory, our text segmentation algorithm needs three elements.

The chunk size is needed to standardize its content within a range since pre-trained Transformers have been trained on fixed sizes, so they struggle to process chunks of very different sizes. Further, such a range helps to change input size, adapting chunks to the GPU memory available and best leveraging the capability of Transformers.

A language model is needed to represent the sentences semantically. Therefore, we test Se3 on legal documents, we use LEGAL-BERT (Chalkidis et al. 2020), a BERT model pre-trained on legal corpora. Further, we fine-tune LEGAL-BERT on a metric learning task to learn if two sentences belong to the same section. Such learning trains the model to enrich the sentence representation with the thematic part, essential for our text segmentation to split sentences based on their thematic meaning. Of course, for different domains, we have to use domain-specific language models, e.g., SciBERT (Beltagy, Lo, and Cohan 2019) for scientific texts. The metric learning uses a public dataset created with a self-supervised approach (Ein-Dor et al. 2018), as done with papers bibliography 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 $m$:

$$loss = \max(||x - x^+|| - ||x - x^-|| + m, 0)$$ (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:

$$loss = l \times ||x - y|| + (1 - l) \times \max(m - ||x - y||, 0)$$ (2)

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 where a summary sentence is within a chunk. For this reason, we assign the most similar part of the actual summary of the source document, searching the chunk where a summary sentence can be better summarized. 2) The precision metric scores how much content of a summary sentence is within a chunk, searching for the best content coverage.

Algorithm

Let $s_{d0}, s_{d1}, \ldots, s_{dn}$ be the sentences of a document $D$ obtained using the state-of-the-art tokenizer PySBD (Sadvilkar and Neumann 2020). Let $s_0, s_1, \ldots, s_m$ 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 $c_t$, along with its target $t_s$, Se3 performs the following steps (Alg. 1):

1. Given $s_{dj}$, if the size of $c_t$ is less than $L_s$, then add $s_{dj}$ to $c_t$. 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 $c_t$ is greater than $L_s$, and the addition of $s_{dj}$ to $c_t$ does not exceed $U_s$, we compute the semantic similarity between sentences (Alg. 2). Otherwise, we create a new chunk $c_{t+1}$ and add $s_{dj}$ to it.
To compute the similarity, Se3 first creates the sentence embeddings using the fine-tuned LEGAL-BERT. Afterwards, 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 lookahead. 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 compute the similarity score of the previous chunk with respect to the next one, obtaining an algorithm free from further hyperparameters.

Once the chunks have been created, we perform the target assignment (Alg. 3). Concretely, given $s_k$, 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

#### Algorithm 2: Semantic Similarity

**Input:** $s_{dj}$ ← Current sentence, $c_i$ ← Current chunk  
model ← LEGAL-BERT  
**Output:** Put $s_{dj}$ into the correct chunk  
1: Let $c_i ← [s_{dj−y}, ..., s_{dj−1}]$  
2: Let $c_{i+1} ← [s_{dj+1}, ..., s_{dj+y}]$  
3: $\text{enc}_{c_i} ← \text{model.encode}(c_i)$  
4: $\text{enc}_{c_{i+1}} ← \text{model.encode}(c_{i+1})$  
5: $\text{score}, c_i ← \text{mean}(\text{cosine.sim}(\text{enc}_{c_i}, s_{dj}))$  
6: $\text{score}, c_{i+1} ← \text{mean}(\text{cosine.sim}(\text{enc}_{c_{i+1}}, s_{dj}))$  
7: if $\text{score}, c_i > \text{score}, c_{i+1}$ then  
8: Put $s_j$ into $c_i$  
9: else  
10: Put $s_j$ into $c_{i+1}$  
11: end if

