Structured Summarization: Unified Text Segmentation and Segment Labeling as a Generation Task

Hakan Inan, Rashi Rungta, Yashar Mehdad

Meta AI

{inan,rashirungta,mehdad}@meta.com

Abstract

Text segmentation aims to divide text into contiguous, semantically coherent segments, while segment labeling deals with producing labels for each segment. Past work has shown success in tackling segmentation and labeling for documents and conversations. This has been possible with a combination of task-specific pipelines, supervised and unsupervised learning objectives. In this work, we propose a single encoder-decoder neural network that can handle long documents and conversations, trained simultaneously for both segmentation and segment labeling using only standard supervision. We successfully show a way to solve the combined task as a pure generation task, which we refer to as *structured summarization*. We apply the same technique to both document and conversational data, and we show state of the art performance across datasets for both segmentation and labeling, under both high- and low-resource settings. Our results establish a strong case for considering text segmentation and segment labeling as a whole, and moving towards general-purpose techniques that don't depend on domain expertise or task-specific components.

1 Introduction

Text segmentation is the task of organizing text (documents or conversations) into contiguous segments that are individually coherent. This problem can be regarded as a standalone task; however, in practical applications, one is typically interested in labeling each segment with a description. The latter problem is akin to the familiar task of text summarization, applied to each segment individually. The two tasks together can then power more practical applications: dividing a long news piece into sections accompanied by section-level headlines, turning an informational document into topic segments with titles for each topic, generating an outline for conversation transcripts, etc.

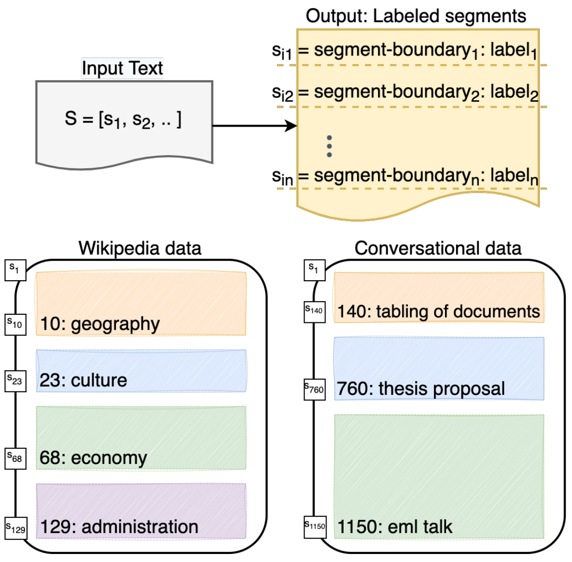

Figure 1: Text segmentation and segment labeling. The input text is depicted as a list of sentences S. Given n segments, the task output comprises of *[(s_{i_1}: $label_1$), . . ., (s_{i_n}: $label_n$)]*. Shown are labeled segments from two examples of input text, along with the sentence positions that mark the segment boundaries.

The segmentation problem was initially tackled as an unsupervised problem, by computing *coherence* scores between consecutive pairs of sentences in text, and determining segment boundaries according to certain procedures (Hearst, 1994). More recently, text segmentation has been recast as a purely supervised problem, where a neural network can learn segment boundaries from direct dataset supervision (Koshorek et al., 2018). These models perform well, especially in the presence of a large amount of training data. Despite the success of the more general-purpose supervised approaches, unsupervised objectives and task-specific procedures are still competitive in low-resource settings, such as for conversation segmentation (Xing and Carenini, 2021; Solbiati et al., 2021). Conversations tend to be longer and less information-dense compared to expository text like from Wikipedia, which may provide additional reasoning for modality-specific treatment to segmentation. Currently, there is not a single, general-purpose approach that works across text modality and training data volume. Further, due to the long nature of input text, existing methods typically adopt specific mechanisms, such as two-level encoding schemes (Koshorek et al., 2018; Arnold et al., 2019; Xing et al., 2020; Solbiati et al., 2021).

In the context of segmentation, label generation is thought of as a boundary-constrained summarization or classification problem, with freely generated or categorical labels respectively. In either case, the segment labels are produced using segmentation-aware task components, like segment-restricted attention mechanisms (Zhang et al., 2019). We note that despite the unifying pressure of the practical applications mentioned above, the label generation task has not been well-coupled to the text segmentation literature, and hence there are fewer works where both segmentation and segment labeling are tackled in a unified context.

In this work, we argue for the unification of the segmentation and segment labeling tasks, and demonstrate that the two can naturally be put in a monolithic and general-purpose framework while mutually benefiting each others' performance. We start by establishing a simple yet strong baseline for segmentation using a single, long-input transformer encoder trained only with segmentation supervision. We then introduce *generative segmentation*, a novel approach to segmentation as a token generation task, and use an encoder-decoder transformer for it. We show a procedure for generative segmentation that matches the performance of the traditional, discriminative, approach. Finally, we put segmentation and segment labeling on equal footing by generating the segment boundary indices as well as labels in the same output token sequence. We show that this approach sets a new state-of-the-art for both segmentation and labeling. Henceforth, we refer to the combined task of generative segmentation + segment label generation as *structured summarization*.

