Text Segmentation as a Supervised Learning Task

Omri Koshorek[*]   **Adir Cohen**[*]   **Noam Mor**   **Michael Rotman**   **Jonathan Berant**
School of Computer Science
Tel-Aviv University, Israel
{omri.koshorek,adir.cohen,noam.mor,michael.rotman,joberant}@cs.tau.ac.il

Abstract

Text segmentation, the task of dividing a document into contiguous segments based on its semantic structure, is a longstanding challenge in language understanding. Previous work on text segmentation focused on unsupervised methods such as clustering or graph search, due to the paucity in labeled data. In this work, we formulate text segmentation as a supervised learning problem, and present a large new dataset for text segmentation that is automatically extracted and labeled from Wikipedia. Moreover, we develop a segmentation model based on this dataset and show that it generalizes well to unseen natural text.

1 Introduction

Text segmentation is the task of dividing text into segments, such that each segment is topically coherent, and cutoff points indicate a change of topic (Hearst, 1994; Utiyama and Isahara, 2001; Brants et al., 2002). This provides basic structure to a document in a way that can later be used by downstream applications such as summarization and information extraction.

Existing datasets for text segmentation are small in size (Choi, 2000; Glavaš et al., 2016), and are used mostly for evaluating the performance of segmentation algorithms. Moreover, some datasets (Choi, 2000) were synthesized automatically and thus do not represent the natural distribution of text in documents. Because no large labeled dataset exists, prior work on text segmentation tried to either come up with heuristics for identifying whether two sentences discuss the same topic (Choi, 2000; Glavaš et al., 2016), or to model topics explicitly with methods such as LDA (Blei et al., 2003) that assign a topic to each paragraph or sentence (Chen et al., 2009).

Recent developments in Natural Language Processing have demonstrated that casting problems as supervised learning tasks over large amounts of labeled data is highly effective compared to heuristic-based systems or unsupervised algorithms (Mikolov et al., 2013; Pennington et al., 2014). Therefore, in this work we (a) formulate text segmentation as a supervised learning problem, where a label for every sentence in the document denotes whether it ends a segment, (b) describe a new dataset, WIKI-727K, intended for training text segmentation models.

WIKI-727K comprises more than 727,000 documents from English Wikipedia, where the table of contents of each document is used to automatically segment the document. Since this dataset is large, natural, and covers a variety of topics, we expect it to generalize well to other natural texts. Moreover, WIKI-727K provides a better benchmark for evaluating text segmentation models compared to existing datasets. We make WIKI-727K and our code publicly available at https://github.com/koomri/text-segmentation.

To demonstrate the efficacy of this dataset, we develop a hierarchical neural model in which a lower-level bidirectional LSTM creates sentence representations from word tokens, and then a higher-level LSTM consumes the sentence representations and labels each sentence. We show that our model outperforms prior methods, demonstrating the importance of our dataset for future progress in text segmentation.

2 Related Work

2.1 Existing Text Segmentation Datasets

The most common dataset for evaluating performance on text segmentation was created by Choi (2000). It is a synthetic dataset containing 920 documents, where each document is a concatena-

---

[*]Both authors contributed equally to this paper and the order of authorship was determined randomly.

tion of 10 random passages from the Brown corpus. Glavaš et al. (2016) created a dataset of their own, which consists of 5 manually-segmented political manifestos from the Manifesto project.[1] (Chen et al., 2009) also used English Wikipedia documents to evaluate text segmentation. They defined two datasets, one with 100 documents about major cities and one with 118 documents about chemical elements. Table 1 provides additional statistics on each dataset.

Thus, all existing datasets for text segmentation are small and cannot benefit from the advantages of training supervised models over labeled data.

2.2 Previous Methods

Bayesian text segmentation methods (Chen et al., 2009; Riedl and Biemann, 2012) employ a generative probabilistic model for text. In these models, a document is represented as a set of topics, which are sampled from a topic distribution, and each topic imposes a distribution over the vocabulary. Riedl and Biemann (2012) perform best among this family of methods, where they define a coherence score between pairs of sentences, and compute a segmentation by finding drops in coherence scores between pairs of adjacent sentences.