#### Algorithm 3: Target Assignment

**Input:** sentences ← [$s_{s0}, ..., s_{sm}$], chunks ← [$c_0, ..., c_w$]  
**Output:** Return the targets of the chunks  
1: Let $\text{targets} ← [t_0 = [], ..., t_w = []]$  
2: for $s_i$ in sentences do  
3: Let $\text{scores} = []$  
4: for $c$ in chunks do  
5: $\text{chunk} \_\text{score} ← \text{rouge} \_\text{precision}(c, s_i)$  
6: $\text{scores}.\text{append}(\text{chunk} \_\text{score})$  
7: end for  
8: $\text{id}x ← \text{argmax}(\text{scores})$  
9: $\text{targets}[\text{id}x].\text{append}(s_i)$  
10: end for  
11: return $\text{targets}$

### Datasets

We use a dataset comprised of labeled 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 use 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, is a corpus of around 4000 legal cases from the Federal Court of Australia. We create a target for each document by using the catch-phrases 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), publicly downloadable from the Hugging Face library and already split into 18,949 (~85%) documents for training and 3,269 (~15%) for testing, consists of 22218 US Congressional Bills with human-written references. The legal datasets statistics, described in Table 1, show that the AustLII documents are much longer than the BillSum ones.

<table>
<thead>
<tr>
<th>Statistic</th>
<th>AustLII</th>
<th>BillSum</th>
</tr>
</thead>
<tbody>
<tr>
<td># sentences</td>
<td>222</td>
<td>14</td>
</tr>
<tr>
<td># words</td>
<td>7362</td>
<td>667</td>
</tr>
<tr>
<td># tokens</td>
<td>7983</td>
<td>722</td>
</tr>
<tr>
<td># docs</td>
<td>1754</td>
<td>22218</td>
</tr>
</tbody>
</table>

Table 1: The datasets 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.

**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. With local attention, each token attends to itself and its neighbors, whereas with global attention, the first token is connected to everything else, as in the full attention. About the quadratic Transformer, we choose BART (Lewis et al. 2020) because: 1) It has an official public checkpoint of the base version. 2) 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-size models.

4 Experiments
Experimental Settings

In order to assess the performance of Se3 in low-resource regimes, the experiments are twofold. First, we consider the limited GPU memories issue. Here we experiment 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 cannot 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-tune LEGAL-BERT with both losses, i.e., the triplet and the contrastive loss. In order to assess if Se3 allows existing models to achieve a performance gain in low-resource regimes, we use BART and LEGAL-BERT as baselines, truncating the input according to each chunk max size without any text segmentation, as they were designed. Therefore, the input sizes and memory requirements are the same, but the solutions with Se3 read the complete document details without truncation.

Second, we consider the labeled data scarcity problem. In particular, we fine-tune both models combined with Se3 with 10 and 100 labeled training instances. We experiment only on the BillSum dataset to compare our results with recent works on the same low-resource summarization task.

Training Details

We train LEGAL-BERT for 1 epoch for metric learning using a batch size of 8 and a learning rate set to $2 \times 10^{-5}$. About abstractive summarization, we train 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 $5 \times 10^{-5}$. At inference time, we use a beam size and a length penalty of 2.

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

<table>
<thead>
<tr>
<th>System (MaxLen)</th>
<th>AustLI</th>
<th>BillSum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baselines</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PEGASUS_BASE</td>
<td>51.42/29.68/37.78</td>
<td></td>
</tr>
<tr>
<td>BART_BASE (1024)</td>
<td>54.42/35.81/41.98</td>
<td></td>
</tr>
<tr>
<td>BART_BASE (512)</td>
<td>48.84/33.67/40.99</td>
<td></td>
</tr>
<tr>
<td>BART_BASE (256)</td>
<td>45.99/32.63/38.09</td>
<td></td>
</tr>
<tr>
<td>BART_BASE (128)</td>
<td>42.32/27.84/31.48</td>
<td></td>
</tr>
<tr>
<td>Baselines w/ Se3 - triplet</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BART_BASE (1024)</td>
<td>59.04/52.46/53.67</td>
<td>57.31/37.85/43.78</td>
</tr>
<tr>
<td>BART_BASE (512)</td>
<td>53.14/46.44/47.38</td>
<td>56.65/35.73/40.99</td>
</tr>
<tr>
<td>BART_BASE (256)</td>
<td>44.55/36.50/37.05</td>
<td>51.99/32.63/37.11</td>
</tr>
<tr>
<td>BART_BASE (128)</td>
<td>37.28/31.42/31.83</td>
<td>44.06/28.69/32.00</td>
</tr>
<tr>
<td>Baselines w/ Se3 - contrastive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BART_BASE (1024)</td>
<td>57.96/50.92/52.49</td>
<td>57.66/38.20/44.11</td>
</tr>
<tr>
<td>BART_BASE (512)</td>
<td>52.66/46.71/46.66</td>
<td>55.96/35.82/41.27</td>
</tr>
<tr>
<td>BART_BASE (256)</td>
<td>45.18/36.82/37.52</td>
<td>52.54/33.00/37.61</td>
</tr>
<tr>
<td>BART_BASE (128)</td>
<td>37.54/31.89/32.27</td>
<td>44.29/28.90/32.27</td>
</tr>
</tbody>
</table>