We summarize our contributions as follows:

- We introduce a strong baseline for text segmentation that works across text modalities (documents, conversations) and data scale (high- and low-resource) using a monolithic, supervised learning approach.
- We introduce generative segmentation, and show that it matches the performance of traditional, discriminative segmentation.
- We propose unifying segmentation and labeling into a single *structured summarization* task with a general-purpose text generation scheme, and show it achieves state-of-the-art performance for both segmentation and segment labeling.
- We show that one can pretrain models for structured summarization and transfer them to low-resource (including conversational) datasets, removing the need to use multiple objectives or task-specific logic for obtaining state-of-the-art performance.

2 Background: Text Segmentation and Segment Labeling

In text segmentation, one is given a list of sentences $S = [s_1, s_2, \ldots]$ with length $|S|$. Although several formulations exist, in a commonly accepted formulation, the task is to identify a subset of sentences as *segment boundaries*, which are understood to be the last sentences in each segment. If there are n target segments, the correct output can be understood as a list of n unique (& monotone) indices, each index being less than $|S|$. We note that in this work we focus on *linear segmentation* as opposed to *hierarchical segmentation*, wherein one can divide each segment into further sub-segments.

One can tackle the segmentation task in several ways. In low-resource (or no-resource) settings, one can proceed without supervision, as has been done traditionally (Hearst, 1994; Xing and Carenini, 2021; Solbiati et al., 2021). The most common approach computes a *coherence* metric between each consecutive sentence pair (refs) and uses a post-processing method (e.g. TextTiling, Hearst (1994)) that determines "deep enough" valleys in the resulting coherence vs sentence index curve to draw segment boundaries at certain sentence indices. For instance, coherence can be calculated as the cosine similarity between sentence embeddings given by a large language model. In high-resource settings, one can use standard supervision: calculate embeddings for each sentence, then train a binary classifier on top of each sentence embedding that determines whether the sentence is the last sentence of a segment. Alternatively, one can implement supervision as sequence tagging, with the sentence classifier determining whether a

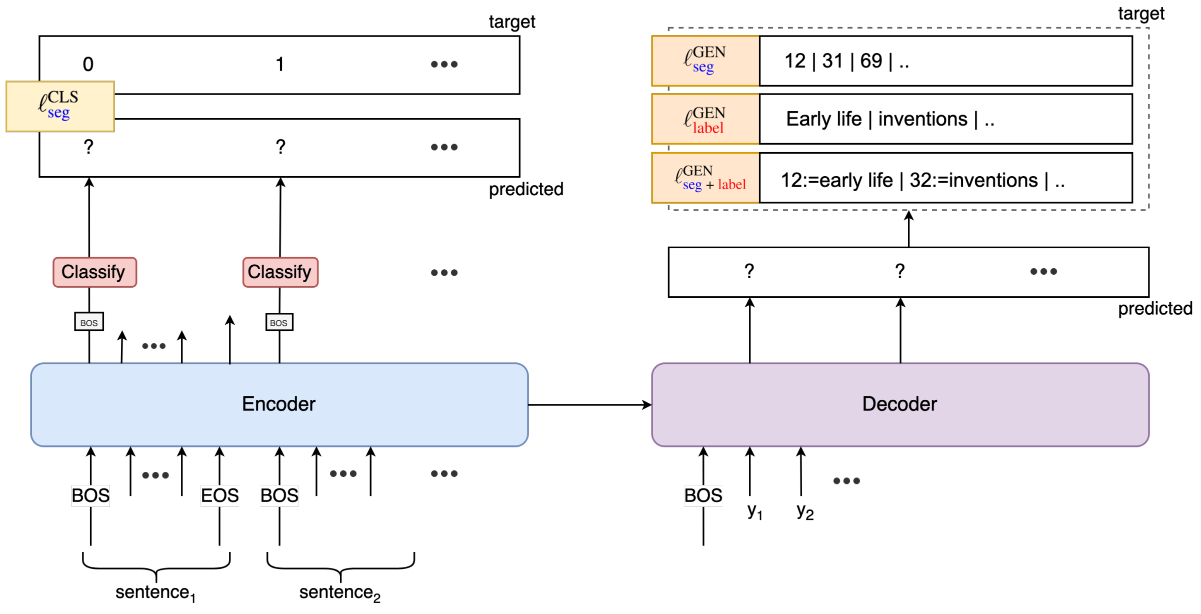
target
0
1
$\ell^{\text{CLS}}_{\text{seg}}$
?
?
predicted
Classify
Classify
BOS
BOS
Encoder
BOS
EOS
BOS
sentence$_1$
sentence$_2$
target
$\ell^{\text{GEN}}_{\text{seg}}$
12 | 31 | 69 | ..
$\ell^{\text{GEN}}_{\text{label}}$
Early life | inventions | ..
$\ell^{\text{GEN}}_{\text{seg + label}}$
12:=early life | 32:=inventions | ..
?
?
predicted
Decoder
BOS
y$_1$
y$_2$

Figure 2: Our proposed setup for segmentation and segment labeling. In structured summarization, we use $\ell^{\text{GEN}}_{\text{seg + label}}$.

sentence is the beginning or inside of a segment (Barrow et al., 2020).

