Learning to Rank Semantic Coherence for Topic Segmentation

Liang Wang[1] **Sujian Li**[1,2] **Yajuan Lyu**[3] **Houfeng Wang**[1,2]
[1]Key Laboratory of Computational Linguistics, Peking University, MOE, China
[2]Collaborative Innovation Center for Language Ability, Xuzhou, Jiangsu, China
[3]Baidu Inc., Beijing, China
{intfloat,lisujian,wanghf}@pku.edu.cn lvyajuan@baidu.com

Abstract

Topic segmentation plays an important role for discourse parsing and information retrieval. Due to the absence of training data, previous work mainly adopts unsupervised methods to rank semantic coherence between paragraphs for topic segmentation. In this paper, we present an intuitive and simple idea to automatically create a "quasi" training dataset, which includes a large amount of text pairs from the same or different documents with different semantic coherence. With the training corpus, we design a symmetric CNN neural network to model text pairs and rank the semantic coherence within the learning to rank framework. Experiments show that our algorithm is able to achieve competitive performance over strong baselines on several real-world datasets.

1 Introduction

The goal of topic segmentation is to segment a document into several topically coherent parts, with different parts corresponding to different topics. Topic segmentation enables better understanding of document structure, and makes long document much easier to navigate. It also provides helpful information for tasks such as information retrieval, topic tracking etc (Purver, 2011).

Due to the lack of large scale annotated topic segmentation dataset, previous work mainly focus on unsupervised models to measure the coherence between two textual segments. The intuition behind unsupervised models is that two adjacent segments from the same topic are more coherent than those from different topics. Under this intuition, one direction of research attempts to measure coherence by computing text similarity. The typical methods include *TextTiling* (Hearst, 1997) and its variants, such as *C99* (Choi, 2000), *TopicTiling* (Riedl and Biemann, 2012b) etc. The other direction of research develops topic modeling techniques to explore topic representation of text and topic change between textual segments (Yamron et al., 1998; Eisenstein and Barzilay, 2008; Riedl and Biemann, 2012a; Du et al., 2013; Jameel and Lam, 2013). With carefully designed generative process and efficient inference algorithm, topic models are able to model coherence as latent variables and outperform lexical similarity based models.

Though unsupervised models make progress in modeling text coherence, they mostly suffer from one of the following two limitations. First, it is not precise to measure coherence with text similarity, since text similarity is just one aspect to influence coherence. Second, many assumptions and manually set parameters are usually involved in the complex modeling techniques, due to the absence of supervised information. To overcome aforementioned limitations, we prefer to directly model the text coherence by exploring possible supervised information. Then, we can learn a function $f(s1, s2)$ which takes two textual segments $s1$ and $s2$ as input, and directly measure their semantic coherence.

As we know, it is hard to directly compile and collect a large number of samples with coherence scores labeling. Here we propose an intuitive and simple strategy to automatically create a "quasi" training corpus for supervision. It is a common sense that the original documents written by human are generally more coherent than a patchwork of sentences or paragraphs randomly extracted from different documents. In such cases, two textual segments from the same document are more coherent than those from different documents, and two segments from the same paragraph

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 1340–1344
Copenhagen, Denmark, September 7–11, 2017.

are more coherent than those from different paragraphs. Then, we can get a large set of text pairs with partial ordering relations, which denote some text pairs are more coherent than other text pairs. With these ordering information, we propose to apply the learning to rank framework to model the semantic coherence function $f(s1, s2)$, based on which topic boundaries are identified.

The next key problem is how to model and represent text pairs. It is fortunate that neural networks have emerged as a powerful tool for modeling text pairs (Lu and Li, 2013; Severyn and Moschitti, 2015; Yin et al., 2015; Hu et al., 2014), freeing us from feature engineering. In this paper, we develop a symmetric convolutional neural network (CNN) framework, whose main idea is to jointly model text representation and interaction between texts. With our acquired large amount of training data, our CNN-based method is capable of reasonably rank semantic coherence and further conduct topic segmentation.

2 Model

2.1 Coherence Ordering between Text Pairs

In our work, we define $f(s1, s2)$ as a function, which returns a real number as semantic coherence score of the text pair <s1,s2>. To model $f(s1, s2)$ of any text pair, we aim to explore the partial ordering relations of coherence between different text pairs, since it is hard to get a corpus with labeled coherence scores.

Next, we exploit the two types of ordering relations stated in Section 1. To formalize, we notate a collection of documents as D. Each document $d_i \in D$ consists of several paragraphs, and each paragraph $p_j \in d_i$ consists of several sentences. We use $T_{ij}^{s:(s+k)}$ to represent a text segment covering k sentences starting from the s-th sentence in document d_i's j-th paragraph. To make symbols less cluttered, we omit k and simply use T_{ij}^{s} for the same meaning. A text pair $< T_{ij}^{s}, T_{i'j'}^{s'} >$ is a tuple of two text segments.