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

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

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.

Models performance comparison. Solutions with Se3 significantly perform the best. In particular, our solution is more effective for the AustLI documents because they are very long, leading to a consistent boost in performance. In fact, the baselines truncate the input, discarding valuable information in the final summary. Comparing the models show no performance difference for short inputs. Instead, LED can process longer input sequences thanks to the linear complexity of its encoder self-attention, obtaining better results than BART that can process input up to 1024 tokens in length.

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

Chunks memory requirements comparison. 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 padded inputs if shorter. Thus, the memory requirements no longer decrease proportionally below such threshold.
The performance of thematic legal language modeling.

We study whether a domain-specific language model trained on thematic similarity improves pairs alignment and summarization performance. For this purpose, we compare the LEGAL-BERT of Se3, which is trained on a thematic metric learning task, with pure BERT and LEGAL-BERT without fine-tuning. Results show the better performance of our method that uses a language model fine-tuned on a metric learning task to learn the sentence thematic representation.

Summaries Accuracy

In order to evaluate the accuracy of the predicted summaries to not rely only on syntactic metrics as ROUGE, we first use BERTSCORE (Zhang et al. 2020b) for semantic assessment. Second, we investigate the eventual redundancy because of the independent chunk processing and the final concatenation. To this end, we use the same approaches as Xiao and Carenini (2020). In detail, we first use a Unique n-gram ratio to measure n-grams uniqueness. Here, the lower the score, the more redundant the document.

\[
\text{Uniq}_{\text{n-gram}} = \frac{\text{count}(\text{uniq}_{\text{n-gram}})}{\text{count}(\text{n-gram})}
\]  (3)

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

\[
\text{NID} = 1 - \frac{\text{entropy}(D)}{\log(|D|)}
\]  (4)

Table 6 reports the results using LED. The semantic assessment score of Se3 with respect to the baselines is similar for the BillSum documents and higher for the AustLII ones. 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.

### Results on Labeled Data Scarcity

Table 4 shows the performance of labeled data scarcity summarization. We use 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 that creating high-correlated source-target pairs is critical in low-resource settings. In detail, the smaller the chunks, the greater the labeled data, allowing Transformers to train on more instances. Indeed, we achieve baseline-like results even with models trained on chunk sizes of 64-128.

### Ablation Studies

We conducted additional experiments with ablation consideration (Table 5). We use a chunk size of 512-1024, fine-tuning LED for 5 epochs as done in Table 2 and Table 3.

#### The performance of semantic segmentation.

We study whether our segmentation helps to create better chunks. To this end, we compare Se3 with a sentence-level segmentation. We follow our algorithm and segment documents based on the sentences without considering the semantic similarity phase (Alg. 2). Results prove that mere sentence-level segmentation leads to the worst results. Applying semantic segmentation using the sentence representation from BERT improves the accuracy because, without Se3, sentences semantically closer can be split into different chunks, worsening the summarization performance. Moreover, Table 5 also reports more content coverage between source-target pairs using Se3, computed with the average ROUGE-1 precision.