Segment labeling is a task that depends on segmentation, and involves producing a descriptive label for each identified segment. It can be implemented as either a discrimination, or generation task. For domains where labels can be binned into predetermined categories (e.g. particular Wikipedia domains with a well-defined ontology), one can implement it as a discriminative task with categorical labels with a fixed vocabulary. We note that the labeling task is discriminative for most prior work that produced or used publicly available datasets (Arnold et al., 2019; Barrow et al., 2020; Lo et al., 2021a). Given a label vocabulary of size N, one can either directly train an N-way classifier on top of sentence embeddings (Arnold et al., 2019), or pool sentence embeddings within a given segment prior to classification (Barrow et al., 2020). Unfortunately, for a vast number of domains, segment labeling is not easily amenable to a discriminative treatment. These domains include free-form conversations, meeting transcripts, news articles, and even most Wikipedia content. In this case, generation models may be used. For instance, one can utilize sequence-to-sequence neural networks with some form of "segmentation-aware" attention mechanism (Zhang et al., 2019; Liu et al., 2021).

3 Structured Summarization

We view segmentation and labeling as a single task that we refer to as *structured summarization*, guided by the principle of using simpler and more general-purpose approaches, as well as by the wholistic nature of practical applications that involves carrying out both tasks at once.

In this section, we first describe our encoder backbone that we use as a strong baseline for discriminative segmentation (Section 3.1). We then introduce generative segmentation (Section 3.2), which turns segmentation into a sequence generation task. We then combine generative segmentation with segment label generation (Section 3.3). Finally, we introduce a recipe for state-of-the-art performance for structured summarization on low-resource datasets (Section 3.4).

3.1 Discriminative Segmentation with a Single Transformer Encoder

As opposed to the common two-level encoding scheme for encoding text as tokens $\rightarrow$ sentences (Koshorek et al., 2018; Glavaš and Somasundaran, 2020; Xing et al., 2020; Xing and Carenini, 2021), we encode the whole sequence at once with a long transformer. Recalling that the original sequence was described as a list of sentences $S = [s_1, s_2, \ldots]$ with length $|S|$ (see Section 2), we expand each sentence in terms of tokens (e.g. SentencePiece, Kudo and Richardson (2018)):

$$s_i = [t_{i_1}, t_{i_2}, \ldots], \qquad i = 1, 2, \ldots, |S|.$$

Thus, the overall sequence becomes

$$S = [t_{1_1}, t_{1_2}, \ldots\ldots, t_{|S|_1}, \ldots, t_{|S|_m}],$$

where m is the number of tokens in the last ($|S|^{\text{th}}$) sentence. We note that the first token of each sentence is a predetermined BOS token index:

$$t_{1_1} = t_{2_1} = \ldots t_{|S|_1} = \text{idx}_{BOS}. \qquad (1)$$

We then feed the whole sequence S to a transformer encoder. In order to perform discriminative segmentation, we apply a shared binary softmax classifier on top of each BOS token. For training, we use the binary sentence labels in the dataset $L_{s_i} \in \{0, 1\}$ that indicate whether the i^{th} sentence is the last sentence of a segment. We use the standard cross-entropy (BCE) loss, which we call $\ell_{\text{seg}}^{\text{CLS}}$:

$$\ell_{\text{seg}}^{\text{CLS}} = \sum_{i=1}^{|S|} \text{BCE}(h(f(t_{i_1})), L_{s_i}).$$

Above, h refers to the binary classifier, while f refers to the representation of an input token using the neural network encoder. Refer to Figure 2 for a visual description.

3.2 Generative Segmentation

A major step towards combining segmentation and labeling tasks is to cast segmentation as a generation problem. To this end, we map target segmentation labels into target sequences. We simply turn segment boundary indices into strings, and combine them in a single target sequence. Particularly, given the sentence positions corresponding to segment boundaries, i.e. the ordered set $pos_{seg} = \{i \mid L_{s_i} = 1\}$. We turn the integer-valued positions to strings, and combine them in a single target sequence with delimiters. For instance, in a text with 3 target segments such that $pos_{seg} = [31, 410, 680]$, the target sequence is the string '31 | 410 | 680'. We then turn this string into a list of tokens $\{y_1, y_2, \ldots, y_m\}$ and use as the target tokens. We use an encoder-decoder transformer with the encoder described as in Section 3.1, and train it with standard teacher forcing. Assuming that the NN decoder outputs a probability distribution over tokens $\hat{p}_i$ at each each time step i, and denoting with 1_k a one-hot (discrete) probability distribution with the k^{th} element set to 1, the generative segmentation loss is

$$\ell_{\text{seg}}^{\text{GEN}} = \sum_{i=1}^{m} \text{CE}(\hat{p}_i, 1_{y_i}).$$

Above, CE refers to the standard cross-entropy loss, and m denotes the sequence length in tokens.

Encoding sentence positions. We note that the neural network needs to develop an implicit ability to convey position information from input to its decoder, since it is required to produce sentence position as tokens at decoder output. One may suspect that position embeddings may facilitate such ability. On the other hand, it is common for transformer architectures to utilize relative position embeddings (the architecture we use in our experiments, LongT5, falls into this category), without other explicit mechanisms to encode sentence or token positions. We attempt to alleviate this problem from a minimal effort perspective: At the encoder input for the i^{th} sentence, we use the i^{th} vocabulary token embedding in place of a fixed BOS token index. Formally, in contrast to (1), we set

$$t_{1_1} = 0, \;\; t_{2_1} = 1, \;\; \ldots, t_{|S|_1} = |S| - 1.$$

In doing so, we expose the decoder to unambiguous position information that it can exploit for producing sentence indices at the decoder. Importantly, we achieve it without employing custom schemes, such as dedicated sentence position embeddings. In Section A of the Appendix, we show that this very simple approach improves substantially over the naive approach, enabling generative segmentation to close the gap with discriminative segmentation.

