Sentence Compression Based on ILP Decoding Method

Hongling Wang, Yonglei Zhang, and Guodong Zhou

Natural Language Processing Lab, Soochow University, Suzhou, Jiangsu, 215006
School of Computer Science & Technology, Soochow University, Suzhou, Jiangsu, 215006
{hlwang,20104227009,gdzhou}@suda.edu.cn

Abstract. With the tremendous increasing of information, the demands of information from people advanced the development of Nature Language Processing (NLP). As a consequent, Sentence compression, which is an important part of automatic summarization, draws much more attention. Sentence compression has been widely used in automatic title generation, Searching Engine, Topic detection and Summarization. Under the framework of discriminative model, this paper presents a decoding method based on Integer Linear Programming (ILP), which considers sentence compression as the selection of the optimal compressed target sentence. Experiment results show that the ILP-based system maintains a good compression ratio while remaining the main information of source sentence. Compared to other decoding method, this method has the advantage of speed and using fewer features in the case of similar results obtained.

Keywords: Sentence Compression, Integer Linear Programming, Structured Learning.

1 Introduction

Recent years have witnessed increasing interest in text-to-text generation methods for many natural language processing applications, ranging from text summarization to question answering and machine translation. At the heart of these methods lies the ability to perform rewriting operations. Sentence compression is perhaps one of the most popular text-to-text rewriting methods. The aim is to produce a summary of a single sentence that retains the most important information while remaining grammatical.

The appeal of sentence compression lies in its potential for summarization and more generally for document compression, e.g., for displaying text on small screens such as mobile phones or PDAs (Corston-Oliver, 2001). Vandeghinste & Pan (2004) generated a title for a dialogue by deleting redundant and non-critical information while retaining the main idea. Another earlier sentence compression application is the use of voice reading devices for the blind (Grefenstette 1998). Text is compressed for voice machine to accelerate reading speed, which enables the blind way of reading similar to a normal way of speed reading. Thus, the research on sentence compression for obtaining useful information has important significance.

We define the sentence compression task as follows: given an input sentence, to produce a sentence which is shorter and retains the important information from the original, and also it is grammatical. In our paper, sentence compression aims to shorten a sentence $x=l_1,l_2,\ldots\ldots,l_n$ into a substring $y^*=c_1,c_2,\ldots\ldots c_m$, where $c_i \in \{ l_1,l_2,\ldots\ldots,l_n \}$. We define the function $F(c_i) \in \{1, \ldots, n\}$ that maps word c_i in the compression to the index of the word in the original sentence. Then, we include the constraint $F(c_i) < F(c_{i+1})$, which forces each word in x to occur at most once in the compression y^*, so in the compression process we don't change the word's order and only delete words or phrases. This paper implements a Chinese sentence compression system by learning a sub-tree from the source parsing tree of a sentence (See Figure 1).

Example:

Original Sentence: 据 法新社 报道 ， 有 目击者 称 ， 以军 23日 空袭 加沙 地带 中部 ， 目前 尚 无 伤亡 报告 。

Pinyin: ju faxinshe baodao , you mujizhe cheng , yijun 23ri kongxi jiasha didai zhongbu , muqian shang wu shangwang baogao .

