A New Chinese Natural Language Understanding Architecture Based on Multilayer Search Mechanism

Wanxiang Che **Ting Liu** **Sheng Li**
School of Computer Science and Technology
Harbin Institute of Technology
P.O. Box 321, HIT
Harbin China, 150001
{car, tliu, ls}@ir.hit.edu.cn

Abstract

A classical Chinese Natural Language Understanding (NLU) architecture usually includes several NLU components which are executed with some mechanism. A new Multilayer Search Mechanism (MSM) which integrates and quantifies these components into a uniform multilayer treelike architecture is presented in this paper. The mechanism gets the optimal result with search algorithms. The components in MSM affect each other. At last, the performance of each component is enhanced. We built a practical system – CUP (Chinese Understanding Platform) based on MSM with three layers. By the experiments on Word Segmentation, a better performance was achieved. In theory the normal cascade and feedback mechanism are just some special cases of MSM.

1 Introduction

At present a classical Chinese NLU architecture usually includes several components, such as Word Segmentation (Word-Seg), POS Tagging, Phrase Analysis, Parsing, Word Sense Disambiguation (WSD) and so on. These components are executed one by one from lower layers (such as Word-Seg, POS Tagging) to higher layers (such as Parsing, WSD) to form a kind of cascade mechanism. But when people build a NLU system based on these complex language analysis, it is a very serious problem since the errors of each layer component are multiplied. With more and more analysis components, the final result becomes too bad to be applicable.

Another problem is that the components in the system affect each other when people build a practical but toy NLU system. Here the toy system means that each component is ideal enough with perfect input. But in fact, on the one hand the lower layer components need the information of higher layer components; on the other hand the incorrect analysis of lower layers must reduce the accuracy of higher layers. In Chinese Word-Seg component, many segmentation ambiguities which cannot be solved using only lexical information. In order to improve the performance of Word-Seg, we have to use some syntax and even semantic information. Without correct Word-Seg results, however the syntax and semantic parser cannot obtain a correct analysis. It is a chain debts problem.

People have tried to solve the error-multiplied problem by integrating multi-layers into a uniform model (Gao et al., 2001; Nagata, 1994). But with the increasing number of integrated layers, the model becomes too complex to build or solve.

The feedback mechanism (Wu and Jiang, 1998) helps to use the information of high layers to control the final result. If the analysis at feedback point cannot be passed, the whole analysis will be denied. This mechanism places too much burden on the function of feedback point. This leads to the problems that a correct lower layer result may be rejected or an error result may be accepted.

We propose a new Multilayer Search Mechanism (MSM) to solve the problems mentioned above. Based on the mechanism, we build a practical Chinese NLU platform – CUP (Chinese Understanding Platform). Section 2 introduces the background and architecture of the new mechanism and how to build it up. Experimental results with CUP is given in Section 3. In Section 4, we discuss why the new mechanism gets better results than the old ones. Conclusions and the some future work follow in Section 5.

2 Multilayer Search Mechanism

The novel Multilayer Search Mechanism (MSM) integrates and quantifies NLU components into a uniform multilayer treelike platform, such as Word-Seg, POS Tagging, Parsing and so on. These components affect each other by computing the final score and then get better results.

2.1 Background

Considering a Chinese sentence, the sentence analysis task can be formally defined as finding a set of word segmentation sequence (W), a POS tagging sequence (POS), a syntax dependency parsing tree (DP) and so on which maximize their joint probability $P(W, POS, DP, \cdots)$. In this paper, we assume that there are only three layers W, POS and DP in MSM. It is relatively straightforward, however, to extend the method to the case for which there are more than three layers. Therefore, the sentence analysis task can be described as finding a triple $< W, POS, DP >$ that maximize the joint probability $P(W, POS, DP)$.

$$< W, POS, DP >= \arg\max_{W,POS,DP} P(W, POS, DP)$$

The joint probability distribution $P(W, POS, DP)$ can be written in the following form using the chain rule of probability:

$$\begin{aligned} & P(W, POS, DP) \\ = & P(W)P(POS|W)P(DP|W, POS) \end{aligned}$$