Post-processing. Finally, we post-process the decoder output into a list of segment boundary sentence positions. We split the output by the expected delimiter ("|"), and turn the resulting list of strings into a list of integers, while enforcing they be between 0 and $|S| - 1$. Any erroneous part of the output is omitted from the final list of boundary sentence positions. As shown in Section B of the Appendix, a well-trained transformer has an erroneous output only about 0.1 percent of the time.

3.3 Combined Generative Segmentation + Label Generation

Having described generative segmentation, we can now combine the segmentation and label information into a single target sequence. Given n target segments for an example text, we first prepare two target sequences:

- n ordered sentence position strings (0-based) separated by a delimiter ("|"), each position indicating the last sentence in a target segment in text. This is optimized via $\ell^{\text{GEN}}_{\text{seg}}$ (see Section 3.2).
- n ordered segment labels separated by a delimiter ("|"). We use the same teacher forcing loss as with generative segmentation. We omit details since label generation is a standard seq2seq task, with no special treatment in our approach. We denote the loss for training segment labeling $\ell^{\text{GEN}}_{\text{label}}$.

For our canonical structured summarization setup, we simply interweave the two output sequences, aligning the target positions and the target labels (separated with a new delimiter, ":="). The decoder output is trained against $\ell^{\text{GEN}}_{\text{seg + label}}$. See Figure 2 for a visual depiction.

3.4 High-resource Pretraining Prior to Low-resource Training

Judging from the dominance of unsupervised or loss-augmented methods in prior work in low-resource segmentation and labeling settings, one can argue that monolithic architectures like a single transformer trained purely with dataset supervision don't do well on low-resource datasets. We shall demonstrate this point quantitatively in our experiments (see Section 4). For generality and simplicity, it is nonetheless desired to close the performance gap for the monolithic approach trained only with dataset supervision. To this end, we adopt the modern findings from the field that demonstrated the use of pretraining (Devlin et al., 2018; Lewis et al., 2019; Brown et al., 2020). We pretrain our structured summarization models on a large dataset, and treat low-resource training as fine-tuning. In this work, our pretraining dataset (Wiki-727K) has ~0.5M training examples, and we consider scales on the order 10^3 or less as low-resource. We show in our low-resource experiments that this simple transfer learning setup is enough to not only close the performance gap, but also to set new state-of-the-art. Of note, we are able to translate performance gains from documents to conversations, the latter modality lacking high-resource datasets.

4 Experiments

In this section, we apply our method (see Section 3) to 3 datasets, each with distinct characteristics:

Wiki-727K (Koshorek et al., 2018): This dataset contains 727,746 English Wikipedia articles, with each article segmented per its existing sections, with segment labels being the section titles. The dataset is split into train/dev/test as 80/10/10 %. We use this dataset as a high-resource arena to compare structured summarization to existing segmentation methods. We also utilize this dataset for pretraining prior to low-resource training.

WikiSection (Arnold et al., 2019): This dataset contains 38k full-text documents from English and German Wikipedia, annotated with sections and topic labels. Differently from Wiki-727K, WikiSection contains normalized topic labels for each section, which are categorical. For that reason, it is amenable to and used for discriminative segment labeling (Arnold et al., 2019; Barrow et al., 2020). In our experiments, we use the English portion, which includes two datasets corresponding to two Wikipedia domains: 1) *en_disease*, containing 3.6K total docs, and 2) *en_city*, containing 19.5K total docs. Both datasets are split into train/dev/test as 70/20/10 %. We use this dataset as a mid-to-low-resource arena to compare structured summarization to existing segmentation as well as discriminative labeling methods.

QMSum (Zhong et al., 2021b): This dataset is a collection of long meeting transcripts from academic, product, and committee domains. Among other useful data, each example contains manually curated topic segments and accompanying topic labels. It contains a total of 232 examples split into train/dev/test roughly as 68/16/16 %. We use this dataset as a low-resource conversational arena, in which we compare structured summarization to existing segmentation approaches.

Throughout, we use the following segmentation and labeling metrics to judge performance:

P$_k$ (Beeferman et al., 1999): A widely used segmentation metric, P_k estimates the probability that randomly drawn sentence indices that are k sentences apart fall into non-agreeing segments between the reference and predicted segmentation. Keeping with the common practice, we set k equal to half the mean segment length of the reference.

Rouge (Lin, 2004): This is a standard text similarity metric used for summarization. In order to compute Rouge metrics, we gather all predicted and target segment labels and serialize them with a delimiter between labels. We use Rouge-1 (R-1), Rouge-2 (R-2), and Rouge-L (R-L).