Target Sentence: 目击者 称 以军 空袭 加沙 地带 中部

Pinyin: mujizhe cheng yijun kongxi jiasha didai zhongbu

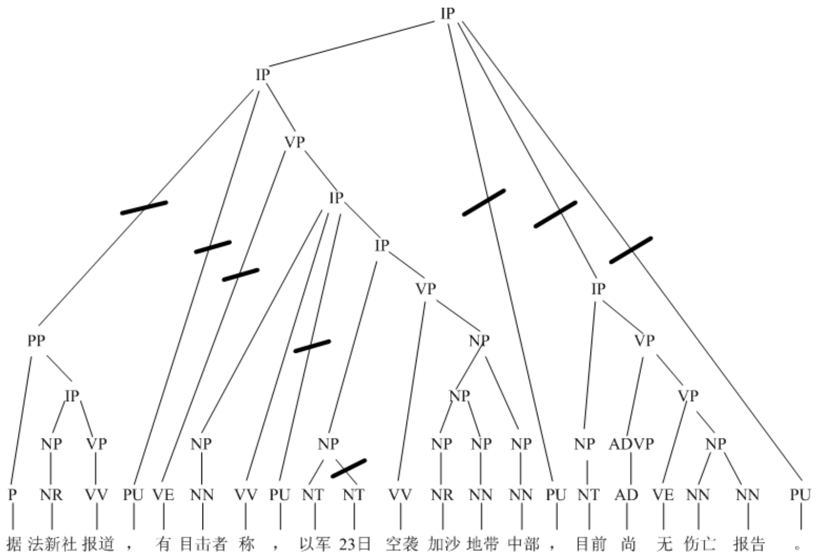

Fig. 1. Compression example, the coarse slashes means in the compressed target sentence the edges are deleted

The rest of this paper is organized as follows: Section 2 briefly reviews the related work on sentence compression and the applications of integer linear programming in the field of NLP. Section 3 describes our sentence compression system, including how to formulate sentence compression problem as an ILP problem, features, loss function and evaluations. Section 4 presents the experimental results. Finally, Sections 5 draws the conclusion.

2 Related Work

2.1 Sentence Compression

Currently, the mainstream solutions to sentence compression problem have been cast mostly in corpus-driven supervised learning models which can be divided into categories: generative model and discriminative model.

Generative model selects the optimal target sentence by estimating the joint probability *P(x, y)* of original sentence *x* having the target sentence *y*. The main advantage of this method is the model training process is simple. And also its parameters can be easily got by counting different context transferring grammars in a parallel corpus. Knight & Marcu (2002) firstly apply the noisy-channel model (one of generative models) for sentence compression. Though the performance of the noisy-channel model is quite well, they do have their shortcomings, such as the source model which represent the probability of compressed sentences, but it is trained on uncompressed sentences, and the channel model requires aligned parse trees for both compressed and uncompressed sentences in the training set in order to calculate probability estimates. These parse trees with many mistakes for both the original and compressed versions will make alignment difficult and the channel probability estimates unreliable as a result.

Discriminative model can be used the rich features to help identify special language phenomenon during the training process. These features may be interrelated and do not meet the independent condition which must meet in the generative model-based method. Discriminative model has been widely used and achieved good performance in many natural language processing tasks, such as the tasks of dependency parsing (McDonald et al., 2005b), entity extraction (Sang & Meulder, 2003), and relation extraction (Zelenko et al. 2003).

Currently, sentence compression is often modeled in a discriminative framework, including the decision tree model, the compression model based on online learning[9], and the model based on SVM.

Knight & Marcu (2002) use the decision tree model to implement the compression by learning a decision tree to incrementally convert between original parse trees and compressed parse trees. There are four operations defined during the process: SHIFT (transfer the first word from the input list into the stack)、REDUCE (pop the k syntactic trees located at the top of the stack; combine them into a new tree; and push the new tree on the top of the stack), DROP (delete from the input list subsequences of words that correspond to syntactic constituents), ASSIGNTYPE (change the label of trees at the top of the stack). This model avoid the unreliable of the tree alignment, but their model features encode properties related to including or dropping constituents from the tree with no encoding of bigram or trigram surface features to promote grammaticality. As a result, the model will generate some short and ungrammatical targets.

McDonald(2006) used max-margin leaning algorithm (MIRA, margin-infused relaxed algorithm) to study the feature weight, then rank the subtrees, and finally select the tree with the highest score as the optimal target sentence. McDonald's work had achieved a well performance by using the manual evaluations. But manual evaluations have some disadvantages, such as heavy workload, strength subjectivity, and so on. In addition, the reliability of the learning algorithm of MIRA is not better than Structured SVM (Tsochantaridis et al., 2005).

