Topical Change Detection in Documents via Embeddings of Long Sequences

Dennis Aumiller * and **Satya Almasian** * and **Sebastian Lackner** and **Michael Gertz**
Heidelberg University, Heidelberg, Germany
lastname@informatik.uni-heidelberg.de

Abstract

In a longer document, the topic often slightly shifts from one passage to the next, where topic boundaries are usually indicated by semantically coherent segments. Discovering this latent structure in a document improves the readability and is essential for passage retrieval and summarization tasks. We formulate the task of text segmentation as an independent supervised prediction task, making it suitable to train on Transformer-based language models. By fine-tuning on paragraphs of similar sections, we are able to show that learned features encode topic information, which can be used to find the section boundaries and divide the text into coherent segments. Unlike previous approaches, which mostly operate on sentence-level, we consistently use a broader context of an entire paragraph and assume topical independence of preceeding and succeeding text. We lastly introduce a novel large-scale dataset constructed from online Terms-of-Service documents, on which we compare against various traditional and deep learning baselines, showing significantly better performance of Transformer-based methods.

1 Introduction

Human written texts are often a sequence of semantically coherent segments, designed to create a smooth transition between various subtopics discussed in a single document. Usually, the user's information needs are satisfied by retrieving only the relevant subtopic, and retrieving the whole document is unwieldy and may result in information overload (Misra et al., 2011; Wilkinson, 1994). In this context, estimating topic boundaries for more efficient information retrieval and summarizing relevant content is essential. To find a fitting representation that captures this topical flow in the text, a robust representation of the sections is required. A section in a document may consist of single or multiple paragraphs that convey a coherent message and share the same topic. Section boundaries often coincide with a change in topic and can segment the text into groups of connected paragraphs. Therefore, the optimal representation of a paragraph should map the topically closer paragraphs to nearby points in the embedding space and further apart from paragraphs with different topics. Despite their importance, many previous works for topical text segmentation ignore paragraphs and focus only on sentence-level granularity. Paragraphs, however, represent another semantically cohesive unit, are almost always available, and define a coarser structure than sentences. Topic boundaries generally do not appear in the middle of a paragraph; consequently, operating on paragraph level can reduce the risk of false-positive segment breaks and lower the computation cost per sentence prediction. Figure 1 shows how paragraphs group sentences and divide a text into coherent parts and how, by overlooking this valuable information, the structure in the text is lost to the model.

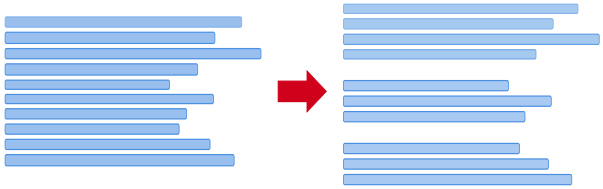

Figure 1: Visual cues such as paragraphs often give away a notion of semantic coherence, which is disregarded in sentence-level models.

Because no large labeled dataset existed, early text segmentation approaches were mainly unsupervised, using heuristics to identify whether two sentences belong to the same topic. Such approaches either exploit the fact that topically related words tend to appear in semantically coherent segments (Choi, 2000a; Hearst, 1997; Malioutov and Barzilay, 2006; Kozima, 1993; Utiyama and Isahara, 2001), or focus on topical representation

* These authors contributed equally to this work.

of text in terms of latent-topic vectors using methods, such as Latent Dirichlet Allocation (Blei et al., 2003; Misra et al., 2011, 2009; Riedl and Biemann, 2012a). Recently, with the availability of annotated data, text segmentation has been formulated as a supervised learning problem. For this, existing methods utilize expensive hierarchical neural models, where the lower-level network creates sentence representations, and a secondary network structure models the dependencies between embedded sentences (Glavas and Somasundaran, 2020; Koshorek et al., 2018). One drawback of these models is their sentence-level granularity. Although this problem is partially solved by hierarchical models, where the dependency between sentences is modeled in a hierarchical structure, training such a model is computationally expensive due to varying document lengths. Moreover, these models fail to take advantage of pre-trained language representations, such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019), which are valuable feature generators with a low cost of fine-tuning.

In this paper, we introduce a novel formulation of the supervised training setup that allows the near-trivial utilization of pre-trained language models, while simultaneously pertaining a scalable setting for the text segmentation task using paragraphs directly without intermediate sentence representations. Our approach avoids handcrafted feature engineering and lengthy and expensive training of hierarchical methods. For the evaluation, a novel dataset of Terms-of-Service documents is proposed, containing annotated paragraphs belonging to the same topic. We assume topical independence between paragraphs and show that it does not deteriorate the performance while avoiding hierarchical models' costly computation. The hypothesis is that by fine-tuning the paragraph embedding for section similarity, we generate paragraph features that detect coherent topical structures. We evaluate our models against the traditional embedding baselines for topical detection and compare against supervised and unsupervised approaches for text segmentation.