Model	CLS	GEN			
	P$_k$ ↓	P$_k$↓	R-1↑	R-2↑	R-L↑
$\ell^{\text{CLS}}_{\text{seg}}$ (DS)	15.4	-	-	-	-
$\ell^{\text{GEN}}_{\text{seg}}$	-	15.8	-	-	-
$\ell^{\text{CLS}}_{\text{seg}} + \ell^{\text{GEN}}_{\text{seg}}$	15.5	15.8	-	-	-
$\ell^{\text{GEN}}_{\text{label}}$	-	-	48.7	28.2	48.2
$\ell^{\text{CLS}}_{\text{seg}} + \ell^{\text{GEN}}_{\text{label}}$	15.4	-	48.7	28.1	48.3
$\ell^{\text{GEN}}_{\text{seg + label}}$ (SS)	-	**15.0**	**49.1**	**28.5**	**48.6**
$\ell^{\text{CLS}}_{\text{seg}} + \ell^{\text{GEN}}_{\text{seg + label}}$	**15.0**	**15.0**	48.7	28.1	48.2

Table 1: Results for different combinations of segmentation and labeling losses on Wiki-727K. DS refers to Discriminative Segmentation and SS refers to Structured Summarization. Structured summarization has the best performance across segmentation and labeling. Boldface fonts indicate the best results.

Label F1: WikiSection dataset contains categorical segment labels. For this reason, we compute the micro-averaged F1 as a measure of accuracy for our generated labels in alignment with prior work. We first align the generated segments to the target segments to have a many-to-one mapping between the two sets of segments. This is done by looking at the closed segment boundary to the left and the right of the predicted segment, mapping the predicted segment to the target segment with maximum overlap. The label for this target segment is then considered the ground truth label for the predicted segment during the F1 calculation.

Model. For all our experiments, we use Long T5-base with transient global attention (Guo et al., 2021). We were not able to set up a reliable workflow with large-sized models due to the limits of our computational setup. On the other hand, our hypotheses are testable using any-sized transformer of sufficient expressibility, and our results are self-contained with fair comparisons including strong baselines of our own with same-sized setups. We use AdamW (Loshchilov and Hutter, 2017) with a learning rate of 0.0005 for all experiments. We use a batch size of 128, 64, and 16 for Wiki-727K, WikiSection, and QMSum respectively. We use maximum sequence length of 16384 for Wiki-727K and WikiSection, 32768 for QMSum.

4.1 Validating Structured Summarization

We train a Long T5-Base on the Wiki-727K training set using all viable combinations of available losses ($\ell^{\text{CLS}}_{\text{seg}}$, $\ell^{\text{GEN}}_{\text{seg}}$, $\ell^{\text{GEN}}_{\text{label}}$, and $\ell^{\text{GEN}}_{\text{seg + label}}$). Our aim is to show the merits of the combined approach,

Model	P$_k$ ↓	R-1 ↑	R-2 ↑	R-L ↑
TextSeg	22.13	-	-	-
TLT-TS	19.41	-	-	-
CATS	15.95	-	-	-
DS ($\ell^{\text{CLS}}_{\text{seg}}$)	15.4	-	-	-
SS ($\ell^{\text{GEN}}_{\text{seg + label}}$)	**15.0**	**49.1**	**28.5**	**48.6**

Table 2: Wiki-727K Test Set Segmentation Results. TextSeg: (Koshorek et al., 2018), TLT-TS: (Glavaš and Somasundaran, 2020), CATS: (Glavaš and Somasundaran, 2020). DS refers to Discriminative Segmentation and SS refers to Structured Summarization.

while understanding the contributions of different components. The results are shown in Table 1. We make the following remarks:

- Generative segmentation performance ($\ell^{\text{GEN}}_{\text{seg}}$, P$_k$ $= 15.8$) is nearly on par with discriminative segmentation ($\ell^{\text{CLS}}_{\text{seg}}$, P$_k$ $= 15.4$), demonstrating the practical feasibility of generative segmentation.
- Although combining $\ell^{\text{CLS}}_{\text{seg}}$ and $\ell^{\text{GEN}}_{\text{seg}}$ doesn't improve segmentation performance, combining $\ell^{\text{GEN}}_{\text{seg}}$ and $\ell^{\text{GEN}}_{\text{label}}$ (aka structured summarization) leads to best segmentation. This shows that the labeling task improves generative segmentation. In contrast, when labeling is combined with discriminative segmentation ($\ell^{\text{CLS}}_{\text{seg}} + \ell^{\text{GEN}}_{\text{label}}$), the segmentation performance does not improve.

Our results provide evidence that unifying segmentation and labeling as a generative task is more favorable compared to carrying them out separately. In our next set of experiments, we validate structured summarization (using $\ell^{\text{GEN}}_{\text{seg + label}}$) against current best methods in high, medium, and low-resource settings, including both document and conversation data.

4.2 High Resource, Document Setting

We first set out to compare structured summarization to existing text segmentation approaches in the high-resource setting. We also aim to establish single transformer-based discriminative segmentation over the standard discriminative segmentation approach, namely using a two-level (token-level→sentence-level) encoding. Although we are not aware of a published baseline for segment label generation for Wiki-727K, we also report *Rouge* metrics for our models where applicable.

Table 2 shows the results. We first note that our single transformer encoder outperforms the best previous model, which uses a two-level transformer augmented with a coherence loss. We also find that structured summarization outperforms discriminative segmentation, setting a new state-of-the-art on Wiki-727K. Overall, these results favor general-purpose methods trained with only dataset supervision.