Cohn & Lapata (2007, 2008, and 2009) formulated the compression problem as tree-to-tree rewriting using a synchronous grammar. Each grammar rule is assigned a weight which is learned discriminatively within a large margin model. A specialized algorithm is used to learn the model weights and find the best scoring compression under the model. This method achieves comparable performance with McDonald's model. The main reason for limiting the performance of the model is that the model needs to do alignment between noisy syntactic trees. Zhang et al. (2013) compressed sentences based on Structured SVM model which treats the compression problem as a structured learning problem, i.e., to learn an optimal sub-tree as its compressed sentence on original sentence parse tree. The experimental results showed that it can generate target sentence which is grammatical and contains the center information of the original sentence in the case of ensuring a better compression rate.

2.2 Integer Linear Programming in NLP

ILPs are constrained optimization problems where both the objective function and the constraints are linear equations with integer variables. Integer linear programming has been applied to many natural language processing tasks, such as relation extraction (Roth & Yih, 2004), semantic role labeling (Punyakanok, 2004), syntactic parsing (Riedel & Clarke, 2006) and so on. Most of these approaches combine a local classifier with an inference procedure based on ILP. The classifier proposes possible answers which are assessed in the presence of global constraints. ILP is used to make a final decision that is consistent with the constraints and likely according to the classifier. For example, the argument for a predicate is the role of non-repetition in semantic role labeling, which can be implemented by adding global non-repeating linear constrains.

Gillick et al. (2009) and Berg-Kirpatrick et al. (2011) applied ILP to multi-document summarization task. They constructed linear constrains for the feature space corresponding to the decoding space, then used ILP to select the optimal predication target according to the feature weight.

Clarke & Lapata (2008) viewed sentence compression as an optimization problem and uses integer linear programming to infer globally optimal compressions in the presence of linguistically motivated constraints. Experimental results on written and spoken texts demonstrate improvements over state-of-the-art models. Woodsend & Lapata (2011) proposed an integer linear programming model for selecting the most appropriate simplification from the space of possible rewrites generated by the grammar. Experimental results showed that the method creates simplifications that significantly reduce the reading difficulty of the input, while maintaining grammaticality and preserving its meaning. Those studies show that ILP could be used for sentence compression or simplification in English. We will use the method in Chinese sentence compression in this paper.

3 Sentence Compression Based on ILP

In this paper, we treat the sentence compression problem as a structured learning problem, i.e. a subtree which learned from the original sentence parse tree is as its compressed sentence. Thus sentence compression task is converted to the task of how to choose the optimal subtree on original sentence syntactic tree. Here we formulate the problem of finding optimal subtree to an ILP decoding problem, that is, to find the target optimal subtree by using ILP decoding method.

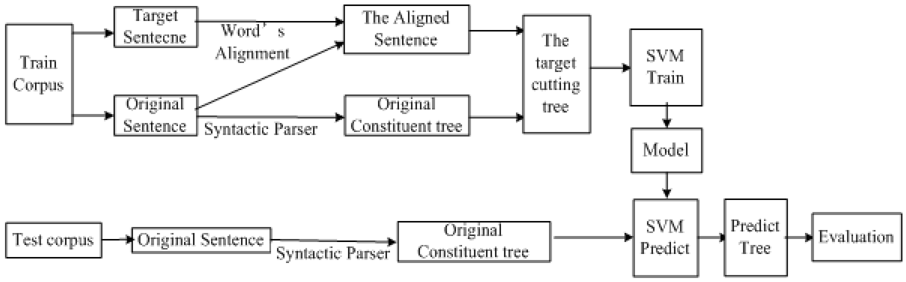

Fig. 2. The Framework of Chinese Sentence Compression

This paper uses the structured learning framework in Zhang et al. (2013) (See Figure 2). After preprocessing for the corpus, the system extracts the features which generated during the original constituent tree transformed into the target tree, and then uses the SVM to train feature weights, finally selects the tree with the highest score as the best target tree.