Another noteworthy approach for text segmentation is GRAPHSEG (Glavaš et al., 2016), an unsupervised graph method, which performs competitively on synthetic datasets and outperforms Bayesian approaches on the Manifesto dataset. GRAPHSEG works by building a graph where nodes are sentences, and an edge between two sentences signifies that the sentences are semantically similar. The segmentation is then determined by finding maximal cliques of adjacent sentences, and heuristically completing the segmentation.

3 The WIKI-727K Dataset

For this work we have created a new dataset, which we name WIKI-727K. It is a collection of 727,746 English Wikipedia documents, and their hierarchical segmentation, as it appears in their table of contents. We randomly partitioned the documents into a train (80%), development (10%), and test (10%) set.

Different text segmentation use-cases require different levels of granularity. For example, for segmenting text by overarching topic it makes sense to train a model that predicts only top-level

segments, which are typically vary in topic – for example, *"History"*, *"Geography"*, and *"Demographics"*. For segmenting a radio broadcast into separate news stories, which requires finer granularity, it makes sense to train a model to predict sub-segments. Our dataset provides the entire segmentation information, and an application may choose the appropriate level of granularity.

To generate the data, we performed the following preprocessing steps for each Wikipedia document:

- Removed all photos, tables, Wikipedia template elements, and other non-text elements.
- Removed single-sentence segments, documents with less than three segments, and documents where most segments were filtered.
- Divided each segment into sentences using the PUNKT tokenizer of the NLTK library (Bird et al., 2009). This is necessary for the use of our dataset as a benchmark, as without a well-defined sentence segmentation, it is impossible to evaluate different models.

We view WIKI-727K as suitable for text segmentation because it is natural, open-domain, and has a well-defined segmentation. Moreover, neural network models often benefit from a wealth of training data, and our dataset can easily be further expanded at very little cost.

4 Neural Model for Text Segmentation

We treat text segmentation as a supervised learning task, where the input x is a document, represented as a sequence of n sentences $s_1, \ldots, s_n$, and the label $y = (y_1, \ldots, y_{n-1})$ is a segmentation of the document, represented by $n - 1$ binary values, where y_i denotes whether s_i ends a segment.

We now describe our model for text segmentation. Our neural model is composed of a hierarchy of two sub-networks, both based on the LSTM architecture (Hochreiter and Schmidhuber, 1997). The lower-level sub-network is a two-layer bidirectional LSTM that generates sentence representations: for each sentence s_i, the network consumes the words $w_1^{(i)}, \ldots, w_k^{(i)}$ of s_i one by one, and the final sentence representation e_i is computed by max-pooling over the LSTM outputs.

The higher-level sub-network is the segmentation prediction network. This sub-network takes a sequence of sentence embeddings $e_1, \ldots, e_n$ as input, and feeds them into a two-layer bidirectional
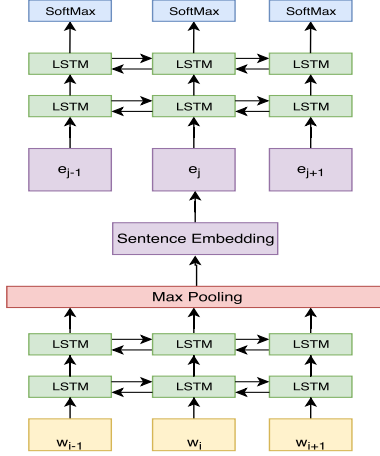
Figure 1: Our model contains a sentence embedding sub-network, followed by a segmentation prediction sub-network which predicts a cut-off probability for each sentence.

LSTM. We then apply a fully-connected layer on each of the LSTM outputs to obtain a sequence of n vectors in $\mathbb{R}^2$. We ignore the last vector (for e_n), and apply a *softmax* function to obtain $n-1$ segmentation probabilities. Figure 1 illustrates the overall neural network architecture.