4.3 Mid- to Low-resource, Document Setting

In order to test our hypothesis that generative segmentation and labeling is competitive for mid- to low- resource settings, we evaluate the English parts of the WikiSection dataset. We set up two types of experiments, namely naive and pretrained. The former uses a Long T5-base as described in Section 4, whereas for the latter we pretrain Long T5-base on Wiki-727k with $\ell^{\text{GEN}}_{\text{seg + label}}$ and then finetune on *en_city* and *en_disease*. We used sentence similarity as described in Section B of the Appendix to verify that there was no data leakage between the Wiki-727k train (pretraining) and Wikisection test (finetuning) datasets.

We evaluate each of these models with discriminative segmentation and structured summarization. Note that in these sets of experiments, we compare discriminative labeling from prior work to our generative labeling. Hence, along with the metrics used in Section 4.2, we also report the micro-averaged F1 for the task of segment labeling, similar to (Barrow et al., 2020) and (Lo et al., 2021a).

The results are shown in Table 3. Finetuning over Wiki-727K gives us the best structured summarization model for *en_disease*. For *en_city*, we notice that pretrained discriminative segmentation does result in a better P_k than pretrained structured summarization, but only marginally so. For all the labeling metrics, namely micro-averaged F1 and the rouge metrics, both naive and pretrained structured summarization models perform better than the prior work, for *en_disease* as well as *en_city*.

4.4 Low-resource, Conversational Setting

Finally, we aim to establish the use of structured summarization in a low-resource setting for a different text modality, namely conversational. In this setting, segmentation regards each conversation turn as one "sentence", and predicts positions of turns at segment boundaries. The QMSum dataset is very low-resource, comprising only 157 training examples.

In the QMSum dataset, we deal with two interrelated problems: presence of very large number of turns (>1000), and large number of input tokens even when we truncate the number of tokens in a turn to a reasonably large number (200). The first problem is detrimental to generative segmentation, since a Wiki-727K-pretrained decoder can learn to map sentence positions up to the number of sentences encountered in Wiki-727K training set. To overcome this, we pretrain on Wiki-727K with a modification: We prepend a random number (up to 1000) of empty sentences to each training example, thereby increasing learnable sentence positions without affecting other task parameters. To address the second problem, we set a maximum number of tokens (95) per turn that keeps all test example inputs under 32768 tokens, while augmenting the training set with replicas with different maximum number of tokens per turn for each new replica (we use 20, 50, and 200). The rationale is to expose the network to varying degrees of truncation of turn tokens for robustness.

The results are shown in Table 4. Given very few training sentences, structured summarization performs labeling well as judged by Rouge metrics. For segmentation, although naive structured summarization isn't able to perform better than prior work, when pretrained on Wiki-727K, it sets a new state-of-the-art with generative segmentation, achieving a large performance improvement from the previous best approach. We note that although pretraining was done using a quite distinct text modality (descriptive documents), this facilitates major gains for the conversational modality.

5 Prior Work

Our work has most overlap with the text segmentation literature, wherein the task is sometimes called topic segmentation (e.g. Takanobu et al. (2018)). The earliest work for segmentation that we're aware of is TextTiling (Hearst, 1994), which draws on the insight that sentences belonging to the same topic/segment should be more coherent than sentences across segments, leading to an unsupervised method for segmentation. Following this work, many other methods that exploit intra-segment coherence/similarity were proposed, including works of Choi (2000), Malioutov (2006), Glavaš et al. (2016), and more recent works that combine coher-

Model	en_city					en_disease				
	$P_k \downarrow$	F1 $\uparrow$	R-1 $\uparrow$	R-2 $\uparrow$	R-L $\uparrow$	$P_k \downarrow$	F1 $\uparrow$	R-1 $\uparrow$	R-2 $\uparrow$	R-L $\uparrow$
SECTOR	14.4	71.6	-	-	-	26.3	55.8	-	-	-
S-LSTM	9.1	76.1	-	-	-	20.0	59.3	-	-	-
$\text{Transformer}^2_{\text{Bert}}$	8.2	-	-	-	-	18.8	-	-	-	-
Naive DS ($\ell^{\text{CLS}}_{\text{seg}}$)	8.2	-	-	-	-	33.5	-	-	-	-
Naive SS ($\ell^{\text{GEN}}_{\text{seg + label}}$)	9.2	73.1	79.8	62.0	79.42	24.8	38.8	52.0	30.8	51.7
Pre-trained DS ($\ell^{\text{CLS}}_{\text{seg}}$)	**6.8**	-	-	-	-	15.3	-	-	-	-
Pre-trained SS ($\ell^{\text{GEN}}_{\text{seg + label}}$)	7.1	**82.3**	**82.1**	**65.0**	**81.7**	**15.0**	**68.3**	**62.6**	**38.5**	**62.2**

Table 3: Wikisection Test set results compared to previous state-of-the-art models. SECTOR refers to the best models from (Arnold et al., 2019), S-LSTM is from (Barrow et al., 2020), and $\text{Transformer}^2_{\text{Bert}}$ from (Lo et al., 2021b). DS refers to Discriminative Segmentation and SS refers to Structured Summarization.