3.1 Linear Objective Function

Assuming the original sentence x has n words, and then the target set has 2^n elements. With the increasing number of the word the original sentence has, decoding set exponential growth. Finding an optimal target sentence in such a large decoding space, time complexity is very large. Zhang (2012) used McDonald's simplify method to decode. In this paper, we formulate the problem of finding the optimal sentence as an ILP decoding problem. Each subtree is ranked according to the trimming features and lexical features; the subtree with the highest score is the optimal target one.

Suppose x is the original sentence syntactic tree, y is the target subtree corresponding to x. Here we define two vectors: $R(y)$ represents the word set of y, $P(y)$ represents the operation set from x to y. The problem of finding the optimal subtree is transformed to solve the maximum value of following objective function.

$$score(y,x) = \arg\max_{y \in Y(x)} \sum_{r \in R(y)} v_r + \sum_{p \in P(y)} v_p \qquad (1)$$

where $Y(x)$ is the target subtree set, v_r is the weight of word r, v_p is the weight of operation p.

Before the weights are learned by machine learning, the word *r* and the operation *p* in Equation (1) need be parameterized. Suppose: *w* is the vector of feature weight, $g(r,y)$ and $h(p,y)$ represent the feature functions of words and operations to target sentence respectively, then:

$$\begin{aligned} v_r &= < w, g(r, y) > \\ v_p &= < w, h(p, y) > \end{aligned} \tag{2}$$

where $g(r,y)$ includes the features of POS tags, is or not stop word, bigram of POS, etc; $h(p,y)$ includes the features of the parent node of deleting node, the parent-child structure of deleting node, etc.

After v_r and v_p parameterized, we assume that $f(y,x)$ is the feature function of bigram and trimming features from *x* to *y*, then the Equation (1) can be transformed to the following form:

$$score(y,x) = \arg\max_{y \in Y(x)} \sum_{r \in R(y)} < w, g(r,y) > + \sum_{p \in P(y)} < w, h(p,y) > \tag{3}$$

$$score(y,x) = \arg\max_{y \in Y(x)} < w, f(y,x) > \tag{4}$$

3.2 Linear Constrain

According to the previous section, to find the optimal subtree is to get the optimal solution for Equation (4). In this paper, we define a bigram indicator variable n_i (if n_i=1, the i^{th} node is remained; n_i=0, the node is dropped) for each non-terminal node of the original constituent tree. In order to maintain the tree structure, its children nodes are all deleted when the i^{th} node is deleted, which can be implemented by adding the following linear constrain: $n_i - n_j \geq 0$ 、 $n_i \leq \sum n_j$, where n_i is the parent node of n_j.

Similarly, we also define a bigram indicator variable w_i (if w_i=1, the i^{th} node is remained; w_i=0, the node is dropped) for each terminal node of the original constituent tree. And a linear constrain is added: $w_i = n_j$, where n_j is the POS node of word w_i. At last, a bigram indicator variable f_i (if f_i=1, the i^{th} feature appears; or, the feature doesn't appear) is defined for the i^{th} feature. According to the restrictions of feature value, the corresponding linear constrains are added.

3.3 Features

In this paper, we mainly adopt the features which are used in Zhang (2012). There are two kinds of features: Word/POS Features and syntax features. Here in order to avoid too slow to decode the ILP problem, we redefine and extract some features under the framework of structure learning.

Due to the small size of our corpora, which will lead to sparseness and over-fitting, we mainly extract the feature of the word's POS and rarely include the word itself. In this paper, following features are used: the remaining word's bigram POS (PosBigram

(目击者 称) = NN&VV[1]), whether the dropped word is a stop word (IsStop (据) = 1), whether the dropped word is the headword of the original sentence, the number of remaining words. The features of word's bigram POS used in this paper refer to the features in original sentence instead of the ones in target sentence.

