Update Legal Documents Using Hierarchical Ranking Models and Word Clustering

Minh Quang Nhat PHAM [a], Minh Le NGUYEN [a] and Akira SHIMAZU [a]

[a] *School of Information Science, Japan Advanced Institute of Science and Technology*

Abstract. Our research addresses the task of updating legal documents when new information emerges. In this paper, we employ a hierarchical ranking model to the task of updating legal documents. Word clustering features are incorporated to the ranking models to exploit semantic relations between words. Experimental results on legal data built from the United States Code show that the hierarchical ranking model with word clustering outperforms baseline methods using Vector Space Model, and word cluster-based features are effective features for the task.

Keywords. Updating Task, Hierarchical Ranking, Word Clustering

Introduction

Updating existing documents when new information emerges is a time-consuming task, especially for document types which need to be changed regularly. In Legal domain, the updating task is challenging because of the large number of legal documents and legal updates. Moreover, relations among documents or among parts within one document make it even harder. One revision in a document may lead to change requirements in other documents or other parts of the same document.

In our understanding, there are very few works on the task of updating documents. Chen proposed a hierarchical ranking model for the task of information insertion [2]. The task was stated as follows. We are given an existing document and a piece of new information represented as a sentence to be input, the task is to determine the best insertion point in the document to place the sentence. Experiments were conducted on Wikipedia articles. The second weakness the work in [2] is the lack of semantic information in extracting lexical features, which are mainly based on word overlap.

In this paper, we address the task of updating a legal document given new information. Our task aims to find the most appropriate section in a legal document, into which new information will be placed. We adopt the hierarchical ranking model presented in [2] and incorporate more semantic features derived from word clusters [1] into the model to improve the system performance. Experiments are conducted on a legal dataset built from the United States Code.

1. Processing method

1.1. Problem setting

Training data is a set of training instances. Each training instance is represented by a tuple (s, T, ℓ) where s represents an input sentence, T is an existing legal document, and ℓ represents the correct section in the document T, into which the s will be inserted. The document T is represented as a tree in which leaf nodes are sections in the document.

For a new pair of sentence-document (s, T), we need to find the most appropriate section ℓ in T, in which to place the sentence s. In order to do that, all sections in the document need to be ranked by a ranking function, and then the section with the highest score will be chosen. Therefore, the task can be reduced to learning the ranking function from training data.

1.2. The hierarchical ranking model

Given an existing document T represented as a tree, and an input sentence s, each pair of the sentence s and a node n in T is associated with a feature vector $\phi(s, n)$. Denote the set of leaf nodes by $L(T)$, and the path from the root of the tree to a node n by $P(n)$. The aggregate feature vector $\Phi(s, \ell)$ associated with a leaf node ℓ is computed by summing up all feature vectors of nodes in the path from the root to ℓ.

$$\Phi(s, \ell) = \sum_{n \in P(\ell)} \phi(s, n) \tag{1}$$

The model consists of a weight vector $\mathbf{w}$, each weight corresponding to a single feature. The following formula is used to determine the section in T, in which to place the sentence s.

$$\hat{\ell} = \arg\max_{\ell \in L(T)} \mathbf{w}.\phi(s, \ell) \tag{2}$$

In the training procedure, average Perceptron algorithm [3] was applied. The advantage of the online Perceptron algorithm is that its implementation is simple and it is memory-efficient when the number of training instances is large.

1.3. Feature Design

Word-based Lexical Features

Lexical features aim to capture the topical overlap of the input sentence and a section in the document. Word-based lexical features at section-level are computed basing on word overlap or text similarity score of the pair (input sentence, section).

Features based on Word Clustering

In many natural language processing tasks, word clustering has been utilized to tackle to the problem of data sparseness by providing a lower-dimensional representation of words: for example, Dependency Parsing [4]. The common method is as follows. First,

Reprinted from Frontiers in Artificial Intelligence and Applications, Volume 223, Minh Quang Nhat PHAM, Minh Le NGUYEN and Akira SHIMAZU, Update Legal Documents Using Hierarchical Ranking Models and Word Clustering, pp.163-166,

word clusters are obtained by running a word clustering algorithm on a large raw text corpus. Then, cluster-based features are extracted and incorporated into the learning model. Word clusters are used as intermediate word representations, and semantic relations among words hidden in clusters can be exploited.

We used English word cluster data from [4], including 1000 word clusters. The word clusters were derived from BLLIP corpus, including 43 million words of Wall Street Journal text. Input of the Brown algorithm is a large raw text corpus, and output is a hierarchy of word clusters. Each word in the word cluster corpus is represented as a binary string. Words having the same binary string representation belong to the same cluster.