Contributions. The contributions of this paper are as follows: (i) We present text segmentation task on coarser cohesive text units (paragraphs/sections). (ii) We investigate the performance of Transformer-based models for topical change detection, and (iii) frame the task as a collection of independent binary predictions instead, eliminating overhead due to hierarchical training setups. (iv) We present a new dataset consisting of online Terms-of-Service partitioned into hierarchical sections, and make the data available for future research.[1] (v) We evaluate our model against classical baselines for text segmentation, and (vi) show our generated embeddings' effectiveness and competitive performance to other text segmentation techniques.

2 Related Work

Our work is closely tied to topic analysis, text segmentation, and Transformer language models, and we briefly review each of these areas in this section.

2.1 Topic Analysis

Detection and analysis of topical change is grounded in topic modeling approaches. Earlier work such as LDA (Blei et al., 2003), treat documents as bag-of-words, where each document is assigned to a topic distribution. Subsequent work has adopted a more sophisticated representation than bag-of-words and generally model Markovian topic or state transitions to capture dependencies between words in a document (Griffiths et al., 2004; Wallach, 2006). With the rise of distributed word representation, the focus has shifted to the combination of LDA and word embeddings (Dieng et al., 2019; Moody, 2016).

2.2 Text Segmentation

Text segmentation is the task of dividing a document into multi-paragraph discourse units that are topically coherent, with the cut-off point usually indicating a change in topic (Hearst, 1994; Utiyama and Isahara, 2001). Although the task itself dates back to 1994 (Hearst, 1994), most existing text segmentation datasets are small and limit their scope to sentences (predicting whether two sentences discuss the same topic or not). The most common one is by Choi (Choi, 2000b), containing only 920 synthesized passages from the Brown corpus. Choi's method C99 is a probabilistic algorithm measuring similarity via term overlap. GraphSeg (Glavas et al., 2016) is an unsupervised graph method that segments documents using a semantic relatedness graph of the document. GraphSeg is also evaluated on a small set of 5 manually-segmented political manifestos from the Manifesto project[2]. Another

[1] https://github.com/dennlinger/TopicalChange

[2] https://manifestoproject.wzb.eu

class of methods are topic-based document segmentations, which are statistical models that find latent topic assignments that reflect the underlying structure of the document (Beeferman et al., 1999; Brants et al., 2002; Chen et al., 2009; Dharanipragada et al., 1999; Misra et al., 2011; Riedl and Biemann, 2012b). TopicTiling (Riedl and Biemann, 2012b) performs best among this family of methods and uses LDA to detect topic shifts, with computing similarities between adjacent blocks based on their term frequency vectors. Brants et al. (Brants et al., 2002) follow a similar approach but employ PLSA (Hofmann, 2017) to compute the estimated word distributions. Another noteworthy approach based on Bayesian topic models is from Chen et al. (Chen et al., 2009), where they constrain latent topic assignments to reflect the underlying organization of document topics. They also publish a test dataset with 218 Wikipedia articles about cities and chemical elements. A hiearchical Bayesian approach was proposed by Du et al. (Du et al., 2013), which performs on par with TopicTiling on the Choi dataset. The only previous paragraph-based dataset available is the segmentation of a medical textbook which was utilized by Eisenstein and Barzilay, who propose another Bayesian approach called Bayeseg (Eisenstein and Barzilay, 2008). Unfortuantely, their dataset consists of a total of 227 chapters with similar structure only.

All mentioned methods are unsupervised learning approaches and small annotated datasets are only used for evaluation, hence, not directly comparable to our approach. Instead, we focus on supervised learning of topics and introduce a new dataset with 43,056 automatically labeled documents.

Comparable supervised approaches are from Koshorek et al. (Koshorek et al., 2018), Li et al. (Li et al., 2018) and Glavas et al. (Glavas and Somasundaran, 2020). Koshorek et al. propose a hierarchical LSTM architecture for learning sentence representation and dependency which they train on a dataset of cleaned Wikipedia articles, called Wiki-727k. Li et al. introduce a model called *SegBot*, which can handle varying levels of granularity. Their approach builds on a bidirectional RNN with pointer mechanisms, which improves labeling efficiency in the absence of large training corpora. Glavas et al. introduce Coherence-Aware Text Segmentation, which encodes a sentence sequence using two hierarchically connected Transformer networks. While similarly using a Transformer-based network architecture, they fail to make use of pre-trained language models, making training more costly compared to our setup. These models are closest in terms of problem formulation, however, there are no trained models for paragraph segmentation available. Finally, Zhang et al. (Zhang et al., 2019) extend text segmentation by outline generation and trained an end-to-end LSTM-model identifying sections and generating corresponding headings for Wikipedia documents.

2.3 Transformer Language Models

The Transformer architecture, much like RNNs, aims to solve sequence-to-sequence tasks, relying entirely on self-attention to compute representations of its input and output (Vaswani et al., 2017). Transformers have made a significant step in bringing transfer learning to the NLP community, which allows the easy adaptation of a generically pre-trained model for specific tasks. Pre-trained models such as BERT, GPT-2, and RoBERTa (Devlin et al., 2019; Liu et al., 2019; Radford et al., 2018) use language modeling for pre-training, and achieve state-of-the-art performance on a wide variety of tasks after fine-tuning. One variation is Sentence-BERT (Reimers and Gurevych, 2019) which combines two BERT models in Siamese fashion to derive semantically meaningful text embeddings. By its design, Sentence-BERT also allows for longer input sequences for pairwise training tasks. Lastly, RoBERTa is a retraining of BERT with improved training methodology, and achieves slightly better result than BERT on some tasks, which is why pre-trained RoBERTa and Sentence-RoBERTa have been chosen for the setup in this work.