In our experiments, two following syntax features are included: the parent-children relationship of the cutting edge (del-Edge (PP) = IP-PP) and the number of the cutting edge. In addition, following dependency features are included: the dependant relation between the dropped word and its dependence word (dep_type(有)=DEP), the relation chain of the dropped word's POS with its dependence word's POS (dep_link (,) = PU-VMOD-VV), whether the dependence tree's root is deleted (del_ROOT (无) = 1), and whether each dropped word is a leaf of the dependence tree (del_Leaf (法新社) = 1).

3.4 Linear Constrains to Features

Due to different limitations of different feature values, linear constrains need to be separately defined for different types of features. For example, for the word "据", since the word is a stop word, the value of its feature indicator variable f_i depends on the value of word indicator variable w_i. When w_i=1, the word will be remained in target sentence, then $f_i = 0$; Or, $f_i = 1$. Therefore, the linear constrain can be defined as f_i=1-w_i.

In our experiments, we find that the solving time of the ILP problem is getting longer with the increasing of constrains. Especially for those bigram features of the words, there are (n-1)*n/2 constrains (n is the number of words) which lead to much longer solving time. So we use the remaining word's bigram POS as a feature of word to the original sentence.

3.5 Loss Function

In our work, loss function means the difference between the predict sentence and the gold target sentence. The selection of the loss function has great influence on system performance. In our earlier experiments, we used loss ratio of the remaining word's bigram as the loss function which lead to a decrease of decoding speed because of its large constrains. In this paper, two loss functions are tested. One is the loss ratio of bigram of the remaining word in original sentence. The other is the sum of the number of the words deleted by mistake and the number of the words remained by mistake between the predict sentence and the gold target sentence, which called word loss-based function. Since we found the latter better in our experiments, the word loss-based function is used in the follow-up experiments.

3.6 Evaluation

Currently, manual evaluation is commonly used for sentence compression. Importance and Grammaticality proposed by Knight & Marcu (2002) are mainly used in

[1] In Figure 1, for example (the same below).

many works. They are ranked a sentence in a scale from 1 to 5. Importance means how well the systems did with respect to selecting the most important words in the original sentence, while Grammaticality means how grammatical the target sentences were. Although manual evaluations have some disadvantages, such as heavy workload, strength subjectivity, and so on, they still have the advantage of high accuracy compared to automatic evaluation, which makes them to be used widely.

In this paper, we also use the two evaluations to evaluate our system. In addition, we use sentence similarity and compression ratio (CR) as automatic evaluation metric. Here BLEU score which usually used in machine translation task is introduced as similarity evaluation to compare the n-gram difference between the predict sentence and the gold target sentence. Since sentence compression can be seen as translating the original sentence into shorter sentence with same language, the BLEU score can be also used to evaluate the compression performance. To better adapt to sentence compression we redefine the parameters of BLUE.

$$Bleu = BP * \exp(\sum_{n=1}^{N} w_n \log p_n) \tag{5}$$

where p_n is the ratio of the number of n consecutive words in candidate compressed sentence occupies the number of n consecutive words in gold target sentence. w_n=1/N, N is the largest n-gram order. The penalty factor BP is redefined as follows:

$$BP = \begin{cases} 1 & if\ c > r \\ e^{(1-r/c)} & if\ c \le r \end{cases} \tag{6}$$

where c is the number of words of candidate compressed sentence, and r is the number of gold target sentence.

4 Experiments

In our experiments, we use the same parallel corpus extracted from news documents and also extend the corpus using same expansion mode in Zhang (2012), i.e. 2400 pair sentences as training set, and 100 pair sentences as test set. At first, the original sentences in the corpus are parsed using the open-source tool Stanford Parser[2]. Then the words in sentence pair are aligned by the tool developed by our own. Although other open-source kits, such as Giza++, Berkeley Aligner etc, can also be used, we don't use them for their poor performance in the same language. And we use Structured SVM (Tsochantaridis etc. 2005) to learn feature weights. The tool we used is an open-source tool SVM$^{\text{struct}}$[3]. The convergence ε is set to 10^{-4} in the training process.