Our method of extracting cluster-based features is as follows. For each pair of an insertion sentence s and a node n at a certain level in the document tree, the binary string representation for each word in the insertion sentence s and in the node n from word clusters set was obtained at first. Second, we compute the text similarity of two text segments s and n based on their binary string representations. Finally, text similarity scores obtained in the second step was incorporated into the learning model as additional features. In the second step, we employed TF-IDF weighted cosine similarity function, Jaccard similarity function, and lexical matching function to compute text similarity of two text segments.

2. Experiments and Results

2.1. Dataset

Since there is no dataset for the updating task in Legal domain, we built the Legal dataset from the United States Code [5]. We built the dataset automatically by recording documents before and after randomly removing a sentence from each of them. Totally, we obtained 1812 insertion sentence/document pairs from 18 legal documents. We used 1450 pairs (80%) for training and 362 (20%) for testing. Legal documents in our dataset are very long documents. Average document has 1472.4 sentences, organized into 141.9 sections.

2.1.1. Evaluation measures

a) *Accuracy in choosing sections* is the percentage of correct predictions.
b) *N-best accuracy* is computed in the same way as computing accuracy in choosing section, except that a prediction will be judged correct if the correct section is in the top N sections returned by the ranker. In experiments, we choose $N = 5$ and $N = 10$.

2.1.2. Baselines

In order to investigate the effects of cluster-based features on the performance of methods, we conducted experiments with three settings: In the first setting, we used only word-based features; in the second setting, we only used cluster-based features; and in the third setting, we combined word-based features with cluster-based features.

We use Flat method and Unsupervised method as baselines. Flat method is the method in which the model is trained by standard Perceptron algorithm, without using decomposition of features in the Equation 1. Unsupervised method uses Vector Space Model with TF-IDF weighting scheme.

Reprinted from Frontiers in Artificial Intelligence and Applications, Volume 223, Minh Quang Nhat PHAM, Minh Le NGUYEN and Akira SHIMAZU, Update Legal Documents Using Hierarchical Ranking Models and Word Clustering, pp.163-166,

Table 1. Results on Legal dataset

		Section (%)	5-best (%)	10-best (%)
	Unsupervised	41.4	75.4	85.0
Word-based Features	Flat	47.8	76.2	85.3
	Hierarchical	50.9	81.8	89.1
Cluster-based Features	Flat	46.0	73.8	83.5
	Hierarchical	49.6	79.8	88.0
Word-based + Cluster-based features	Flat	49.5	80.0	87.0
	Hierarchical	52.3	83.0	90.1

2.2. Results

Table 1 shows the experimental results on the legal dataset. The results indicate that the Hierarchical model outperforms both the Unsupervised method and the Flat method. The best performance was obtained when combining word-based features with cluster-based features. In that setting, the Hierarchical model obtained 52.3% accuracy in choosing sections; 5-best accuracy and 10-best accuracy are 83.0% and 90.1% respectively.

3. Conclusion

Updating legal documents when new information emerges is a challenging task, due to the large number of legal documents and legal updates. In this paper, we have presented a hierarchical ranking model for the task of updating legal documents. Features based on word clustering are incorporated into the ranking model to exploit semantic relations among words and improve the system performance. In order to evaluate the proposed method, a legal dataset was constructed. Despite disadvantages of the automatically-constructed data, our research indicated that the hierarchical ranking model is a potential solution for the task of updating legal documents.

Acknowledgement. This research was partly supported by the 21st Century COE Program 'Verifiable and Evolvable e-Society' and Grant-in-Aid for Scientific Research (19650028 and 20300057).

References

[1] Brown, P. F., Della Pietra, V. J., deSouza, P. V., Lai, J. C., and Mercer, R. L. (1992). Class-based n-gram models of natural language. Computational Linguistics, 18(4), pp. 467-479.
[2] Chen, E., Snyder, B., and Barzilay, R. (2007). Incremental text structuring with online hierachical ranking. In Proceedings of the EMNLP, pp. 83-91.
[3] Collins, M. (2002). Discriminative training methods for Hidden Markov Models: Theory and experiments with Perceptron algorithms. In Proceedings of the EMNLP, pp. 1-8.
[4] Koo, T., Carreras, X., and Collins, M. (2008). Simple semi-supervised dependency parsing. In Proceedings of ACL-08, pp. 595-603.
[5] United States Code. Retrieved from the Website of the U.S. Government Printing Office: http://www.gpoaccess.gov/uscode/about.html