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

<table>
<thead>
<tr>
<th>System (MaxLen)</th>
<th>BillSum (10)</th>
<th>BillSum (100)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baselines</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MTL-ABS</td>
<td>41.22/18.61/26.33</td>
<td>45.29/22.74/29.56</td>
</tr>
<tr>
<td>PEGASUS/LARGE</td>
<td>40.48/18.49/27.27</td>
<td>44.78/26.40/34.40</td>
</tr>
<tr>
<td>BART_BASE</td>
<td>39.58/18.94/26.63</td>
<td>44.66/24.87/31.09</td>
</tr>
<tr>
<td>LED_BASE</td>
<td>41.10/21.15/27.93</td>
<td>47.68/26.98/32.43</td>
</tr>
<tr>
<td>Solutions w/ Se3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BART_BASE (1024)</td>
<td>44.37/21.17/27.57</td>
<td>47.85/26.67/33.36</td>
</tr>
<tr>
<td>BART_BASE (512)</td>
<td>46.58/22.03/28.23</td>
<td>49.88/26.84/33.33</td>
</tr>
<tr>
<td>BART_BASE (256)</td>
<td>46.50/23.24/28.54</td>
<td>48.17/26.55/31.51</td>
</tr>
<tr>
<td>BART_BASE (128)</td>
<td>41.48/22.73/26.37</td>
<td>42.42/25.42/28.98</td>
</tr>
<tr>
<td>LED_BASE (4096)</td>
<td>38.48/19.26/26.36</td>
<td>48.11/26.44/31.91</td>
</tr>
<tr>
<td>LED_BASE (2048)</td>
<td>42.35/20.70/27.12</td>
<td>47.71/26.33/32.12</td>
</tr>
<tr>
<td>LED_BASE (1024)</td>
<td>45.32/22.67/29.12</td>
<td>48.28/26.97/33.46</td>
</tr>
<tr>
<td>LED_BASE (512)</td>
<td>46.94/23.04/29.29</td>
<td>50.45/27.73/33.74</td>
</tr>
<tr>
<td>LED_BASE (256)</td>
<td>46.22/24.32/29.16</td>
<td>48.13/27.16/31.89</td>
</tr>
<tr>
<td>LED_BASE (128)</td>
<td>40.14/22.76/26.05</td>
<td>40.93/25.29/28.55</td>
</tr>
</tbody>
</table>
Table 5: The ablations to study how each module of our method contributes to the performance gain. We gradually include each component of our solution to show performance improvement. Se3 is our final configuration, which includes a semantic segmentation with LEGAL-BERT trained on a thematic metric learning task. Best values are bolded.

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

5 Conclusion

In this paper, we introduced Se3 to address the abstractive long document summarization in the legal domain under low-resource regimes, namely with limited GPU memories and labeled data scarcity, where the accuracy of existing approaches drops. According to our extensive experiments, state-of-the-art abstractive summarization Transformers, thanks to Se3, process all document details without truncations, significantly boosting performance in low-resource scenarios. Moreover, we proved that our method generates semantically accurate summaries.

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

Broader Impact and Ethical Statement

Summarize long documents can benefit from using our solution, even in small organizations with very scarce resources. However, because of the social impact of legislation and biases in pre-trained Transformers, domain experts should guide the usage of our method to validate the quality of the inferred summaries.

Acknowledgments

We thank the Maggioli Group 4 for granting a Ph.D. scholarship to Luca Ragazzi. We also thank the anonymous reviewers and Giacomo Frisoni and Lorenzo Valgimigli, Ph.D. students at Dept. of Computer Science and Engineering in Cesena, for reading the paper and suggesting corrections.

---

4in particular Manlio Maggioli, Paolo Maggioli, Cristina Maggioli, Amalia Maggioli, Nicoletta Belardinelli, and Andrea Montefiori. https://www.maggioli.com/who-we-are/company-profile
References


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


Frisoni, G.; Moro, G.; and Carbonaro, A. 2020b. Towards Rare Disease Knowledge Graph Learning from Social Posts of Patients. In RiiForum, 577–589. Springer.


