How Text Segmentation Algorithms Gain from Topic Models

Martin Riedl and **Chris Biemann**
Ubiquitous Knowledge Processing Lab
Computer Science Department, Technische Universität Darmstadt
Hochschulstrasse 10, D-64289 Darmstadt, Germany
riedl@ukp.informatik.tu-darmstadt.de, biem@cs.tu-darmstadt.de

Abstract

This paper introduces a general method to incorporate the LDA Topic Model into text segmentation algorithms. We show that semantic information added by Topic Models significantly improves the performance of two word-based algorithms, namely TextTiling and C99. Additionally, we introduce the new TopicTiling algorithm that is designed to take better advantage of topic information. We show consistent improvements over word-based methods and achieve state-of-the art performance on a standard dataset.

1 Introduction

Texts are often structured into segments to ease understanding and readability of texts. Knowing about sentence boundaries is advantageous for natural language processing (NLP) tasks such as summarization or indexing. While many genres such as encyclopedia entries or scientific articles follow rather formal conventions of breaking up a text into meaningful units, there are plenty of electronically available texts without defined segments, e.g. web documents. *Text segmentation* is the task of automatically segmenting texts into parts. Viewing a well-written text as sequence of subtopics and assuming that subtopics correspond to segments, a segmentation algorithm needs to find changes of subtopics to identify the natural division of an unstructured text.

In this work, we utilize semantic information from *Topic Models (TMs)* to inform text segmentation algorithms. For this, we compare two early word-based algorithms with their topic-based variants, and construct our own algorithm called *TopicTiling*. We show that using topics estimated by *Latent Dirichlet Allocation (LDA)* in lieu of words substantially improves earlier segmentation algorithms. In comparison to *TextTiling (TT)*, neither smoothing nor a blocksize or window size is needed. TT using TMs and our own algorithm improve on the state-of-the-art for a standard dataset, while being conceptually simpler and computationally more efficient than other topic-based segmentation algorithms.

2 Related Work

Based on the observation of Halliday and Hasan (1976) that the density of coherence relations is higher within segments than between segments, most algorithms compute a coherence score to measure the difference of textual units for informing a segmentation decision. *TextTiling (TT)* (Hearst, 1994) relies on the simplest coherence relation – word repetition – and computes similarities between textual units based on the similarities of word space vectors. With *C99* (Choi, 2000) an algorithm was introduced that uses a matrix-based ranking and a clustering approach in order to relate the most similar textual units and to cluster groups of consecutive units into segments. Both *TT* and *C99* characterize textual units by the words they contain. Galley et al. (2003) showed that using TF-IDF term weights in the term vector improves the performance of *TT*. Proposals using Dynamic Programming (DP) are given in (Utiyama and Isahara, 2001; Fragkou et al., 2004). Related to our work are the approaches described in (Misra et al., 2009; Sun et al., 2008): here, TMs are also used to alleviate the sparsity of word vectors. Misra et al. (2009) extended the DP algorithm U00 from Utiyama and Isahara (2001) us-

2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 553–557,
Montréal, Canada, June 3-8, 2012.

ing TMs. At this, the topic assignments have to be inferred for each possible segment, resulting in high computational cost. In addition to these linear topic segmentation algorithms, there are hierarchical segmentation algorithms, see (Yaari, 1997; Hsueh et al., 2006; Eisenstein, 2009).

For topic modeling, we use the widely applied LDA (Blei et al., 2003). This generative probabilistic model uses a training corpus of documents to create document-topic and topic-word distributions and is parameterized by the number of topics N as well as by two hyperparameters. To generate a document d the topic proportions are drawn using a Dirichlet distribution with hyperparameter α. Adjacent for each word i a topic z_{d_i} is chosen according to a multinomial distribution using hyperparameter $\beta_{z_{d_i}}$. Unseen documents can be annotated with an existing TM using Bayesian inference methods (here: Gibbs sampling).