[2] http://nlp.stanford.edu/software/lex-parser.shtml

[3] http://download.joachims.org/svm_struct/current/svm_struct.tar.gz

4.1 Selection of Upper and Lower Bound to Compression Ratio

Since we use an ILP decoding method to find the optimal sub-tree, it has a large decoding space. We can narrow the feasible space by adding the following linear constrain.

$$CR_{lower} * n \leq \sum_{i=1}^{n} w_i \leq CR_{up} * n \tag{7}$$

where, CR_{lower} is the lower bound of compression ratio, CR_{up} is the upper bound of compression ratio, n is the number of words in the original sentence. To select the appropriate lower and upper bound of compression ratio has a greater influence on the system performance.

Since the target compressed sentences in the corpus come from documents' title, they have a relatively uniform length. Moreover, due to the higher compression ratio and using the feature of the remaining word's bigram POS, the system will tend to generate shorter and not well grammatical compressed target. So we set the values of lower bound and upper bound at the same time. In the experiments, lower bound is set to 0.7, and upper bound is set to 10.

4.2 Experimental Results and Discussion

Table 1 shows the manual and automatic results of our Chinese sentence compression system. In the Table 1, the McDonald row shows the result of Zhang et al. (2003) which used McDonald's decoding model and the ILP row shows the result of our system used ILP decoding model.

Table 1. Results of various experiments

Model	Manual Evaluations		Automatic Evaluations	
	Importance	Grammaticality	CR	BLEU
Gold	4.335±0.265	4.977±0.077	0.291	
McDonald	4.200±0.562	4.444±0.776	0.401	0.686±0.160
ILP	4.190±0.578	4.390±0.827	0.3783	0.688±0.166

From the table 1, we can see that:

1. Compared to McDonald's model, the evaluation of CR improved 2.2% shows that using the ILP decoding method can find an optimal target sentence with better compression ratio.
2. Compared to McDonald's model, the manual evaluations (Importance and Grammaticality) are decreased slightly for two reasons. One is that our system uses simple features because many features into linear constrains is more complex. The other is the more features mean the slower decoding speed, so the number of features is less than the one in McDonald's model.
3. The BLUE scores are similar whether using McDonald's decoding method or ILP method.

In summary, compared to the McDonald's decoding method, the system based ILP decoding method achieves a comparable performance using simpler and less features. We can also be expected that the system performance will gradually improve with the number of features increases. In addition, from the operating efficiency perspective, the overall system's running time is less than before. Here are some data may illustrate the problem: for the same test set, the running time is 8.6 seconds using ILP decoding while 18.4 seconds using McDonald's decoding method.

5 Conclusions

In this paper, under the framework of structured learning, the problem of sentence compression is formulated as a problem of finding an optimal sub-tree using ILP decoding method. Compared to the previous work using McDonald's decoding method, the system which only uses simpler and fewer features achieves a comparable performance on same conditions. And this method has the advantage of speed and using fewer features in the case of similar results obtained. In the future, we will explore more efficient features and linear constrains to use in the ILP decoding method.

Acknowledgements. This research is supported by Project 10KJB520016 under the High Educational Natural Science Foundation of Jiangsu, China.