The first one ordering relation is: coherence score of a text pair from different documents is lower than that from the same document. Formally, its mathematical expression is shown below:

$$f(< T_{i\cdot}^{\cdot}, T_{i'\cdot}^{\cdot} >) < f(< T_{jm}^{\cdot}, T_{jm'}^{\cdot} >), \quad i \neq i', m \neq m' \tag{1}$$

where dot $\cdot$ means arbitrary value.

The second one is: coherence score of text pair from different paragraphs is lower than those from the same paragraph, as represented below.

$$f(< T_{ip}^{\cdot}, T_{ip'}^{\cdot} >) < f(< T_{jq}^{n}, T_{jq}^{n+k} >), p \neq p' \tag{2}$$

As our defined relations are partially ordering, they have the properties of reflexivity, transitivity, and antisymmetry, Then we can easily infer that coherence score of a text pair from different documents is also lower than that from the same paragraph.

2.2 Learning to Rank Semantic Coherence

Learning to rank is a widely used learning framework in the field of information retrieval (Liu et al., 2009). There are generally three formulations (Li, 2011): *pointwise ranking*, *pairwise ranking*, and *listwise ranking*. The goal is to learn a ranking function $f(\mathbf{w}, tp_i) \rightarrow y_i$ where tp_i denotes a text pair <s1,s2>. f maps tp_i to a real value y_i which is semantic coherence score in this paper, $\mathbf{w}$ is weight vector. We examine both *pointwise ranking* and *pairwise ranking* methods, *listwise ranking* is not naturally fit for our task, so it is not discussed here.

2.2.1 Pointwise Ranking

For pointwise formulation, $y_i = f(\mathbf{w}, tp_i) \in [0, 1]$ computes inner product between weight vector $\mathbf{w}$ and text pair tp_i's representation vector $\mathbf{h_i}$. Here we apply a $sigmoid$ non-linearity function.

$$y_i = \sigma(\mathbf{w} \cdot \mathbf{h_i}) \tag{3}$$

Representation vectors $\mathbf{h_i}$ of the text pair can be jointly learned through a neural network, which will be introduced in next subsection.

To conform to the partial ordering relations, we score each training instance tp_i as follows.

$$y_i^* = \begin{cases} 0, & \text{If } tp_i \text{ comes from different documents.} \\ 1, & \text{If } tp_i \text{ comes from same paragraph.} \\ \alpha, & \text{If } tp_i \text{ comes from different paragraphs.} \end{cases}$$

where $0 < \alpha < 1$ and α is a hyper-parameter chosen to maximize performance on validation dataset.

With N training instances, we formulate the coherence scoring as a regression problem and use cross entropy as loss function:

$$\min -\frac{1}{N}\sum_{i=1}^{N}(y_i \log y_i^* + (1-y_i)\log(1-y_i^*)) \tag{4}$$

Generally speaking, *pointwise ranking* is simple, scalable and efficient to train.

2.2.2 Pairwise Ranking with Sampling

Pairwise formulation explicitly compares each pair of training instance and requires a minimal margin ϵ between their ranking score.

$$f(\mathbf{w}, tp_i) > f(\mathbf{w}, tp_j) + \epsilon \quad (5)$$

Here, the text pair tp_i has a higher ranking score than tp_j, and $y_i = f(\mathbf{w}, tp_i) \in (-\infty, +\infty)$. Without loss of generality, we set $\epsilon = 1$ and use squared hinge loss as optimization function.

$$\min \ -\frac{1}{M} \sum_{i,j} max(0, 1 + y_j - y_i)^2 \quad (6)$$

where M is the number of pairs we need to compare. As we can see, in our problem setting, $M \approx N^2$, which makes M an extremely large number when $N \approx 10^5$.

To make training feasible, we adopt a straightforward sampling mechanism, which randomly samples pairs from different groups to construct a mini-batch on the fly during training.

Pairwise ranking is reported to have better performance than *pointwise ranking*, but it is less efficient to train.