3 Method: From Words to Topics

The underlying mechanism described here is very simple: Instead of using words directly as features to characterize textual units, we use the topic IDs assigned by Bayesian inference. LDA inference assigns a topic ID to each word in the test document in each inference iteration step, based on a TM estimated on a training corpus. We use the topic ID, lastly assigned to each word. This might lead to instabilities as a word with high probabilities for several topics could be assigned to different topics in different inference iterations. To avoid these instabilities, we save all topic IDs assigned to a word for each inference iteration. Finally, the most frequent topic ID is assigned to each word. This mechanism we call the *mode method*. Both word replacements can be applied to most segmentation algorithms.

In this work, we use this general setup to implement topic-based versions of *TT* and *C99* and develop a new TextTiling-based method called *TopicTiling*.

4 Topic-based Segmentation Algorithms

4.1 TextTiling using Topic Models

In *TextTiling (TT)* (Hearst, 1994) using topic IDs (*TTLDA*), a document D, which is subject to segmentation, is represented as a sequence of n topic IDs[1]. *TT* splits the document into *topic-sequences*, instead of sentences, where each sequence consists of w topic IDs. To calculate the similarity between two topic-sequences, called *sequence-gap*, *TT* uses k topic-sequences, named *block*, to the left and to the right of the sequence gap. This parameter k defines the so-called *blocksize*. The cosine similarity is applied to computed a similarity score based on the topic frequency of the adjacent blocks at each sequence-gap. A value close to 1 indicates a high similarity among two blocks, a value close to zero denotes a low similarity. Then for each sequence-gap a *depth score* d_i is calculated for describing the sharpness of a gap, by $d_i = 1/2(hl(i) - s_i + hr(i) - s_i)$. The function $hl(i)$ returns the highest similarity score on the left side of the sequence-gap index i that does not increase and $hr(i)$ returns the highest score on the right side. Then all local maxima positions are searched based on the depth scores.

In the next step, these obtained maxima scores are sorted. If the number of segments n is given as input parameter, the n highest depth scores are used, otherwise a cut-off function is used that applies a segment only if the depth score is larger than $\mu - \sigma/2$, where mean μ and the standard deviation σ are calculated based on the entirety of depth scores. As *TT* calculates the depth on every topic-sequence using the highest gap, this could lead to a segmentation in the middle of a sentence. To avoid this, a final step ensures that the segmentation is positioned at the nearest sentence boundary.

4.2 C99 using Topic Models

For the C99 algorithm (Choi, 2000), named (*C99LDA*) when using topic IDs, the text is divided into minimal units on sentence boundaries. A similarity matrix $S_{m \times m}$ is computed, where m denotes the number of units (sentences). Every element s_{ij} is calculated using the cosine similarity between unit i and j. Next, a rank matrix R is computed to improve the contrast of S: Each element r_{ij} contains the number of neighbors of s_{ij} that have lower similarity scores then s_{ij} itself. In a final step a top-down clustering algorithm is performed to split the document into m segments $B = b_1, \dots, b_m$. This algo-

[1] words instead of topic IDs are utilized in the original approach.

rithm starts with the whole document considered as one segment and splits off segments until the stop criteria are met, e.g. the number of segments or a similarity threshold.

4.3 TopicTiling

TopicTiling is a new TextTiling-based algorithm and is adjusted to use TMs. As we have found in data analysis, it is frequently the case that a topic dominates within a sampling unit (sentence), and that units from the same segment frequently are dominated by the same topic. In contrast to word-based representations, we expect no need to face sparsity issues that require smoothing methods (see TT) and ranking methods (see C99), which allows us to simplify the algorithm. Initially, the document is split into minimal units on sentence boundaries. To measure the coherence between units, the cosine similarity (vector dot product) between two adjacent sentences is computed. Each sentences s is represented as a N-dimensional vector, where N is the number of topics defined in the TMs. The i-th element of the vector contains the number of times the i-th topic is observed in the sentence. In comparison to TT we search all local minima based on these similarity scores and calculate for these positions the depth score as described in TT. If the number of segments is known in advance, the segments of the n-highest depth-scores are used, otherwise the cut-off score criteria used in TT is adapted.

5 Evaluation

As laid out in Section 3, a LDA Model is estimated on a training dataset and used for inference on the test set. To ensure that we do no use information from the test set, we perform a 10-fold Cross Validation (CV) for all reported results. To reduce the variance of the shown results, derived by the random nature of sampling and inference, the results for each fold are calculated 30 times using different LDA models.