4.1 Training

Our model predicts for each sentence s_i, the probability p_i that it ends a segment. For an n-sentence document, we minimize the sum of cross-entropy errors over each of the $n-1$ relevant sentences:

$$J(\Theta) = \sum_{i=1}^{n-1} \left[-y_i \log p_i - (1 - y_i) \log (1 - p_i) \right].$$

Training is done by stochastic gradient descent in an end-to-end manner. For word embeddings, we use the GoogleNews word2vec pre-trained model. We train our system to only predict the top-level segmentation (other granularities are possible). In addition, at training time, we removed from each document the first segment, since in Wikipedia it is often a summary that touches many different topics, and is therefore less useful for training a segmentation model. We also omitted lists and code snippets tokens.

4.2 Inference

At test time, the model takes a sequence of word embeddings divided into sentences, and returns a vector p of cutoff probabilities between sentences. We use greedy decoding, i.e., we create a new segment whenever p_i is greater than a threshold τ. We

optimize the parameter τ on our validation set, and use the optimal value while testing.

5 Experimental Details

We evaluate our method on the WIKI-727 test set, Choi's synthetic dataset, and the two small Wikipedia datasets (CITIES, ELEMENTS) introduced by Chen et al. (2009). We compare our model performance with those reported by Chen et al. (2009) and GRAPHSEG. In addition, we evaluate the performance of a random baseline model, which starts a new segment after every sentence with probability $\frac{1}{k}$, where k is the average segment size in the dataset.

Because our test set is large, it is difficult to evaluate some of the existing methods, which are computationally demanding. Thus, we introduce WIKI-50, a set of 50 randomly sampled test documents from WIKI-727K. We use WIKI-50 to evaluate systems that are too slow to evaluate on the entire test set. We also provide human segmentation performance results on WIKI-50.

We use the P_k metric as defined in Beeferman et al. (1999) to evaluate the performance of our model. P_k is the probability that when passing a sliding window of size k over *sentences*, the sentences at the boundaries of the window will be incorrectly classified as belonging to the same segment (or vice versa). To match the setup of Chen et al. (2009), we also provide the P_k metric for a sliding window over *words* when evaluating on the datasets from their paper. Following (Glavaš et al., 2016), we set k to half of the average segment size in the ground-truth segmentation. For evaluations we used the SEGEVAL package (Fournier, 2013).

In addition to segmentation accuracy, we also report runtime when running on a mid-range laptop CPU.

We note that segmentation results are not always directly comparable. For example, Chen et al. (2009) require that all documents in the dataset discuss the same topic, and so their method is not directly applicable to WIKI-50. Nevertheless, we attempt a comparison in Table 2.

5.1 Accuracy

Comparing our method to GRAPHSEG, we can see that GRAPHSEG gives better results on the synthetic Choi dataset, but this success does not carry over to the natural Wikipedia data, where they underperform the random baseline. We ex-

Table 1: Statistics on various text segmentation datasets.

	WIKI-727K	CHOI	MANIFESTO	CITIES	ELEMENTS
Documents	727,746	920	5	100	118
Segment Length[2]	13.6 ± 20.3	7.4 ± 2.96	8.99 ± 10.8	5.15 ± 4.57	3.33 ± 3.05
Segments per document[2]	3.48 ± 2.23	9.98 ± 0.12	127 ± 42.9	12.2 ± 2.79	6.82 ± 2.57
Real-world	✓	✗	✓	✓	✓
Large variety of topics	✓	✗	✗	✗	✗

Table 2: P_k Results on the test set.

P_k variant	WIKI-727K sentences	WIKI-50 sentences	CHOI sentences	CITIES sentences	CITIES words	ELEMENTS sentences	ELEMENTS words
(Chen et al., 2009)	-	-	-	-	22.1	-	**20.1**
GraphSeg	-	63.56	**5.6-7.2**	39.95	-	49.12	-
Our model	22.13	**18.24**	26.26[3]	**19.68**	**18.14**	**41.63**	33.82
Random baseline	53.09	52.65	49.43	47.14	44.14	50.08	42.80
Human performance	-	14.97	-	-	-	-	-

plain this by noting that since the dataset is synthetic, and was created by concatenating unrelated documents, even the simple word counting method in Choi (2000) can achieve reasonable success. GRAPHSEG uses a similarity measure between word embedding vectors to surpass the word counting method, but in a natural document, word similarity may not be enough to detect a change of topic within a single document. At the word level, two documents concerning completely different topics are much easier to differentiate than two sections in one document.