Where $P(W)$ is considered as the probability of the word segmentation layer, $P(POS|W)$ is the conditional probability of POS Tagging with a given word segmentation result, $P(DP|W, POS)$ is the conditional probability of a dependency parsing tree with a given word segmentation and POS Tagging result similarly. So the form of $< W, POS, DP >$ can be transformed into:

$$\begin{aligned} & < W, POS, DP > \\ = & \arg\max_{W,POS,DP} P(W, POS, DP) \\ = & \arg\max_{W,POS,DP} P(W)P(POS|W)P(DP|W, POS) \\ = & \arg\max_{W,POS,DP} \log P(W) + \log P(POS|W) \\ & + \log P(DP|W, POS) \\ = & \arg\min_{W,POS,DP} -\log P(W) - \log P(POS|W) \\ & - \log P(DP|W, POS) \end{aligned}$$

We consider that each inversion of probability's logarithm at the last step of the above equation is a score given by a component (Such as Word-Seg, POS Tagging and so on). So at last, we find an n-tuple $< W, POS, DP, \cdots >$ that minimizes the last score S_n of a sentence analysis result with n layers. S_n is defined as:

$$S_n = s_1 + s_2 + \cdots + s_n \quad (1)$$

s_i denotes the score of the ith layer component.

2.2 The Architecture of Multilayer Search Mechanism

Because there are lots of analysis results at each layer, it's a combinatorial explosion problem to find the optimal result. Assuming that each component produces m results for an input on average and there are n layers in a NLU system, the final search space is m^n. With the increasing of n, it's impossible for a system to find the optimal result in the huge search space.

The classical cascade mechanism uses a greedy algorithm to solve the problem. It only keeps the optimal result at each layer. But if it's a fault analysis result for the optimal result at a layer, it's impossible for this mechanism to find the final correct analysis result.

To overcome the difficulty, we build a new Multilayer Search Mechanism (MSM). Different from the cascade mechanism, MSM maintains a number of results at each component, so that the correct analysis should be included in these results with high probability. Then MSM tries to use the information of all layer components to find out the correct analysis result. Different from the feedback mechanism, the acceptance of an analysis is not based on a higher layer components alone. The lower layer components provide some information to help to find the correct analysis result as well.

According to the above idea, we design the architecture of MSM with multilayer treelike structure. The original input is root and the several analysis results of the input become branches. Iterating this progress, we get a bigger analysis tree. Figure 1 gives an analysis example of a Chinese sentence "他喜爱美丽的鲜花。" (He likes beautiful flowers). For the input sentence, there are several Word-Seg results with scores (the lower the better). Then for each of Word-Seg results, there are several POS Tagging results, too. And for each of POS Tagging result, the same thing happens. So we get a big tree structure and the correct analysis result is a path in the tree from the root to the leaf except for there is no correct analysis result in some analysis components.

A search algorithm can be used to find out the correct analysis result among the lowest score in the tree. But because each layer cannot give the exact score in Equation 1 as the standard score and the ability of analysis are different with different layers, we should weight every score. Then the last score is the linear weighted sum (Equation 2).

$$S_n = w_1 s_1 + w_2 s_2 + \cdots + w_n s_n \tag{2}$$

s_i denotes the score of the ith layer component which we will introduce in Section 3; w_i denotes the weight of the ith layer components which we will introduce in the next section.

In order to get the optimal result, all kinds of tree search algorithms can be used. Here the BEST-FIRST SEARCH Algorithm (Russell and Norvig, 1995) is used. Figure 2 shows the main algorithm steps.