Model	$P_k\downarrow$	R-1$\uparrow$	R-2$\uparrow$	R-L$\uparrow$
DialogLM	38.0	-	-	-
Naive DS ($\ell^{\text{CLS}}_{\text{seg}}$)	36.7	-	-	-
Naive SS ($\ell^{\text{GEN}}_{\text{seg + label}}$)	44.4	19.8	6.6	17.9
Pre-trained DS ($\ell^{\text{CLS}}_{\text{seg}}$)	33	-	-	-
Pre-trained SS ($\ell^{\text{GEN}}_{\text{seg + label}}$)	**32.8**	**27.0**	**11.1**	**24.2**

Table 4: QMSUM Test Set segmentation results. DialogLM is from (Zhong et al., 2021a). DS refers to Discriminative Segmentation and SS refers to Structured Summarization. All DS and SS metrics are averaged over 7 runs.

ence objectives with neural networks, such as Xing et al. (2020), Xing and Carenini (2021), Glavaš and Somasundaran (2020), and Solbiati et al. (2021). There are also works for topic segmentation that draw on topic modeling. These works typically learn latent topic representations for segments, such as Riedl and Biemann (2012), Misra et al. (2009).

With the introduction of large enough segmentation datasets for the document modality, such as Wiki-727K (Koshorek et al., 2018) and WikiSection (Arnold et al., 2019), recent work has shifted focus on supervised text segmentation for documents. These works train neural networks with the segment boundary supervision from the datasets. Examples from this category include works of Koshorek et al. (2018), Arnold et al. (2019), Xing et al. (2020), Barrow et al. (2020), Glavaš and Somasundaran (2020), and Lo et al. (2021a).

Whereas document segmentation enjoyed gains facilitated by large datasets, conversation segmentation has been most successfully performed in an unsupervised way to this day. Works in the conversation modality heavily depend on intra-segment coherence, with recent examples being Xing and Carenini (2021), Solbiati et al. (2021). Zhong et al. (2021a) perform segmentation on the QMSum dataset with standard supervision, although with limited success, as evidenced by the accuracy gap compared to our method (see Table 4).

Some works tackle generating labels in the context of segmentation. Most works that used publicly available datasets developed discriminative labeling techniques, such as Arnold et al. (2019) and Barrow et al. (2020). Some works also aimed at solving the labeling task using generation methods. Zhang et al. (2019) introduced "document outline generation", where they used multiple GRU networks, separately for segmentation and label generation, while using "segment-aware" attention to constrain generation. Liu et al. (2021) use an encoder-decoder transformer for segmenting and labeling news articles. Their method is closely related to our setup, although they use discriminative-only segmentation, and utilize the decoder for only generating labels.

6 Conclusion

We introduced a generally applicable technique for unifying text segmentation and segment labeling as a single sequence generation task (*structured summarization*). We have shown that the task is suited to modern-day transformers that handle long inputs, and that one can achieve state-of-the-art performance for both segmentation and labeling across data scales and text modalities. We hope that these results will guide the field towards more general methods that perform structured summarization in research as well as in production settings.

7 Ethics Statement

We note that any generative AI technology has the potential to produce harmful, misleading, or offensive content. This should be a guiding principle when considering adopting the technology into real-life use cases.

References

Sebastian Arnold, Rudolf Schneider, Philippe Cudré-Mauroux, Felix A Gers, and Alexander Löser. 2019. Sector: A neural model for coherent topic segmentation and classification. *Transactions of the Association for Computational Linguistics*, 7:169–184.

Joe Barrow, Rajiv Jain, Vlad Morariu, Varun Manjunatha, Douglas W Oard, and Philip Resnik. 2020. A joint model for document segmentation and segment labeling. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 313–322.

Doug Beeferman, Adam Berger, and John Lafferty. 1999. Statistical models for text segmentation. *Machine learning*, 34(1):177–210.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Freddy YY Choi. 2000. Advances in domain independent linear text segmentation. *arXiv preprint cs/0003083*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Goran Glavaš, Federico Nanni, and Simone Paolo Ponzetto. 2016. Unsupervised text segmentation using semantic relatedness graphs. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*, pages 125–130. Association for Computational Linguistics.

Goran Glavaš and Swapna Somasundaran. 2020. Two-level transformer and auxiliary coherence modeling for improved text segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7797–7804.

Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2021. Longt5: Efficient text-to-text transformer for long sequences. *arXiv preprint arXiv:2112.07916*.

Marti A Hearst. 1994. Multi-paragraph segmentation of expository text. *arXiv preprint cmp-lg/9406037*.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning.

Omri Koshorek, Adir Cohen, Noam Mor, Michael Rotman, and Jonathan Berant. 2018. Text segmentation as a supervised learning task. *arXiv preprint arXiv:1803.09337*.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Yang Liu, Chenguang Zhu, and Michael Zeng. 2021. End-to-end segmentation-based news summarization. *arXiv preprint arXiv:2110.07850*.

Kelvin Lo, Yuan Jin, Weicong Tan, Ming Liu, Lan Du, and Wray Buntine. 2021a. Transformer over pre-trained transformer for neural text segmentation with enhanced topic coherence. *arXiv preprint arXiv:2110.07160*.

Kelvin Lo, Yuan Jin, Weicong Tan, Ming Liu, Lan Du, and Wray Buntine. 2021b. Transformer over pre-trained transformer for neural text segmentation with enhanced topic coherence. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3334–3340, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Igor Igor Mikhailovich Malioutov. 2006. *Minimum cut model for spoken lecture segmentation*. Ph.D. thesis, Massachusetts Institute of Technology.

Hemant Misra, François Yvon, Joemon M Jose, and Olivier Cappé. 2009. Text segmentation via topic modeling: an analytical study. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1553–1556.