The LDA model is trained with N=100 topics, 500 sampling iterations and symmetric hyperparameters as recommended by Griffiths and Steyvers (2004)(α=50/N and β=0.01), using JGibbsLda (Phan and Nguyen, 2007). For the annotation of unseen data with topic information, we use LDA inference, sampling 100 iterations. Inference is executed sentence-wise, since sentences form the minimal unit of our segmentation algorithms and we cannot use document information in the test setting. The performance of the algorithms is measured using P_k and *WindowDiff* (WD) metrics (Beeferman et al., 1999; Pevzner and Hearst, 2002). The C99 algorithm is initialized with a 11×11 ranking mask, as recommended in Choi (2000). TT is configured according to Choi (2000) with sequence length w=20 and block size k=6.

5.1 Data Set

For evaluation, we rely on the Choi data set (Choi, 2000), which has been used in several other text segmentation approaches to ensure comparability. This data set is generated artificially using the Brown corpus and consists of 700 documents. Each document consists of 10 segments. For its generation, 3–11 sentences are sequentially extracted from a randomly selected document and merged together. While our CV evaluation setting is designed to avoid using the same documents for training and testing, this cannot be guaranteed as the segments within the documents generated by Choi are included in several documents. This problem also occurs in other approaches, but has not be described in (Fragkou et al., 2004; Misra et al., 2009; Galley et al., 2003), where parts or the whole dataset are used for training either TF-IDF values or topic models.

5.2 Results

For the experiments the C99 and TT implementations[2] are executed in two settings: using words and using topics. When using words, TT and C99 use stemmed words and filter out words using a stopword list. C99 additional removes words using predefined regular expressions. In the case of topic IDs, no stopword filtering was deemed necessary. Table 1 shows the result of the different algorithms with all combination of provided segment number and using the mode method.

We note that WD values are always higher than the P_k values, and these measures are highly correlated. First we discuss results for the setting with number of segments provided (see column 2-5 of

[2] We use the implementations by Choi available at `http://code.google.com/p/uima-text-segmenter/`.

Method	Segments provided				Segments unprovided			
	mode=false		mode=true		mode=false		mode=true	
	Pk	WD	Pk	WD	Pk	WD	Pk	WD
C99	11.20	12.07			12.73	14.57		
C99LDA	4.16	4.89	2.67	3.08	8.69	10.52	3.24	4.08
TT	44.48	47.11			49.51	66.16		
TTLDA	1.85	2.10	**1.04**	**1.18**	16.41	21.40	2.89	3.67
TopicTiling	2.65	3.02	2.12	2.42	4.12	5.75	2.30	3.08
TopicTiling (filtered)	**1.50**	**1.72**	1.06	1.21	**3.24**	**4.58**	**1.39**	**1.84**

Table 1: Results by segment length for TT with words and topics (TTLDA), C99 with words and topics (C99LDA) and TopicTiling using all sentences and using only sentences with more then 5 word tokens (filtered).

Method	Segments	
	provided	**unprovided**
TT	44.48	49.51
C99	11.20	12.73
U00 (Utiyama and Isahara, 2001)	9	10
F04 (Fragkou et al., 2004)	5.39	
M09 (Misra et al., 2009)	2.73	
C99LDA (mode = true)	2.67	3.24
TTLDA (mode=true)	**1.04**	2.89
TopicTiling (mode=true, filtered)	1.06	**1.39**

Table 2: List of lowest P_k values for the Choi data set for different algorithms in the literature.