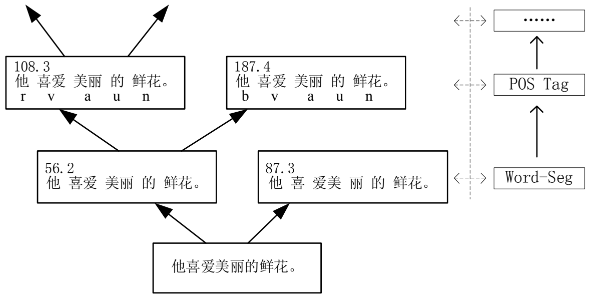

Figure 1: An Example of Multilayer Search Mechanism

1. Add the initial node (starting point) to the queue.
2. Compare the front node to the goal state. If they match then the solution is found.
3. If they do not match then expand the front node by adding all the nodes from its links.
4. If all nodes in the queue are expanded then the goal state is not found (e.g.there is no solution). Stop.
5. According to Equation 2 evaluate the score of expanded nodes and reorder the nodes in the queue.
6. Go to step 2.

Figure 2: BEST-FIRST SEARCH Algorithm

2.3 Layer Weight

We should find out a group of appropriate $w_1, w_2, \cdots, w_n$ in Equation 2 to maximize the number of the optimal paths in MSM which can get the correct results. They are expressed by W^*.

$$W^* = \arg\max_{W} ObjFun(\min S_n) \tag{3}$$

Here W^* is named as Whole Layer Weight. $ObjFun(*)$ denotes a function to value the result that a group of W can get. Here we can consider that the performance of each layer is proportional to the last performance of the whole system in MSM. So it maybe the F-Score of Word-Seg, precision of POS Tagging and so on. $\min S_n$ returns the optimal analysis results with the lowest score.

Here, the F-Score of Word-Seg can be defined as the harmonic mean of recall and precision of Word-Seg. That is to say:

$$Seg.F\text{-}Score = \frac{2 * Seg.Pre * Seg.Rec}{Seg.Pre + Seg.Rec}$$

$$Seg.Pre = \frac{\#\text{words correctly segmented}}{\#\text{words segmented}}$$

$$Seg.Rec = \frac{\#\text{words correctly segmented}}{\#\text{words in input texts}}$$

Finding out the most suitable group of W is an optimization problem. Genetic Algorithms (GAs) (Mitchell, 1996) is just an adaptive heuristic search algorithm based on the evolutionary ideas of natural selection and genetics to solve optimization problems. It exploits historical information to direct the search into the region of better performance within the search space.

To use GAs to solve optimization problems (Wall, 1996) the following three questions should be answered:

1. How to describ genome?
2. What is the objective function?
3. Which controlling parameters to be selected?

A solution to a problem is represented as a genome. The genetic algorithm then creates a population of solutions and applies genetic operators such as mutation and crossover to evolve the solutions in order to find the best one(s) after several generations. The numbers of population and generation are given by controlling parameters. The objective function decides which solution is better than others.

In MSM, the genome is just the group of W which can be denoted by real numbers between 0 and 1. Because the result is a linear weighted sum, we should normalize the weights to let $w_1 + w_2 + \cdots + w_n = 1$. The objective function is just $ObjFun(*)$ in Equation 3. Here the F-Score of Word-Seg is used to describe it. We set the genetic generations as 10 and the populations in one generation as 30. The Whole Layer Weight shows in the row of WLW in Table 4. The F-Score of Word-Seg shows as Table 3.

We can see that the Word-Seg layer gets an obviously large weight. So the final result is inclined to the result of Word-Seg.

2.4 Self Confidence

Our analysis indicates that the method of weighting a whole layer uniformly cannot reflect the individual information of each sentence to some component. So the F-Score of Word-Seg drops somewhat comparing with using Only Word-Seg. For example, the most sentences which have ambiguities in Word-Seg component are still weighted high with Word-Seg layer weight. Then the final result may still be the same as the result of Word-Seg component. It is ambiguous, too. So we must use a parameter to decrease the weight of a component with ambiguity. It is used to describe the analysis ability of a component for an input. We name it as Self Confident (SC) of a component. It is described by the difference between the first and the second score of a component. Then the bigger SC of a component, the larger weight of it.