We compare our method to Chen et al. (2009) on the two small Wikipedia datasets from their paper. Our method outperforms theirs on CITIES and obtains worse results on ELEMENTS, where presumably our word embeddings were of lower quality, having been trained on Google News, where one might expect that few technical words from the domain of Chemistry are used. We consider this result convincing, since we did not exploit the fact that all documents have similar structure as Chen et al. (2009), and did not train specifically for these datasets, but still were able to demonstrate competitive performance.

Interestingly, human performance on WIKI-50 is only slightly better than our model. We assume that because annotators annotated only a small number of documents, they still lack familiarity with the right level of granularity for segmentation, and are thus at a disadvantage compared to the model that has seen many documents.

5.2 Run Time

Our method's runtime is linear in the number of words and the number of sentences in a docu-

ment. Conversely, GRAPHSEG has a much worse asymptotic complexity of $O(N^3 + V^k)$ where N is the length of the longest sentence, V the number of sentences, and k the largest clique size. Moreover, neural network models are highly parallelizable, and benefit from running on GPUs.

In practice, our method is much faster than GRAPHSEG. In Table 3 we report the average run time per document on WIKI-50 on a CPU.

Table 3: Average run time in seconds per document.

	WIKI-50
Our model (CPU)	1.6
GRAPHSEG (CPU)	23.6

6 Conclusions

In this work, we present a large labeled dataset, WIKI-727K, for text segmentation, that enables training neural models using supervised learning methods. This closes an existing gap in the literature, where thus far text segmentation models were trained in an unsupervised fashion.

Our text segmentation model outperforms prior methods on Wikipedia documents, and performs competitively on prior benchmarks. Moreover, our system has linear runtime in the text length, and can be run on modern GPU hardware. We argue that for text segmentation systems to be useful in the real world, they must be able to segment arbitrary natural text, and this work provides a path towards achieving that goal.

In future work, we will explore richer neural models at the sentence-level. Another important

---

[2]Statistics on top-level segments.
[3]We optimized τ by cross validation on the CHOI dataset.

direction is <mark>developing a structured global model that will take all local predictions into account and then perform a global segmentation decision</mark>.

Acknowledgements

References

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

S. Bird, E. Loper, and E. Klein. 2009. *Natural Language Processing with Python*. OReilly Media Inc.

D. Blei, A. Ng, and M. I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research (JMLR)* 3:993–1022.

Thorsten Brants, Francine Chen, and Ioannis Tsochantaridis. 2002. Topic-based document segmentation with probabilistic latent semantic analysis. In *Proceedings of the eleventh international conference on Information and knowledge management*. ACM, pages 211–218.

Harr Chen, SRK Branavan, Regina Barzilay, and David R Karger. 2009. Global models of document structure using latent permutations. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 371–379.

Freddy YY Choi. 2000. Advances in domain independent linear text segmentation. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*. Association for Computational Linguistics, pages 26–33.

Chris Fournier. 2013. Evaluating Text Segmentation using Boundary Edit Distance. In *Proceedings of 51st Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, page to appear.

Goran Glavaš, Federico Nanni, and Simone Paolo Ponzetto. 2016. Unsupervised text segmentation using semantic relatedness graphs. Association for Computational Linguistics.

Marti A Hearst. 1994. Multi-paragraph segmentation of expository text. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, pages 9–16.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

T. Mikolov, W. Yih, and G. Zweig. 2013. Linguistic regularities in continuous space word representations. In *hlt-Naacl*. volume 13, pages 746–751.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*. pages 1532–1543. http://www.aclweb.org/anthology/D14-1162.

Martin Riedl and Chris Biemann. 2012. Topictiling: a text segmentation algorithm based on lda. In *Proceedings of ACL 2012 Student Research Workshop*. Association for Computational Linguistics, pages 37–42.

Masao Utiyama and Hitoshi Isahara. 2001. A statistical model for domain-independent text segmentation. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, pages 499–506.