Table 1). A significant improvement for C99 and TT can be achieved when using topic IDs. In case of C99LDA, the error rate is at least halved and for TTLDA the error rate is reduced by a factor of 20. Using the most frequent topic ID assigned during the Bayesian inference (mode method) reduces the error rates further for the TM-based approaches, as the probability for randomly assigned topic IDs is decreased. The newly introduced algorithm TopicTiling as described above does not improve over TTLDA. Analysis revealed that the Choi corpus includes also captions and other "non-sentences" that are marked as sentences, which causes TopicTiling to introduce false positive segments since the topic vectors are too sparse for these short "non-sentences". We therefore filter out "sentences" with less than 5 words (see bottom line in Table 1). This leads to errors values that are close to the results achieved with TTLDA when the mode is used. When the number of segments is not given in advance (see columns 6-9 in Table 1), we again observe significantly better results comparing topic-based methods to word-based methods. But the error rates of TTLDA are unexpectedly high when the mode method is not used. We discovered in data analysis that TT estimates too many segments, as the topic ID distributions between adjacent sentences within a segment are often too diverse, especially in face of random fluctuations from the topic assignments. Estimating the number of segments is better achieved using TopicTiling instead of TTLDA.

In Table 2, we compare TTLDA, C99LDA and our TopicTiling algorithm to other published results on the same dataset. We can see that all introduced topic-based methods outperform the yet best published M09 algorithm (Misra et al., 2009). The improvements of C99, TTLDA and TopicTiling in comparison to M09 are significant[3].

TopicTiling and TTLDA are computationally more efficient than M09. Whereas our linear method has a complexity of $O(T)$ (T is the number of sentences), dynamic algorithms like M09 have a complexity of $O(T^2)$ (cf. Fragkou et al. (2004)), which also applies to the number of topic inference runs. When the number of segments is not given in advance, TopicTiling outperforms TTLDA significantly. As an additional benefit, TopicTiling is even simpler than TT, as no smoothing parameter is needed and the depth scores are only calculated for the minima of the similarity scores.

6 Conclusion

The method introduced in this paper shows that using semantic information, provided by TMs, can improve existing algorithm significantly. This is attested modifying the algorithm TT and C99. With TopicTiling a new simplistic topic based algorithm is developed that can produce state-of-the-art results based on the Choi corpus and outperform TTLDA when the number of segments is unknown. Additionally this method is computationally more efficient in comparison to other topic based segmentation algorithms. Another contribution is the mode method for stabilizing topic ID assignments.

7 Acknowledgments

This work has been supported by LOEWE as part of the research center "Digital Humanities". We would like to thank the anonymous reviewers for their comments, which truly helped to improve the paper.

[3] using a one sampled t-test with $\alpha = 0.05$

References

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

David M. Blei, Andrew Y Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022.

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

Jacob Eisenstein. 2009. Hierarchical text segmentation from multi-scale lexical cohesion. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 353–361, Boulder, CO, USA.

Pavlina Fragkou, Vassilios Petridis, and Athanasios Kehagias. 2004. A Dynamic Programming Algorithm for Linear Text Segmentation. *Journal of Intelligent Information Systems*, 23(2):179–197.

Michel Galley, Kathleen McKeown, Eric Fosler-Lussier, and Hongyan Jing. 2003. Discourse segmentation of multi-party conversation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, volume 1, pages 562–569, Sapporo, Japan.

Thomas L. Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101:5228–5235.

M A K Halliday and Ruqaiya Hasan. 1976. *Cohesion in English*, volume 1 of *English Language Series*. Longman.

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

P.-Y. Hsueh, J. D. Moore, and S. Renals. 2006. Automatic segmentation of multiparty dialogue. AMI-156.

Hemant Misra, Joemon M Jose, and Olivier Cappé. 2009. Text Segmentation via Topic Modeling : An Analytical Study. In *Proceeding of the 18th ACM Conference on Information and Knowledge Management*, pages 1553–1556, Hong Kong.

Lev Pevzner and Marti A. Hearst. 2002. A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistic*, 28(1):19–36.

Xuan-Hieu Phan and Cam-Tu Nguyen. 2007. GibbsLDA++: A C/C++ implementation of latent Dirichlet allocation (LDA). http://jgibblda.sourceforge.net/.

Qi Sun, Runxin Li, Dingsheng Luo, and Xihong Wu. 2008. Text segmentation with LDA-based Fisher kernel. *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies*, pages 269–272.

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*, pages 499–506, Toulouse, France.

Yaakov Yaari. 1997. Segmentation of expository texts by hierarchical agglomerative clustering. In *Proceedings of the Conference on Recent Advances in Natural Language Processing*, Tzigov Chark, Bulgaria.