There are lots of methods to value the difference between two numbers. So there are many kinds of definitions of SC. We use A and B to denote the first and the second score of a component respectively. Then the SC can be defined as $B - A$, $\frac{B}{A}$ and so on. We must select the better one to represent SC. The *better* means that a method which gets a lower Error Rate with a threshold t^* which gets the Minimal Error Rate.

$$t^* = \arg\min_t ErrRate(t)$$

$ErrRate(t)$ denotes the Error Rate with the threshold t. An error has two definitions:

- SC is higher than t but the first result is fault
- SC is lower than t but the first result is right

Then the Error Rate is the ratio between the error number and the total number of sentences.

Table 2 is the comparison list between different definitions of SC and their Minimal Error Rate of Word-Seg. By this table we select $B-A$ as the last SC because it gets the minimal Minimal Error Rate within the different definitions of SC.

SC is added into Equation 2 to describe the individual information of each sentence intensively. Equation 4 shows the new score method of a path.

$$S_n = w_1 sc_1 s_1 + w_2 sc_2 s_2 + \cdots + w_n sc_n s_n \quad (4)$$

sc_i denotes the SC of a component in the ith layer.

3 Experimental Results

3.1 Score of Components

We build a practical system CUP (Chinese Understanding Platform) based on MSM with three layers – Word-Seg, POS Tagging and Parsing. Each component not only provides the n-best analysis result, but also the score of each result.

In the Word-Seg component, we use the unigram model (Liu et al., 1998) to value different results of Word-Seg. So the score of a result is:

$$Score_{Word-Seg} = -\log P(W) = -\sum \log P(w_i)$$

w_i denotes the ith word in the Word-Seg result of a sentence.

In the POS Tagging component the classical Markov Model (Manning and Schütze, 1999) is used to select the n-best POS results of each Word-Seg result. So the score of a result is:

$$\begin{aligned} Score_{POS} &= -\log P(POS|W) \\ &= -\log \frac{P(W|POS)P(POS)}{P(W)} \\ &= -\sum \log P(w_i|t_i) - \sum \log P(t_i|t_{i-1}) \\ &\quad + \log P(W) \end{aligned}$$

t_i denotes the POS of the ith word in a Word-Seg result of a sentence.

In the Parsing component, we use a Chinese Dependency Parser System developed by HIT-IRLab[1]. The score of a result is:

$$\begin{aligned} Score_{Parsing} &= -\log P(DP|W, POS) \\ &= -\log \frac{P(W, POS, DP)}{P(W, POS)} \\ &= -\sum \log P(l_{ij}) \\ &\quad + \log P(W, POS) \end{aligned}$$

l_{ij} denotes a link between the ith and jth word in a Word-Seg and POS Tagging result of a sentence.

Table 1 gives the one and five-best results of each component with a correct input. The test data comes from Beijing Univ. and Fujitsu Chinese corpus (Huiming et al., 2000). The F-Score is used to value the performance of the Word-Seg, Precision to POS Tagging and the correct rate of links to Parsing.

Table 1: The five-best results of each component

	1-best	5-best
Word-Seg	87.83%	94.45%
POS Tag	85.34%	93.28%
Parsing	80.25%	82.13%

3.2 Self Confidence Selection

In order to select a better SC, we test all kinds of definition form to calculate their Minimal Error Rate. For example $B-A$, $\frac{B}{A}$ and so on. A and B denote the first and the second score of a component respectively. Table 2 shows the relationship between definition forms of SC and their Minimal Error Rate. Here, we experimented with the first and the second Word-Seg results of more than 7100 Chinese sentences.

3.3 F-Score of Word-Seg

The result of Word-Seg is used to test our system's performance, which means that the $ObjFun(*)$ returns the F-Score of Word-Seg.

There are 1,500 sentences as training data and 500 sentences as test data. Among these data about 10% sentences have ambiguities and the others come from Beijing Univ. and Fujitsu Chinese corpus (Huiming et al., 2000). In CUP the five-best results of each component are selected. Table 3 lists the F-Score of Word-Seg. They use Only Word-Seg (OWS), Whole Layer Weight (WLW), SC (SC) and FeedBack mechanism (FB) separately. Using the feedback mechanism means that the last analysis result of a sentence is decided by the Parsing. We select the result which has the lowest score of Parsing. Table 4 shows the weight distributions in WLW and SC weighting methods.