3 Same Topic Prediction

We formulate text segmentation as a supervised learning task of same topic prediction on coarser units than a single sentence. By formulating the task in this manner, we gain more contextual information than a sentence and bypass the expensive step of combining sentence representations. Furthermore, this approach allows us to experiment with different sampling strategies for topic prediction task and investigate their implications on text segmentation. Our model consists of two steps: (i) Independent and Identically Distributed Same Topic Prediction (IID STP) and (ii) Sequential Inference. We fine-tune Transformer-based models to detect topical change for both paragraphs and en-

tire sections in the first step. We assume that each paragraph's or section's topic is independent of the text before and after, and later show empirically that this assumption yields good performance without a costly training of hierarchical models. Note that we only consider chunks of the same type, namely, either *only* sections or *only* paragraphs, in each model. In the second step, we use the fine-tuned Transformer-based classifiers for sequential inference on entire documents, where the segment boundaries are defined by topical change. In the following, we discuss these steps in more detail.

3.1 IID Same Topic Prediction (STP)

A document $d \in D$ is represented as a sequence of N sections $S_d = (s_1, ..., s_N)$, where each section is assigned to one of M topics $T = (t_1, ..., t_M)$, and each section contains up to K paragraphs $P_n = (p_1, ...p_K)$. We assume topical consistency within a paragraph and argue that the results for classification do not change based on the position of the paragraph within the document, since the most relevant part for our inference is the intra-section information. Therefore, if the topic assignment is defined by the function $Topic$, we have:

$$s_n = (p_1, .., p_k) \implies Topic(s_n) = t_1 \implies Topic(p_1) = t_1 = ... = Topic(p_k) = t_1 \quad (1)$$

If we define C as a chunk of text corresponding to either a section or paragraph, the topic prediction task is defined for section and paragraph granularity as follows: Given two chunks of text of the same type (c_1, c_2) and labels $y \in \{0, 1\}$, indicating whether the two chunks belong to the same topic, topical change detection can be formulated as a binary classification problem. By formulating the problem as a binary classification, detecting the topic consistency between two chunks of text can now be solved with any classifier type. In this work, we train two types of the Transformer-based classifiers for this task, one from the pre-trained language models (Liu et al., 2019) and another Siamese network (Reimers and Gurevych, 2019) variation, which is more suitable for encoding pairwise similarity. Subsequently, the two variations are discussed.

RoBERTa is a replication study of BERT pre-training with optimized hyper-parameters that applies minor adjustments to the BERT language model to achieve better performance (Liu et al., 2019). BERT and RoBERTa are a pre-trained Transformer Encoder stack. The Transformer is an architecture for shaping one sequence into another with the self-attention mechanism, which helps the model extract features from each word relative to all the other words. The Encoder stacks in BERT and RoBERTa consists of one or multiple self-attention blocks followed by a feed-forward network. These encoders learn task-independent features from the text used in the fine-tuning stage for transfer tasks during pre-training. Since the performance difference between most transformer-based language models is negligible, we choose RoBERTa as the representative of this family. In the training process, the model receives two chunks as input and learns to predict whether they belong to the same topic or not. To distinguish between two chunks in training a `[CLS]` token is inserted at the beginning of the first chunk, and a `[SEP]` token at the end of both the first and second chunk. A simple classification layer uses the embedding of the `[CLS]` token, learned during fine-tuning, for the final prediction. Since the input size is limited to a maximum of 512 tokens, shorter than many sections and paragraphs in our dataset, any longer chunk of text is truncated.

Sentence-Transformers (SRoBERTa) aims to enhance the sentence embeddings by alteration of RoBERTa using a Siamese architecture to derive semantically meaningful sentence embeddings (Reimers and Gurevych, 2019). Their method is available for several Transformer models. We choose a RoBERTa-based variant to make the results comparable to the first approach. SRoBERTa allows for specific new tasks, such as large-scale semantic similarity comparison, faster inference, and better representation for sentence-pair tasks. Moreover, because of the Siamese structure and coupling of two RoBERTa networks, the input size doubles, which allows for longer sequences and thus more context. The sentence embeddings are derived from a pooling operation over the output of two models with tied weights. Sentence-Transformers introduce several learning objectives. We choose the classification objective function, i.e., binary cross-entropy loss.

3.2 Sequential Inference

For inference, we use the classifiers of the previous step as topic change detectors for text segmentation. Given a document $d \in D$ divided into consecutive paragraphs $P = (p_1, ...p_k)$, section breaks are

marked as where the paragraph's topic changes. Considering a Transformer TF as our classifier and two consecutive paragraph as our input, the classifier outputs the probability of the two paragraphs belonging to the same topic, independent of their surrounding context, e.g., $TF(p_1, p_2) = P(Topic(p_1) = Topic(p_2))$. Therefore, given sequences of paragraphs $p_1, ...p_k$, and the corresponding predicted labels $y = (y_1, ..., y_{k-1})$, a segmentation of the document is given by $k - 1$ predictions of TF, where $y_i = 0$ denotes the end of a segment by p_i. It is worth noting that the segmentation module operates on paragraphs only.