References

1. Corston-Oliver, S.: Text Compaction for Display on Very Small Screens. In: Proceedings of the NAACL Workshop on Automatic Summarization, Pittsburgh, PA, pp. 89–98 (2001)
2. Clarke, J., Lapata, M.: Global Inference for Sentence Compression An Integer Linear Programming Approach. Journal of Artificial Intelligence 31, 399–429 (2008)
3. Vandeghinste, V., Pan, Y.: Sentence compression for automated subtitling: A hybrid approach. In: Marie-Francine Moens, S.S. (ed.) Text Summarization Branches Out: Proceedings of the ACL 2004 Workshop, Barcelona, Spain, pp. 89–95 (2004)
4. Grefenstette, G.: Producing Intelligent Telegraphic Text Reduction to Provide an Audio Scanning Service for the Blind. In: Hovy, E., Radev, D.R. (eds.) Proceedings of the AAAI Symposium on Intelligent Text Summarization, Stanford, CA, USA, pp. 111–117 (1998)
5. Knight, K., Marcu, D.: Summarization beyond sentence extraction: a probabilistic approach to sentence compression. Artificial Intelligence 139(1), 91–107 (2002)
6. McDonald, R., Crammer, K., Pereira, F.: Online large-margin training of dependency parsers. In: 43rd Annual Meeting of the Association for Computational Linguistics, Ann Arbor, MI, USA, pp. 91–98 (2005b)
7. Sang, E.F.T.K., Meulder, F.: Introduction to the conll-2003 shared task: language-independent named entity recognition. In: Proceedings of the Seventh Conference on Natural Learning at HLT-NAACL 2003, pp. 142–147. Association for Computational Linguistics, Morristown (2003)
8. Zelenko, D., Aone, C., Richardella, A.: Kernel methods for relation extraction. J. Mach. Learn. Res. 3, 1083–1106 (2003)

9. McDonald, R.: Discriminative sentence compression with soft syntactic constraints. In: Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics, Trento, Italy, pp. 297–309 (2006)
10. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large margin methods for structured and interdependent output variables. Journal of Machine Learning Research 6, 1453–1484 (2005)
11. Cohn, T., Lapata, M.: Large margin synchronous generation and its application to sentence compression. In: Proceedings of the EMNLP/CoNLL 2007, Prague, Czech Republic, pp. 73–82 (2007)
12. Cohn, T., Lapata, M.: Sentence compression beyond word deletion. In: Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), Manchester, UK, pp. 137–144 (2008)
13. Cohn, T., Lapata, M.: Sentence Compression as Tree Transduction. Journal of Artificial Intelligence Research 34, 637–674 (2009)
14. Roth, D., Yih, W.: A linear programming formulation for global inference in natural language tasks. In: Proceedings of the Annual Conference on Computational Natural Language Learning, Boston, MA, USA, pp. 1–8 (2004)
15. Punyakanok, V., Roth, D., Yih, W., Zimak, D.: Semantic role labeling via integer linear programming inference. In: Proceedings of the International Conference on Computational Linguistics, Geneva, Switzerland, pp. 1346–1352 (2004)
16. Riedel, S., Clarke, J.: Incremental integer linear programming for non-projective dependency parsing. In: Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, Sydney, Australia, pp. 129–137 (2006)
17. Gillick, D., Favre, B.: A scalable global model for summarization. In: Proc. of ACL Workshop on Integer Linear Programming for Natural Language Processing, Boulder, Colorado, pp. 10–18 (2009)
18. Berg-Kirkpatrick, T., Gillick, D., Klein, D.: Jointly Learning to Extract and Compress. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, Portland, Oregon, pp. 481–490 (2011)
19. Woodsend, K., Lapata, M.: Learning to Simplify Sentences with Quasi-Synchronous Grammar and Integer Programming. In: EMNLP 2011, pp. 409–420 (2011)
20. Zhang, Y.L., Wang, H.L., Zhou, G.D.: Sentence Compression Based on Structured Learning. Journal of Chinese Information Processing 27(2), 10–16 (2013)
21. Zhang, Y., Peng, C., Wang, H.: Research on Chinese Sentence Compression for the Title Generation. In: Ji, D., Xiao, G. (eds.) CLSW 2012. LNCS, vol. 7717, pp. 22–31. Springer, Heidelberg (2013)