2.3 Semantic Coherence Neural Network

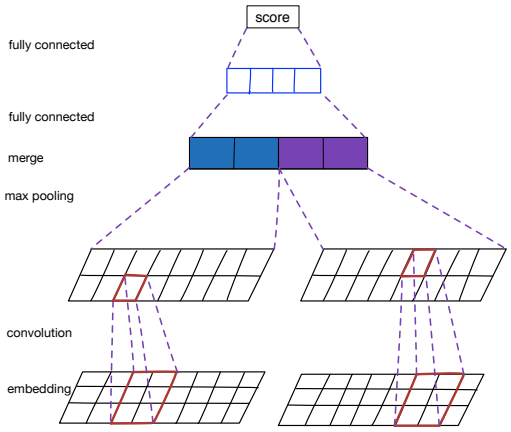

Figure 1: Semantic Coherence Neural Network

To model the text pair instances, we develop a symmetric convolutional neural network (CNN) architecture, as shown in Figure 1. Our model consists of two symmetric CNN models, and the two CNNs share their network configuration and parameters. Each CNN converts one text into a low-dimensional representation, and two generated text representation vectors are finally concatenated and fed into the scoring layer to get a real value as the coherence score.

2.4 Inference

At test time, coherence scores between any two adjacent paragraphs are computed. $T - 1$ paragraph boundaries with lowest semantic coherence score are chosen as topic boundaries, where T is ground-truth number of topics.

This inference procedure is computationally efficient. Unlike *TextTiling*, it doesn't need to calculate a so-called "depth score".

3 Experiments

3.1 Experimental Setup

Data In order to train our ranking neural network, we use full *English Wikipedia dump*, which consists of more than 5 million documents, to automatically construct text pairs.

For performance evaluation, we use topic segmentation dataset from (Jeong and Titov, 2010)[1]. This dataset consists of 864 manually labeled documents from four different areas, as shown in Table 1.

	News	*Lecture*	*Report*	*Biography*
#documents	184	120	160	400

Table 1: Overview of four datasets.

Baselines To compare with our method, *TextTiling* (Hearst, 1997), *TopicTiling* (Riedl and Biemann, 2012b) and *BayesSeg* (Eisenstein and Barzilay, 2008) are adopted as three baselines. We use open source implementations of *TextTiling*[2] and *TopicTiling*[3], and results of *BayesSeg* are from (Jeong and Titov, 2010).

Hyperparameters Our neural network implementation is based on *Tensorflow* (Abadi et al., 2015). We use pre-trained 50 dimensional *Glove* vectors (Pennington et al., 2014)[4] for word embeddings initialization. Each text pair consists of 2 text segments, and each text segment consists of

[1] We do not compare with *MultiSeg* model proposed by (Jeong and Titov, 2010), since our model is for single-document topic segmentation while *MultiSeg* is for multi-document topic segmentation.

[2] https://github.com/nltk/nltk/tree/develop/nltk/tokenize

[3] https://github.com/ldulcic/text-segmentation

[4] http://nlp.stanford.edu/projects/glove/

	News			*Lecture*			*Report*			*Biography*		
	P_k	WD	$F1$	P_k	WD	$F1$	P_k	WD	$F1$	P_k	WD	$F1$
TextTiling	0.340	0.344	0.447	0.204	0.206	0.231	0.466	0.469	0.365	0.335	0.403	0.361
TopicTiling	0.415	0.436	0.338	0.359	0.379	0.571	0.288	0.296	0.383	0.381	0.423	0.390
BayesSeg	0.318	0.326	0.537	**0.173**	0.190	0.526	0.254	0.255	0.526	**0.186**	**0.208**	0.470
Ours-pair-finetune	0.180	0.181	0.570	0.200	0.202	0.560	0.263	0.263	0.492	0.223	0.228	0.448
Ours-point-finetune	0.182	0.183	0.572	0.197	0.200	0.569	0.245	0.247	0.511	0.229	0.232	0.442
Ours-pair-static	0.176	0.178	0.580	0.177	0.180	0.600	0.252	0.253	0.518	0.220	0.224	0.472
Ours-point-static	**0.173**	**0.175**	**0.587**	0.176	**0.179**	**0.608**	**0.240**	**0.241**	**0.529**	0.216	0.219	**0.479**

Table 2: Experimental results. (a) *Ours-pair-finetune* is pairwise ranking model with word embedding fine-tuning. (b) *Ours-point-static* is pointwise ranking model without word embedding fine-tuning, etc.

no more than 3 sentences. Stop words and digits are removed from input text, and all words are converted to lowercase. We pad input sequence to 40 tokens. In order to capture information of different granularity, convolution window size of both 2 and 3 are used, with 64 filters for each window size. L2 regularization coefficient is set to 0.001. *Adam* algorithm (Kingma and Ba, 2014) is used for loss function minimization. We set α to 0.7 for *pointwise ranking*.

Evaluation System performance is evaluated according to three metrics: P_k (Beeferman et al., 1999), WindowDiff(WD) (Pevzner and Hearst, 2002) and $F1$ score. P_k and WD are calculated based on sliding windows, and can assign partial score to incorrect segmentation. Note that P_k and WD are penalty metrics, smaller value means better performance.

3.2 Results and Analysis

Experimental results are shown in Table 2. Our proposed model is examined in 4 different settings, including whether to use *pointwise ranking* or *pairwise ranking* algorithm, and whether to fine-tune word embeddings or not. The best model *Ours-pointwise-static* is able to achieve better or competitive performance compared to *BayesSeg* and *TopicTiling* according to all three metrics, especially on *News* dataset. *TopicTiling* is reported to perform well on heuristically constructed dataset (Riedl and Biemann, 2012b), but behave mediocre on manually labeled dataset in our experiments.

One interesting phenomenon is that fine-tuned word embeddings has negative impact on overall performance, which is generally not the case in many NLP tasks. The reason may be that our task involves domain adaptation, and word embeddings should generalize well across different domains rather than adapt to *Wikipedia* text. Though our proposed sampling mechanism enables easier training of *pairwise ranking* model, it inevitably loses some ordering information, which makes *pairwise ranking* model perform slightly worse than *pointwise ranking* model.

Text Pair	Score
A: A variety of techniques have been directed toward the study of blood group antibodies. **B:** If I'd work on my place-kicking he thought he could use me.	0.022
A: A second miracle is required for her to proceed to canonization. **B:** Mother Teresa inspired a variety of commemorations.	0.587
A: Plants have an amazing ability to respond to stimuli from their environment. **B:** These responses to environmental factors are known as tropisms.	0.861

Table 3: Coherence Score between Text Pairs.

To illustrate what the model has learned, we show some typical examples of coherence score for text pair **<A,B>** in Table 3. There is almost no lexical overlap for all the three text pairs, cosine similarity between one-hot vectors would surely fail to rank them, even though *"canonization"* and *"commemorations"*, *"respond"* and *"responses"*, *"environment"* and *"environmental"* are closely related semantically. As we expect, our proposed model is able to capture such semantic relatedness and assign reasonable score to each text pair, which is a key to topic boundary detection.

4 Conclusion

This paper proposes a novel approach for topic segmentation by learning to rank semantic coherence. Symmetric convolutional neural network is used for text pair modeling. Training data can be automatically constructed from unlabeled documents, and no labeled data is needed. Experiments show promising performance on dataset from various domains.

Acknowledgments

We thank the anonymous reviewers for their insightful comments on this paper. This work was partially supported by National Natural Science Foundation of China (61572049 and 61333018) and Baidu-Peking University Joint Project. The correspondence author of this paper is Sujian Li.

References

Martın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. 2015. Tensorflow: Large-scale machine learning on heterogeneous systems, 2015. *Software available from tensorflow. org* 1.

Doug Beeferman, Adam Berger, and John Lafferty. 1999. Statistical models for text segmentation. *Machine learning* 34(1-3):177–210.

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.

Lan Du, Wray L Buntine, and Mark Johnson. 2013. Topic segmentation with a structured topic model. In *HLT-NAACL*. pages 190–200.

Jacob Eisenstein and Regina Barzilay. 2008. Bayesian unsupervised topic segmentation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 334–343.

Marti A Hearst. 1997. Texttiling: Segmenting text into multi-paragraph subtopic passages. *Computational linguistics* 23(1):33–64.

Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Advances in neural information processing systems*. pages 2042–2050.

Shoaib Jameel and Wai Lam. 2013. An unsupervised topic segmentation model incorporating word order. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*. ACM, pages 203–212.

Minwoo Jeong and Ivan Titov. 2010. Multi-document topic segmentation. In *Proceedings of the 19th ACM international conference on Information and knowledge management*. ACM, pages 1119–1128.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Hang Li. 2011. A short introduction to learning to rank. *IEICE TRANSACTIONS on Information and Systems* 94(10):1854–1862.

Tie-Yan Liu et al. 2009. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval* 3(3):225–331.

Zhengdong Lu and Hang Li. 2013. A deep architecture for matching short texts. In *Advances in Neural Information Processing Systems*. pages 1367–1375.

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.

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

Matthew Purver. 2011. Topic segmentation. *Spoken language understanding: systems for extracting semantic information from speech* pages 291–317.

Martin Riedl and Chris Biemann. 2012a. Text segmentation with topic models. *Journal for Language Technology and Computational Linguistics* 27(1):47–69.

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

Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, pages 373–382.

Jonathan P Yamron, Ira Carp, Larry Gillick, Steve Lowe, and Paul van Mulbregt. 1998. A hidden markov model approach to text segmentation and event tracking. In *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*. IEEE, volume 1, pages 333–336.

Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2015. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *arXiv preprint arXiv:1512.05193* .