4 Terms-of-Service (ToS) Dataset

Due to data governance policies in many countries, it is mandated that commercial websites contain the necessary legal information for site users, mostly reachable via the landing page. We automatically extract the ToS from the Alexa 1M URL dataset[3] divided into paragraphs and respective hierarchical section headings. We limit ourselves to only English websites and disregard any ToS where the majority of the text is in a different language. Although HTML is a structured format, it is a non-trivial task to extract text and hierarchies. Mainly because Web pages often contain a lot of boilerplate (e.g., navigational elements, advertisements, etc.), and that websites do not always conform to the HTML standards. For boilerplate removal, we use the `boilerpipe` package by Kohlschütter et al. (Kohlschütter et al., 2010), which is based on shallow text features for classifying the text elements on a Web page. Moreover, to deal with websites that do not conform to HTML standards, we perform several cleanup steps, including fixing mistakes such as text appearing without a corresponding paragraph (`<p>` tag), or incorrectly nested tags (e.g., section headings within a `<p>` tag). We fix such mistakes by adding missing tags and adjusting nested tags similar to how a web-browser would interpret the code. To obtain the hierarchy, we split the document into smaller chunks. Splits are done in the following order: first on section headings (`<h1>`-`<h6>` tags), then on bold text (`<b>` tag) starting with an enumeration pattern, then enumerations (`<li>` tags), then on underline text (`<u>` tag) starting with an enumeration pattern, and lastly on regular text (`<p>` tag) starting with an enumeration pattern. To prevent spurious splits, each criterion is only used if there are at least five occurrences within the document. Each time the document is split, we save the corresponding headings, which then form the content hierarchy. The majority of documents contain at most two levels of section hierarchy. The raw dataset consists of around 70,000 domain-specific documents.

[3] `http://s3.amazonaws.com/alexa-static/top-1m.csv.zip`

4.1 Data Cleaning

In addition to the full dataset, we provide a cleaned subset including paragraph annotations. We manually grouped 554 similar sections into 82 topics, and keep only sections that have at least one of these aliases as a heading. For the groups, we included any heading that has been occurring at least 250 times across the whole dataset, disregarding any ambiguous heading. The grouping is done manually by combining headings that infer the same topic into one. Different headings often contain spelling variations or synonyms words, e.g., "*limitation of liability*", "*limitations on liability*" and "*limitations of liability*" are all grouped under "*limitation of liability*". The full list of merges is present in our code. It is worth noting that the topic classification is only utilized for the same section prediction and random paragraph setting, since the consecutive sampling does not require topic information. For our work, we only group document content into top-level sections, and any further distinction is discarded, but is present in the raw data and may be used for future work. After removing documents without any valid sections, we are left with approximately 43,000 documents for the same section task, and around 40,000 documents for the paragraph setup. We randomly split the data 80/10/10 into the train, validation, and test set. The average number of sections per document is 6.56, and each document consists of 22.32 paragraphs on average. To our knowledge, this is the first large paragraph-based and publicly available text segmentation dataset that contains not only section annotations but also the present hierarchy in text.

5 Evaluation

We demonstrate the capabilities of Transformer-based architectures for topical change detection using a new dataset consisting of online Terms-of-Service (ToS) documents. Results are compared for the introduced IID STP task as well as a downstream comparison of text segmentation results to

a range of baselines and existing methods. Results show a great improvement in the performance for all Transformer-based models.

5.1 Evaluation of Models

We compare our methods against a range of baselines, including averaging over Global Vectors (*GLVavg*) (Pennington et al., 2014), tf-idf vectors (*tf-idf*), and Bag of Words (*BoW*) (Harris, 1954). For Transformer language models, we evaluate the standard `[CLS]` sequence classification with `roberta-base` (*Ro-CLS*). For Sentence-Transformers (Reimers and Gurevych, 2019) we use the Siamese Transformer setup with a variant of `roberta-base` (*ST-Ro*) and an additional model that has been pre-trained on NLI sentence similarity tasks (*ST-Ro-NLI*) to investigate performance of further pretraining.

Transformer models are trained using the HuggingFace Transformer library (Wolf et al., 2019) for the `[CLS]` models, and the Sentence-Transformers package (Reimers and Gurevych, 2019) for Siamese variants. We use two Nvidia Titan RTX GPUs for training, and each model variant has been trained with five different random seeds. Details for the training parameters can be found in our repository. Due to the length limitation of 512 tokens, we employ an iterative truncation strategy for two-sentence inputs. Coupling of two Transformers for the Sentence-Transformers doubles the input size, accepting inputs of up to 1024 tokens.