3.4 The Efficiency of CUP

The efficiency test of CUP was done with 7112 sentences with 20 Chinese characters averagely. It costs 58.97 seconds on a PC with PIV 2.0 CPU and 512M memory. The average cost of a sentence is 0.0083 second.

4 Discussions

According to Table 1, we can see that the performance of each component improved with the increasing of the number of results. But at the same time, the processing time must increase. So we should balance the efficiency and effectiveness with an appropriate number of results. Thus, it's more possible for CUP to find out the correct analysis than the original cascade mechanism if we can invent an appropriate method.

We define SC as $B-A$ which gets the minimal Minimal Error Rate with the analysis of the

Table 2: SC and Minimal Error Rate

Definition Form of SC	**Minimal Error Rate**
$\frac{1}{A}-\frac{1}{B}$	23.85%
$B-A$	21.07%
$\frac{B}{A}$	23.98%
$\frac{B}{A}-\frac{A}{B}$	23.98%
$\frac{B-A}{\text{length of a sentence}}$	24.12%
$\frac{B-A}{\text{length of a sentence}+100}$	23.71%

Table 3: F-Score of Word-Seg

	OWS	WLW	SC	FB
F-Score	86.99%	85.80%	88.13%	80.72%

Table 4: Layer Weight

	1-layer	2-layer	3-layer
In WLW	0.84	0.12	0.04
In SC	0.44	0.40	0.16

[1] The Parser has not been published still.

Table 2. Take the case of Word Segmentation:

$$B - A = \sum_i \log P(w_i^A) - \sum_j \log P(w_j^B)$$

It's just the difference between logarithms of different word results' probability of the first and the second result of Word Segmentation.

Table 3 shows that MSM using SC gets a better performance than other methods. For a Chinese sentence "桌子下放着几坛酒。". (There are some drinks under the table). The CUP gets the correct analysis – "桌子/n 下/nd 放/v 着/u 几/m 坛/q 酒/n 。/w". But the cascade and feedback mechanism's result is "桌子/n 下放/v 着/u 几/m 坛/q 酒/n 。/w".

The cascade mechanism uses the Only Word-Seg result. In this method $P(\text{下放})$ is more than $P(\text{下}) * P(\text{放})$. At the same time, the wrong analysis is a grammatical sentence and is accepted by Parsing. These create that these two mechanisms cannot get the correct result. But the MSM synthesizes all the information of Word-Seg, POS Tagging and Parsing. Finally it gets the correct analysis result.

Now, CUP integrates three layers and its efficiency is high enough for practical applications.

5 Conclusions and Future Work

A new Chinese NLU architecture based on Multilayer Search Mechanism (MSM) integrates almost all of NLU components into a uniform multilayer treelike platform and quantifies these components to use the search algorithm to find out the optimal result. Thus any component can be added into MSM conveniently. They only need to accept an input and give several outputs with scores. By experiments we can see that a practical system – CUP based on MSM improves the performance of Word-Seg to a certain extent. And its efficiency is high enough for most practical applications.

The cascade and the feedback mechanism are *JUST* the special cases of MSM. If greedy algorithm is used at each layer to expand the result with the lowest score, MSM becomes the cascade mechanism. If the weight of each layer except the feedback point is set 0, the MSM becomes the feedback mechanism.

In the future we are going to add the Phrase Analysis, WSD (Word Sense Disambiguation) and Semantic Analysis components into CUP, because it is impossible to analyze some sentences correctly without semantic understanding and the Phrase Analysis helps to enhance the performance of Parsing. At last, CUP becomes a whole Chinese NLU platform with Word-Seg, POS Tagging, Phrase Analysis, Parsing, WSD and Semantic Analysis, six components from lower layers to higher layers. