Learning to Order Natural Language Texts

Jiwei Tan[a, b], Xiaojun Wan[a]* and Jianguo Xiao[a]
[a]Institute of Computer Science and Technology, The MOE Key Laboratory of Computational Linguistics, Peking University, China
[b]School of Information Science and Technology, Beijing Normal University, China
tanjiwei8@gmail.com, {wanxiaojun,jgxiao}@pku.edu.cn

Abstract

Ordering texts is an important task for many NLP applications. Most previous works on summary sentence ordering rely on the contextual information (e.g. adjacent sentences) of each sentence in the source document. In this paper, we investigate a more challenging task of ordering a set of unordered sentences without any contextual information. We introduce a set of features to characterize the order and coherence of natural language texts, and use the learning to rank technique to determine the order of any two sentences. We also propose to use the genetic algorithm to determine the total order of all sentences. Evaluation results on a news corpus show the effectiveness of our proposed method.

1 Introduction

Ordering texts is an important task in many natural language processing (NLP) applications. It is typically applicable in the text generation field, both for concept-to-text generation and text-to-text generation (Lapata, 2003), such as multiple document summarization (MDS), question answering and so on. However, ordering a set of sentences into a coherent text is still a hard and challenging problem for computers.

Previous works on sentence ordering mainly focus on the MDS task (Barzilay et al., 2002; Okazaki et al., 2004; Nie et al., 2006; Ji and Pulman, 2006; Madnani et al., 2007; Zhang et al., 2010; He et al., 2006; Bollegala et al., 2005; Bollegala et al., 2010). In this task, each summary sentence is extracted from a source document. The timestamp of the source documents and the adjacent sentences in the source documents can be used as important clues for ordering summary sentences.

In this study, we investigate a more challenging and more general task of ordering a set of unordered sentences (e.g. randomly shuffle the sentences in a text paragraph) without any contextual information. This task can be applied to almost all text generation applications without restriction.

In order to address this challenging task, we first introduce a few useful features to characterize the order and coherence of natural language texts, and then propose to use the learning to rank algorithm to determine the order of two sentences. Moreover, we propose to use the genetic algorithm to decide the overall text order. Evaluations are conducted on a news corpus, and the results show the prominence of our method. Each component technique or feature in our method has also been validated.

2 Related Work

For works taking no use of source document, Lapata (2003) proposed a probabilistic model which learns constraints on sentence ordering from a corpus of texts. Experimental evaluation indicated the importance of several learned lexical and syntactic features. However, the model only works well when using single feature, but unfortunately, it becomes worse when multiple features are combined. Barzilay and Lee (2004) investigated the utility of domain-specific content model for representing topic and topic shifts and the model performed well on the five selected domains. Nahnsen (2009) employed features which were based on discourse entities, shallow syntactic analysis, and temporal precedence relations retrieved from VerbOcean. However, the model does not perform well on datasets describing the consequences of events.

3 Our Proposed Method

3.1 Overview

The task of text ordering can be modeled like (Cohen et al., 1998), as measuring the coherence of a text by summing the association strength of any sentence pairs. Then the objective of a text ordering model is to find a permutation which can maximize the summation.

* Xiaojun Wan is the corresponding author.

Formally, we define an association strength function $\mathrm{PREF}(u,v) \in \mathrm{R}$ to measure how strong it is that sentence u should be arranged before sentence v (denoted as $u \succ v$). We then define function AGREE(ρ,PREF) as:

$$\mathrm{AGREE}(\rho,\mathrm{PREF}) = \sum_{u,v:\rho(u)>\rho(v)} \mathrm{PREF}(u,v) \quad (1)$$

where ρ denotes a sentence permutation and $\rho(u) > \rho(v)$ means $u \succ v$ in the permutation ρ. Then the objective of finding an overall order of the sentences becomes finding a permutation ρ to maximize $\mathrm{AGREE}(\rho,\mathrm{PREF})$.

The main framework is made up of two parts: defining a pairwise order relation and determining an overall order. Our study focuses on both the two parts by learning a better pairwise relation and proposing a better search strategy, as described respectively in next sections.

3.2 Pairwise Relation Learning

The goal for pairwise relation learning is defining the strength function PREF for any sentence pair. In our method we define the function PREF by combining multiple features.

Method: Traditionally, there are two main methods for defining a strength function: integrating features by a linear combination (He et al., 2006; Bollegala et al., 2005) or by a binary classifier (Bollegala et al., 2010). However, the binary classification method is very coarse-grained since it considers any pair of sentences either "positive" or "negative". Instead we propose to use a better model of learning to rank to integrate multiple features.

In this study, we use Ranking SVM implemented in the svm^{rank} toolkit (Joachims, 2002; Joachims, 2006) as the ranking model. The examples to be ranked in our ranking model are sequential sentence pairs like $u \succ v$. The feature values for a training example are generated by a few feature functions $f_i(u,v)$, and we will introduce the features later. We build the training examples for svm^{rank} as follows:

For a training query, which is a paragraph with n sequential sentences as $s_1 \succ s_2 \succ \ldots \succ s_n$, we can get $A_n^2 = n(n-1)$ training examples. For pairs like $s_a \succ s_{a+k} (k>0)$ the target rank values are set to $n-k$, which means that the longer the distance between the two sentences is, the smaller the target value is. Other pairs like $s_{a+k} \succ s_a$ are all set to 0. In order to better capture the order information of each feature, for every sentence pair $u \succ v$, we derive four feature values from each function $f_i(u,v)$, which are listed as follows:

$$V_{i,1} = f_i(u,v) \quad (2)$$

$$V_{i,2} = \begin{cases} 1/2, & \text{if } f_i(u,v) + f_i(v,u) = 0 \\ \dfrac{f_i(u,v)}{f_i(u,v) + f_i(v,u)}, & \text{otherwise} \end{cases} \quad (3)$$

$$V_{i,3} = \begin{cases} 1/|S|, & \text{if } \sum_{y \in S \cap y \neq u} f_i(u,y) = 0 \\ f_i(u,v) / \sum_{y \in S \cap y \neq u} f_i(u,y), & \text{otherwise} \end{cases} \quad (4)$$

$$V_{i,4} = \begin{cases} 1/|S|, & \text{if } \sum_{x \in S \cap x \neq v} f_i(x,v) = 0 \\ f_i(u,v) / \sum_{x \in S \cap x \neq v} f_i(x,v), & \text{otherwise} \end{cases} \quad (5)$$

where S is the set of all sentences in a paragraph and $|S|$ is the number of sentences in S. The three additional feature values of (3) (4) (5) are defined to measure the priority of $u \succ v$ to $v \succ u$, $u \succ v$ to $u \succ \forall y \in S - \{u,v\}$ and $u \succ v$ to $\forall x \in S - \{u,v\} \succ v$ respectively, by calculating the proportion of $f_i(u,v)$ in respective summations.

The learned model can be used to predict target values for new examples. A paragraph of unordered sentences is viewed as a test query, and the predicted target value for $u \succ v$ is set as $\mathrm{PREF}(u,v)$.

Features: We select four types of features to characterize text coherence. Every type of features is quantified with several functions distinguished by i in the formulation of $f_i(u,v)$ and normalized to $[0,1]$. The features and definitions of $f_i(u,v)$ are introduced in Table 1.

Type	Description
Similarity	$\mathrm{sim}(u,v)$
	$\mathrm{sim}(\mathrm{latter}(u),\mathrm{former}(v))$
Overlap	$\mathrm{overlap}_j(u,v) / \min(\lvert u \rvert, \lvert v \rvert)$
	$\dfrac{\mathrm{overlap}_j(\mathrm{latter}(u),\mathrm{former}(v))}{\mathrm{overlap}_j(u,v)}$
Coreference	Number of coreference chains
	Number of coreference words
Probability Model	Noun
	Verb
	Verb & noun dependency
	Adjective & adverb

Table 1: Features used in our model.

As in Table 1, function $\text{sim}(u,v)$ denotes the cosine similarity of sentence u and v; $\text{latter}(u)$ and $\text{former}(v)$ denotes the latter half part of u and the former part of v respectively, which are separated by the most centered comma (if exists) or word (if no comma exits); $\text{overlap}_j(u,v)$ denotes the number of mutual words of u and v, for $j = 1,2,3$ representing lemmatized noun, verb and adjective or adverb respectively; $|u|$ is the number of words of sentence u. The value will be set to 0 if the denominator is 0.

For the coreference features we use the ARK-ref[1] tool. It can output the coreference chains containing words which represent the same entity for two sequential sentences $u \succ v$.

The probability model originates from (Lapata, 2003), and we implement the model with four features of lemmatized noun, verb, adjective or adverb, and verb and noun related dependency.

3.3 Overall Order Determination

Cohen et al. (1998) proved finding a permutation ρ to maximize AGREE(ρ,PREF) is NP-complete. To solve this, they proposed a greedy algorithm for finding an approximately optimal order. Most later works adopted the greedy search strategy to determine the overall order.

However, a greedy algorithm does not always lead to satisfactory results, as our experiment shows in Section 4.2. Therefore, we propose to use the genetic algorithm (Holland, 1992) as the search strategy, which can lead to better results.

Genetic Algorithm: The genetic algorithm (GA) is an artificial intelligence algorithm for optimization and search problems. The key point of using GA is modeling the individual, fitness function and three operators of crossover, mutation and selection. Once a problem is modeled, the algorithm can be constructed conventionally.

In our method we set a permutation ρ as an individual encoded by a numerical path, for example a permutation $s_2 \succ s_1 \succ s_3$ is encoded as (2 1 3). Then the function AGREE(ρ,PREF) is just the fitness function. We adopt the order-based crossover operator which is described in (Davis, 1985). The mutation operator is a random inversion of two sentences. For selection operator we take a tournament selection operator which randomly selects two individuals to choose the one with the greater fitness value AGREE(ρ,PREF). After several generations of evolution, the individual with the greatest fitness value will be a close solution to the optimal result.

4 Experiments

4.1 Experiment Setup

Data Set and Evaluation Metric: We conducted the experiments on the North American News Text Corpus[2]. We trained the model on 80 thousand paragraphs and tested with 200 shuffled paragraphs. We use Kendall's τ as the evaluation metric, which is based on the number of inversions in the rankings.

Comparisons: It is incomparable with other methods for summary sentence ordering based on special summarization corpus, so we implemented Lapata's probability model for comparison, which is considered the state of the art for this task. In addition, we implemented a random ordering as a baseline. We also tried to use a classification model in place of the ranking model. In the classification model, sentence pairs like $s_a \succ s_{a+1}$ were viewed as positive examples and all other pairs were viewed as negative examples. When deciding the overall order for either ranking or classification model we used three search strategies: greedy, genetic and exhaustive (or brutal) algorithms. In addition, we conducted a series of experiments to evaluate the effect of each feature. For each feature, we tested in two experiments, one of which only contained the single feature and the other one contained all the other features. For comparative analysis of features, we tested with an exhaustive search algorithm to determine the overall order.

4.2 Experiment Results

The comparison results in Table 2 show that our Ranking SVM based method improves the performance over the baselines and the classification based method with any of the search algorithms. We can also see the greedy search strategy does not perform well and the genetic algorithm can provide a good approximate solution to obtain optimal results.

Method	Greedy	Exhaustive	Genetic
Baseline	-0.0127		
Probability	0.1859		
Classification	0.5006	0.5360	0.5264
Ranking	0.5191	0.5768	0.5747

Table 2: Average τ of different methods.

[1] http://www.ark.cs.cmu.edu/ARKref/

[2] The corpus is available from http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC98T30

Ranking vs. Classification: It is not surprising that the ranking model is better, because when using a classification model, an example should be labeled either positive or negative. It is not very reasonable to label a sentence pair like $s_a \succ s_{a+k} (k > 1)$ as a negative example, nor a positive one, because in some cases, it is easy to conclude one sentence should be arranged after another but hard to decide whether they should be adjacent. As we see in the function AGREE, the value of $\text{PREF}(s_a, s_{a+k})$ also contributes to the summation. In a ranking model, this information can be quantified by the different priorities of sentence pairs with different distances.

Single Feature Effect: The effects of different types of features are shown in Table 3. *Prob* denotes Lapata's probability model with different features.

Feature	Only	Removed
Similarity	0.0721	0.4614
Overlap	0.1284	0.4631
Coreference	0.0734	0.4704
$Prob_{noun}$	0.3679	0.3932
$Prob_{verb}$	0.0615	0.4544
$Prob_{adjective\&adverb}$	0.2650	0.4258
$Prob_{dependency}$	0.2687	0.4892
All	0.5768	

Table 3: Effects of different features.

It can be seen in Table 3 that all these features contribute to the final result. The two features of noun probability and dependency probability play an important role as demonstrated in (Lapata, 2003). Other features also improve the final performance. A paragraph which is ordered entirely right by our method is shown in Figure 1.

(1) Vanunu, 43, is serving an 18-year sentence for treason.
(2) He was kidnapped by Israel's Mossad spy agency in Rome in 1986 after giving The Sunday Times of London photographs of the inside of the Dimona reactor.
(3) From the photographs, experts determined that Israel had the world's sixth largest stockpile of nuclear weapons.
(4) Israel has never confirmed or denied that it has a nuclear capability.

Figure 1: A right ordered paragraph.

Sentences which should be arranged together tend to have a higher similarity and overlap. Like sentence (3) and (4) in Figure 1, they have a highest cosine similarity of 0.2240 and most overlap words of "Israel" and "nuclear". However, the similarity or overlap of the two sentences does not help to decide which sentence should be arranged before another. In this case the overlap and similarity of half part of the sentences may help. For example latter((3)) and former((4)) share an overlap of "Israel" while there is no overlap for latter((4)) and former((3)).

Coreference is also an important clue for ordering natural language texts. When we use a pronoun to represent an entity, it always has occurred before. For example when conducting coreference resolution for $(1) \succ (2)$, it will be found that "He" refers to "Vanunu". Otherwise for $(2) \succ (1)$, no coreference chain will be found.

4.3 Genetic Algorithm

There are three main parameters for GA including the crossover probability (PC), the mutation probability (PM) and the population size (PS). There is no definite selection for these parameters. In our study we experimented with a wide range of parameter values to see the effect of each parameter. It is hard to traverse all possible combinations so when testing a parameter we fixed the other two parameters. The results are shown in Table 4.

Value / Para	Avg	Max	Min	Stddev
PS	0.5731	0.5859	0.5606	0.0046
PC	0.5733	0.5806	0.5605	0.0038
PM	0.5741	0.5803	0.5337	0.0045

Table 4: Results of GA with different parameters.

As we can see in Table 4, when adjusting the three parameters the average τ values are all close to the exhaustive result of 0.5768 and their standard deviations are low. Table 4 shows that in our case the genetic algorithm is not very sensible to the parameters. In the experiments, we set PS to 30, PC to 0.5 and PM to 0.05, and reached a value of 0.5747, which is very close to the theoretical upper bound of 0.5768.

5 Conclusion and Discussion

In this paper we propose a method for ordering sentences which have no contextual information by making use of Ranking SVM and the genetic algorithm. Evaluation results demonstrate the good effectiveness of our method.

In future work, we will explore more features such as semantic features to further improve the performance.

Acknowledgments

The work was supported by NSFC (61170166), Beijing Nova Program (2008B03) and National High-Tech R&D Program (2012AA011101).

References

Danushka Bollegala, Naoaki Okazaki, Mitsuru Ishizuka. 2005. A machine learning approach to sentence ordering for multi-document summarization and its evaluation. In *Proceedings of the Second international joint conference on Natural Language Processing (IJCNLP '05)*, 624-635.

Danushka Bollegala, Naoaki Okazaki, and Mitsuru Ishizuka. 2010. A bottom-up approach to sentence ordering for multi-document summarization. *Inf. Process. Manage.* 46, 1 (January 2010), 89-109.

John H. Holland. 1992. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA.

Lawrence Davis. 1985. Applying adaptive algorithms to epistatic domains. In *Proceedings of the 9th international joint conference on Artificial intelligence - Volume 1* (IJCAI'85), Aravind Joshi (Ed.), Vol. 1. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 162-164.

Mirella Lapata. 2003. Probabilistic text structuring: experiments with sentence ordering. In*Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*(ACL '03), Vol. 1. Association for Computational Linguistics, Stroudsburg, PA, USA, 545-552.

Naoaki Okazaki, Yutaka Matsuo, and Mitsuru Ishizuka. 2004. Improving chronological sentence ordering by precedence relation. In *Proceedings of the 20th international conference on Computational Linguistics* (COLING '04). Association for Computational Linguistics, Stroudsburg, PA, USA, , Article 750 .

Nitin Madnani, Rebecca Passonneau, Necip Fazil Ayan, John M. Conroy, Bonnie J. Dorr, Judith L. Klavans, Dianne P. O'Leary, and Judith D. Schlesinger. 2007. Measuring variability in sentence ordering for news summarization. In *Proceedings of the Eleventh European Workshop on Natural Language Generation* (ENLG '07), Stephan Busemann (Ed.). Association for Computational Linguistics, Stroudsburg, PA, USA, 81-88.

Paul D. Ji and Stephen Pulman. 2006. Sentence ordering with manifold-based classification in multi-document summarization. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing* (EMNLP '06). Association for Computational Linguistics, Stroudsburg, PA, USA, 526-533.

Regina Barzilay, Noemie Elhadad, and Kathleen McKeown. 2002. Inferring strategies for sentence ordering in multidocument news summarization. *Journal of Artificial Intelligence Research*, 17:35–55.

Regina Barzilay and Lillian Lee. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *HLT-NAACL2004: Proceedings of the Main Conference*, pages 113–120.

Renxian Zhang, Wenjie Li, and Qin Lu. 2010. Sentence ordering with event-enriched semantics and two-layered clustering for multi-document news summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters* (COLING '10). Association for Computational Linguistics, Stroudsburg, PA, USA, 1489-1497.

Thade Nahnsen. 2009. Domain-independent shallow sentence ordering. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Student Research Workshop and Doctoral Consortium* (SRWS '09). Association for Computational Linguistics, Stroudsburg, PA, USA, 78-83.

Thorsten Joachims. 2002. Optimizing search engines using click through data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining* (KDD '02). ACM, New York, NY, USA, 133-142.

Thorsten Joachims. 2006. Training linear SVMs in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining* (KDD '06). ACM, New York, NY, USA, 217-226.

William W. Cohen, Robert E. Schapire, and Yoram Singer. 1998. Learning to order things. In*Proceedings of the 1997 conference on Advances in neural information processing systems 10*(NIPS '97), Michael I. Jordan, Michael J. Kearns, and Sara A. Solla (Eds.). MIT Press, Cambridge, MA, USA, 451-457.

Yanxiang He, Dexi Liu, Hua Yang, Donghong Ji, Chong Teng, and Wenqing Qi. 2006. A hybrid sentence ordering strategy in multi-document summarization. In *Proceedings of the 7th international conference on Web Information Systems* (WISE'06), Karl Aberer, Zhiyong Peng, Elke A. Rundensteiner, Yanchun Zhang, and Xuhui Li (Eds.). Springer-Verlag, Berlin, Heidelberg, 339-349.

Yu Nie, Donghong Ji, and Lingpeng Yang. 2006. An adjacency model for sentence ordering in multi-document summarization. In Proceedings of the Third Asia conference on Information Retrieval Technology (AIRS'06), 313-322.