5.2 Prediction Tasks

As previously introduced, we train models with an independent classification setup, which is generally much faster than convoluted hierarchical sequential models. Specifically, we highlight the differences in the setup for the same section prediction task, compared to the two paragraph-based methods. Note that development and test sets differ for the sampling strategies in Table 1, and thus prohibit a direct comparison of the intermediate training results across different sampling strategies. We show in the subsequent section, however, that downstream performance is in line with results on the topic prediciton task.

Section (S) Topic Prediction. The section task showcases how different levels of granularity can affect outcomes in the prediction results. Specifically, the extremely long input sequences test the limits of what Transformers can predict from partial observations, since the majority of inputs will be heavily truncated. To ensure an equal distribution of samples from within the same and different sections, we match each section with three samples from the same topic, and three from different topics. The same strategy is employed for the generation of the development and test set.

Despite the constraints with respect to the input length, we find that all Transformers perform on a near-perfect level, compare Table 1. Comparing these results to already very well-performing baselines, we suspect that certain keywords give away similar sections, but highlight the fact that the explicit representation of different topics is not given during training in the binary classification task, which makes this a suitable method for dealing with imbalanced topics.

Random Paragraph (RP) Topic Prediction. In contrast to the section-level task, we revert back to a more fine-grained distinction of paragraphs in a text. In the Random Paragraph setting, we still generate samples in a similar fashion, meaning we include three paragraphs from other documents with the same topic, and three negative samples from random paragraphs with different topics in other documents. Results show a sharp drop in the performance, which can come from a much narrower context of the paragraphs, as well as different test set samples compared to the section task. Solely the BoW model seems to be largely unaffected, which is simply due to its low performance in either setting.

Consecutive Paragraph (CP) Topic Prediction. To boost performance, we employ a sampling strategy inspired by Ein Dor et al. (Ein Dor et al., 2018). For their triplet loss, samples are generated from within the same document only, which can be directly translated into sampling from intra-document paragraphs. Note that this strategy also no longer requires any merging of topics across documents, as all relevant information is now contained within a single ToS. To ensure comparison across models, we stick to the same annotated subset of our ToS data. Again, results are only conclusive for the performance between models.

The result of different sampling strategies along with the performance of the baselines is shown in Table 1, where the Transformer-based models all outperform the baselines by significant margin. Among the baselines BoW has the worst perfor-

mance overall, with the accuracy close to random, indicating that distinct word occurrences are not a sufficient indicator. Average GloVe has the best performance of all baselines models, but is still behind the Transformers by a large margin. Despite the NLI-pretrained SRoBERTa model (ST-Ro-N) achieving better scores than the base model (ST-Ro) for most setups, the difference is insignificant and shows that the pre-training on sentence similarity tasks does not directly influence our topic prediction setup.

5.3 Text Segmentation

By generating a text segmentation over the paragraphs of a full document, the independent prediction results from the previous section can now be compared across several approaches. Specifically, we compare the paragraph-based training methods CP and RP. As an evaluation metric, we follow related literature and adopt the P_k metric introduced by Beeferman et al. (Beeferman et al., 1999), which is the error rate of two segments at k sentences apart being classified incorrectly. We use the default window size of half the document length for our evaluation, again following related work. Furthermore, we count the number of explicit misclassifications, and use the accuracy acc_k of "up to k mistakes per document" as an evaluation metric. Due to the coarser nature of paragraphs and lower number of predictions per document compared to the sentence-level segmentation, this is a more illustrative metric. It also relates to the "exact match" metric $\mathbf{EM}_{outline}$ employed by Zhang et al. (Zhang et al., 2019), where $acc_0 = \mathbf{EM}_{outline}$.

Here, we also include performance of related works where public and up-to-date code repositories are available. Specifically, we compare to GraphSeg (Glavas et al., 2016), and the model by Koshorek et al. (Koshorek et al., 2018), which we dub "WikiSeg". Both approaches are trained on a sentence-level approach, though, and predictions have to be translated back to a paragraph level for comparison of results. We train each model with the suggested parameters in the respective repositories. Code for TSM (Du et al., 2013) and TopicTiling (Riedl and Biemann, 2012b) was also available, but only for outdated Java versions. Despite considerable efforts, we were unable to obtain meaningful results from both implementations.

For an additional pseudo-sequential baseline, we use an informed random oracle that has a-priori information on the number of topics in the document, and samples with adjusted $P(\text{"next section"}) = \#sections/\#paragraphs$. Note that no additional parameters are learned for any model, and predictions are binarized with a simple 0.5 threshold over the predictions. We provide ensembling results for the majority voting decisions by the five seed runs of each model variant (*Ens*), which provides further improvement on the score. Best results are obtained by ensembling all consecutive Transformer-based methods (*Ens consec*).