Martin Riedl and Chris Biemann. 2012. Topictiling: a text segmentation algorithm based on lda. In *Proceedings of ACL 2012 student research workshop*, pages 37–42.

Alessandro Solbiati, Kevin Heffernan, Georgios Damaskinos, Shivani Poddar, Shubham Modi, and

Jacques Cali. 2021. Unsupervised topic segmentation of meetings with bert embeddings. *arXiv preprint arXiv:2106.12978*.

Ryuichi Takanobu, Minlie Huang, Zhongzhou Zhao, Feng-Lin Li, Haiqing Chen, Xiaoyan Zhu, and Liqiang Nie. 2018. A weakly supervised method for topic segmentation and labeling in goal-oriented dialogues via reinforcement learning. In *IJCAI*, pages 4403–4410.

Linzi Xing and Giuseppe Carenini. 2021. Improving unsupervised dialogue topic segmentation with utterance-pair coherence scoring. *arXiv preprint arXiv:2106.06719*.

Linzi Xing, Brad Hackinen, Giuseppe Carenini, and Francesco Trebbi. 2020. Improving context modeling in neural topic segmentation. *arXiv preprint arXiv:2010.03138*.

Ruqing Zhang, Jiafeng Guo, Yixing Fan, Yanyan Lan, and Xueqi Cheng. 2019. Outline generation: Understanding the inherent content structure of documents. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 745–754.

Ming Zhong, Yang Liu, Yichong Xu, Chenguang Zhu, and Michael Zeng. 2021a. Dialoglm: Pre-trained model for long dialogue understanding and summarization. *arXiv preprint arXiv:2109.02492*.

Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, et al. 2021b. Qmsum: A new benchmark for query-based multi-domain meeting summarization. *arXiv preprint arXiv:2104.05938*.

Appendix for "Structured Summarization: Unified Text Segmentation and Segment Labeling as a Generation Task"

A Encoding sentence positions by reuse of vocab tokens improves over naive models

As mentioned in Section 3.2, we validate the use of our simple approach to encode sentence positions in the model. We first note that the native LongT5 tokenizer doesn't use a dedicated BOS token. Therefore, when not using our approach, there is no BOS token we can use for discriminative segmentation. Regardless, we train models to classify the first token of each sentence. Additionally, we train models where we use the EOS token instead for segment boundary classification. We then compare both models to the model where we encode sentence positions according to Section 3.2. The results are in Table 5. Our approach substantially improves over the naive approach (when using either EOS or BOS), bringing generative segmentation accuracy to the same accuracy of discriminative segmentation within the same model.

Model	$P_k\downarrow$
naive LongT5 (BOS)	16.9
naive LongT5 (EOS)	16.2
LongT5 (BOS) + sentence positions	15.0

Table 5: Wiki-727K test set results for models trained with $\ell^{\text{CLS}}_{\text{seg}} + \ell^{\text{GEN}}_{\text{seg + label}}$

B Generative segmentation leads to accurately predicted sentence positions in the output sequence

Here we follow up on our claim in Section 3.2 that generative segmentation leads to non-erroneous segmentation output when generated in the list of tokens at the NN output. To back this claim, we calculated the fraction of time any output sequence includes an invalid sentence boundary position. An example could be "10 | 31 | 413" in a text with only 300 sentences (last segment boundary is over 300). Another example could be one wherein the output sequence has a non-integer-convertible component, like in "10 | 31e | 299". In Table 6, we show this erroneous fraction for structured summarization models when tested on Wiki-727K, WikiSection, and QMSum. From the table, it is clear that transformer decoders are easily able to generate tokens that represent integers within the bounds of the task semantics.

Wiki-727K	en_city	en_disease	QMSum
0.0001	0.0025	0	0

Table 6: Fraction of examples with at least one erroneous segment boundary position. This is for the structured summarization models, when tested on the respective test set.

C Verifying no data leakage between pretraining train- and finetuning test-datasets

Both Wiki727-K and Wikisection datasets are derived from Wikipedia, and thus pretraining on one and finetuning on the other had a risk of exposing the model to examples it will encounter in the test set. For part of the Wikisection experiments which use a pretrained model as described in Section 3, we use such a setup.

To mitigate the above-mentioned risk, we use Contriever (Izacard et al., 2021) to check for text similarity between the Wiki727-K train examples and Wikisection *en_disease* and *en_city* test examples to find examples that need to be eliminated from the Wiki727-K train set. In result, do not find any pairs with exact matches or alarmingly high cosine similarity.

D Details on compute and training the models

We used 8 A100 GPUs (40Gb memory) for training Wiki-727K and WikiSection models, and 1 A100 GPU for training QMSum models. The neural network we use (LongT5) has 220M parameters. One epoch of training took about 15 hrs on Wiki-727K, 30 minutes on Wikisection-en_city, 6 minutes on WikiSection-en_disease, and 20 minutes on QMSum. In all experiments, we report test results for the epoch for which the validation metric was best. The metric of interest for all models was P_k where applicable, and Rouge otherwise. We performed hyperparameter tuning by selecting runs with the best validation metric. Instead of a full grid search, we used human judgement (over a few 10s of runs) and observed that results were mostly robust to modest changes in most parameters. Our manual approach is mostly necessitated by the lack of sufficiently large computing infrastructure. The only hyperparameters we tuned were learning rate and batch size. We set the maximum number of training epochs to 10 for all experiments.