Table 2 shows the results of the evaluation, where we can see that results in the sequential segmentation are directly linked to the performance on the independent classification task seen in Table 1. The importance of training strategy shows itself in the comparison of CP and RP models. In general, CP training setup yields better P_k scores across all models, partly because the intra-document dependencies are captured better with this sampling strategy. Especially the Ro-CLS was very inconsistent during RP runs, and only converged in some cases, which led to detrimental performance. Additional pre-training of ST models does not show any significant improvement in the performance, but all ST models outperform the `[CLS]` model as well as the baselines. To our surprise, both GraphSeg and WikiSeg show a significantly lower performance, and fall even behind the simpler baselines. For GraphSeg, the unsupervised approach on a per-document basis seems to significantly prohibit correct predictions on primarily short documents. WikiSeg heavily preprocesses the data and discards many samples, thus significantly shrinking the training set. Since performance on the reduced training set is decent, this indicates that training a network from scratch is not suitable with the smaller training set of a reduced corpus and tends to overfit. We expect a significant increase in performance if the training would instead be performed on a paragraph-level without such strict preprocessing criteria.

Results for acc_k in Figure 2 indicate a correlation with the P_k measure, something that was prohibitive for sentence-level evaluations. Overall, our largest ensemble classifies around 25% of documents without any mistake (acc_0), and around 70% with up to two mistakes (acc_2).

Table 1: Prediction accuracy for the independent topic prediction tasks, Same Topic Prediction (STP), Random Paragraph (RP), Consecutive Paragraph (CP) with different sampling strategies. Standard deviation is reported over 5 runs and the best model on each respective set is depicted in bold. Development and test sets vary between sampling strategies S, RP and CP.

		GLVavg	tf-idf	BoW	Ro-CLS	ST-Ro	ST-Ro-N
S	Dev	89.70 ±.07	82.10 ±.05	50.94 ±.33	96.42 ±.52	96.38 ±.03	**96.39 ±.03**
	Test	90.01 ±.06	82.54 ±.07	51.05 ±.51	96.58 ±.52	96.45 ±.06	**96.46 ±.02**
RP	Dev	76.63 ±.04	70.94 ±.07	50.34 ±.04	57.63 ±10.4	**87.50 ±.13**	87.39 ±.08
	Test	76.16 ±.06	70.41 ±.09	50.31 ±.37	57.48 ±10.2	**87.19 ±.64**	86.88 ±.11
CP	Dev	77.64 ±6.6	74.94 ±.11	56.34 ±.83	89.63 ±.12	**91.17 ±.05**	91.12 ±.04
	Test	78.63 ±6.8	76.17 ±.07	56.58 ±1.1	90.34 ±.08	91.17 ±.04	**91.69 ±.02**

Table 2: Boundary error rate P_k for compared models (lower is better), based on sampling strategies Random Paragraph (RP), Consecutive Paragraph (CP) and their Ensemble variates, RP$_{Ens}$ and CP$_{Ens}$, respectively. Ensemble ("*Ens*") predictions are obtained by majority voting over model runs.

	RP	CP	RP$_{Ens}$	CP$_{Ens}$
GLVavg	29.97 ±.09	26.23 ±6.2	29.55	23.06
tf-idf	39.87 ±.24	29.70 ±.28	39.36	28.60
BoW	45.76 ±.67	43.46 ±1.5	46.20	41.80
Random Oracle	35.08 ±.15	-	31.88	-
GraphSeg	-	32.48 ±.46	-	32.28
WikiSeg	-	48.29 ±.30	-	48.29
Ro-CLS	37.26 ±4.8	15.15 ±.00	41.15	15.15
ST-Ro	15.72 ±.11	14.06 ±.14	**14.62**	13.14
ST-Ro-N	15.97 ±.14	13.97 ±.19	14.81	**12.95**
Ens consec	-	-	-	12.50

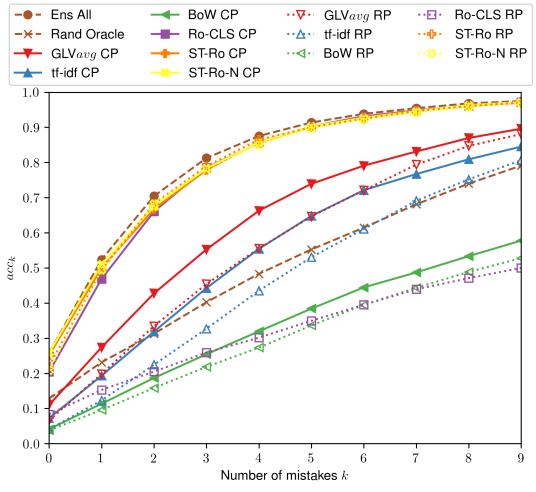

Figure 2: Mistake rate of per-model ensembles.

6 Conclusion and Future Work

Despite a multitude of previous works, text segmentation methods have generally focused on very finely segmented text chunks in the form of sentences. In this work, we have shown that a relaxation of this problem to coarser text structures reduces the complexity of the problem, while still allowing for a meaningful semantic segmentation. Further, we reformulate the sequential setup of text segmentation as a supervised Same Topic Prediction task, which reduces training time, while allowing for a near-trivial generation of samples from automatically crawled text documents and the utilization of large pre-trained language models. To show the applicability of our method, we present a new domain-specific and large corpus of online Terms-of-Service documents, and train Transformer-based models that vastly outperform a number of text segmentation baselines.

We are currently investigating the setup for hierarchical sections, which our dataset allows for, to see whether such notions can also be picked up by an independent classifier. Since our Consecutive Paragraph model requires no additional document annotation, it is also suitable for larger-scale studies on cross-domain collections.

References

Doug Beeferman, Adam L. Berger, and John D. Lafferty. 1999. Statistical Models for Text Segmentation. *Mach. Learn.*, 34(1-3):177–210.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *J. Mach. Learn. Res.*, 3:993–1022.

Thorsten Brants, Francine Chen, and Ioannis Tsochantaridis. 2002. Topic-based Document Segmentation with Probabilistic Latent Semantic Analysis. In *Proceedings of the 2002 ACM CIKM International Conference on Information and Knowledge Management, McLean, VA, USA*, pages 211–218. ACM.

Harr Chen, S. R. K. Branavan, Regina Barzilay, and David R. Karger. 2009. Global Models of Document Structure using Latent Permutations. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, 2009, Boulder, Colorado, USA*, pages 371–379.

Freddy Y. Y. Choi. 2000a. Advances in Domain Independent Linear Text Segmentation. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference*, NAACL 2000, page 26–33, USA. ACL.

Freddy Y. Y. Choi. 2000b. Advances in Domain Independent Linear Text Segmentation. In *6th Applied Natural Language Processing Conference, ANLP, Seattle, Washington, USA, 2000*, pages 26–33. ACL.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT , Minneapolis, USA, 2019, Volume 1*, pages 4171–4186.

Satya Dharanipragada, Martin Franz, J. Scott McCarley, Salim Roukos, and Todd Ward. 1999. Story Segmentation and Topic Detection for Recognized Speech. In *Sixth European Conference on Speech Communication and Technology, EUROSPEECH 1999, Budapest, Hungary*. ISCA.

Adji B. Dieng, Francisco J. R. Ruiz, and David M. Blei. 2019. Topic Modeling in Embedding Spaces. *CoRR*, abs/1907.04907.

Lan Du, Wray L. Buntine, and Mark Johnson. 2013. Topic Segmentation with a Structured Topic Model. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pages 190–200. The Association for Computational Linguistics.

Liat Ein Dor, Yosi Mass, Alon Halfon, Elad Venezian, Ilya Shnayderman, Ranit Aharonov, and Noam Slonim. 2018. Learning Thematic Similarity Metric from Article Sections Using Triplet Networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 49–54, Melbourne, Australia. ACL.

Jacob Eisenstein and Regina Barzilay. 2008. Bayesian Unsupervised Topic Segmentation. In *2008 Conference on Empirical Methods in Natural Language Processing, EMNLP 2008, Proceedings of the Conference, 25-27 October 2008, Honolulu, Hawaii, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 334–343. ACL.

Goran Glavas, 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, *SEM@ACL, Berlin, Germany, 2016*. The *SEM 2016 Organizing Committee.

Goran Glavas and Swapna Somasundaran. 2020. Two-Level Transformer and Auxiliary Coherence Modeling for Improved Text Segmentation. *CoRR*, abs/2001.00891.

Thomas L. Griffiths, Mark Steyvers, David M. Blei, and Joshua B. Tenenbaum. 2004. Integrating Topics and Syntax. In *Advances in Neural Information Processing Systems 17 [Neural Information Processing Systems, 2004, British Columbia, Canada]*, pages 537–544.

Zellig S. Harris. 1954. Distributional structure. *WORD*, 10(2-3):146–162.

Marti A. Hearst. 1994. Multi-Paragraph Segmentation of Expository Text. In *32nd Annual Meeting of the Association for Computational Linguistics, 1994, Las Cruces, New Mexico, USA, Proceedings*, pages 9–16. ACL.

Marti A. Hearst. 1997. TextTiling: Segmenting Text into Multi-Paragraph Subtopic Passages. *Comput. Linguist.*, 23(1):33–64.

Thomas Hofmann. 2017. Probabilistic Latent Semantic Indexing. *SIGIR Forum*, 51(2):211–218.

Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. 2010. Boilerplate Detection Using Shallow Text Features. In *Proceedings of the Third International Conference on Web Search and Web Data Mining, WSDM, New York, USA 2010*, pages 441–450. ACM.

Omri Koshorek, Adir Cohen, Noam Mor, Michael Rotman, and Jonathan Berant. 2018. Text Segmentation as a Supervised Learning Task. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New*

Orleans, Louisiana, USA, 2018, Volume 2 (Short Papers), pages 469–473. ACL.

Hideki Kozima. 1993. Text Segmentation Based on Similarity between Words. In *31st Annual Meeting of the Association for Computational Linguistics, 1993, Ohio State University, Columbus, Ohio, USA, Proceedings*, pages 286–288. ACL.

Jing Li, Aixin Sun, and Shafiq R. Joty. 2018. Segbot: A generic neural text segmentation model with pointer network. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*, pages 4166–4172. ijcai.org.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Igor Malioutov and Regina Barzilay. 2006. Minimum Cut Model for Spoken Lecture Segmentation. In *ACL, 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, Sydney, Australia, 2006*. ACL.

Hemant Misra, François Yvon, Olivier Cappé, and Joemon M. Jose. 2011. Text Segmentation: A Topic Modeling Perspective. *Inf. Process. Manag.*, 47(4):528–544.

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, CIKM , Hong Kong, China, 2009*, pages 1553–1556.

Christopher E. Moody. 2016. Mixing Dirichlet Topic Models and Word Embeddings to Make lda2vec. *CoRR*, abs/1605.02019.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP, Hong Kong, China*, pages 3980–3990. ACL.

Martin Riedl and Chris Biemann. 2012a. How Text Segmentation Algorithms Gain from Topic Models. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, 2012, Montréal, Canada*, pages 553–557. ACL.

Martin Riedl and Chris Biemann. 2012b. TopicTiling: A Text Segmentation Algorithm based on LDA. In *Proceedings of the Student Research Workshop of the 50th Meeting of the Association for Computational Linguistics*, pages 37–42, Republic of Korea.

Masao Utiyama and Hitoshi Isahara. 2001. A statistical model for domain-independent text segmentation. In *Association for Computational Linguistic, 39th Annual Meeting and 10th Conference of the European Chapter, Proceedings of the Conference, 2001, Toulouse, France*, pages 491–498. ACL.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA*, pages 5998–6008.

Hanna M. Wallach. 2006. Topic modeling: Beyond Bag-of-Words. In *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML), Pittsburgh, Pennsylvania, USA, 2006*, volume 148 of *ACM International Conference Proceeding Series*, pages 977–984.

Ross Wilkinson. 1994. Effective Retrieval of Structured Documents. In *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval. Dublin, Ireland, 1994 (Special Issue of the SIGIR Forum)*, pages 311–317. ACM/Springer.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv*, abs/1910.03771.

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, SIGIR, Paris, France, 2019*, pages 745–754.

A Appendices

We additionally provide a longer explanation of the crawling process which we used to obtain the Terms-of-Service dataset with.
Terms of Services (ToS) must be easily reachable via the landing page, which makes it comparatively easy to be crawled. For each ToS, we automatically extract the content divided into paragraphs and respective hierarchical section headings.

A.1 Crawling

As seeds to our crawler we use the Alexa 1M URL dataset, see the reference in the main paper. For each URL in the dataset, we try to access the website both with and without the `www` prefix. First the landing page is downloaded and parsed using the *Beautiful Soup* Python package. We then search for hyperlinks with texts *Terms of Service*, *Terms of Use*, *Terms and Conditions*, and *Conditions of Use*, and follow them to get to the respective terms-of-service pages. Levenshtein distance with a threshold of 0.75 is used to allow for spelling mistakes and different wording (e.g., *Terms & Conditions* instead of *Terms and Conditions*). The raw Hypertext Markup Language (HTML) content of the terms-of-service page is downloaded and stored for further processing. In case of an error, e.g., if the website is temporarily unreachable, we retry the same website 2 additional times before skipping it. The unprocessed dataset contains HTML code for roughly 74,000 websites. Note that due to limitations of the current crawler implementation, websites that rely on JavaScript to display content are not supported. The idea of using `Selenium` was discarded due to the significant overhead in crawling time.

A.2 Section Extraction

Despite the fact that HTML is a structured format, it is a non-trivial task to extract text and hierarchies. The main two reasons are that Web pages often contain a lot of boilerplate (e.g., navigational elements, advertisements, etc.), and that websites do not always conform to the HTML standards. Here, only a rough outline of the pipeline is given. For further reference, please refer to the reference implementation in our repository.

Boilerplate Removal. For boilerplate removal we use the `boilerpipe` package by Kohlschütter et al. (Kohlschütter et al., 2010), which is based on shallow text features for classifying the text elements on a Web page. The result is an HTML page with all navigational elements, advertisements, and template code removed. Most importantly though, the structure we are interested in is preserved in this step.

HTML Cleanup. To deal with websites that do not conform to HTML standards, we perform several cleanup steps. This includes, for example, fixing mistakes such as text appearing without a corresponding paragraph (`<p>` tag), or incorrectly nested tags (e.g., section headings within a `<p>` tag). We fix such mistakes by adding missing tags and adjusting nested tags similar to how a web-browser would interpret the code.

Language Detection. Since the Alexa dataset also contains many non-English websites, we reject extracted terms-of-services, where the majority of text most likely has a different language. We use the `langid` Python package for detecting the language of each paragraph (`<p>` tag).

Extracting Hierarchy. To obtain the hierarchy, we split the document into smaller chunks. Splits are done in the following order: first we split on each section heading (`<h1>`-`<h6>` tags), then on bold text (`<b>` tag) starting with an enumeration pattern, then on enumerations (`<li>` tags), then on underline text (`<u>` tag) starting with an enumeration pattern, and lastly on regular text (`<p>` tag) starting with an enumeration pattern. To prevent spurious splits, each criteria is only used if there are at least 5 occurrences within the document. Each time the document is split, we save the corresponding headings, which then form the hierarchy. As enumeration patterns we recognize Latin numbers, roman numerals, and letters, optionally prefixed with *Part*, *Section*, or *Article*. The majority of documents contain at most two levels of section hierarchy.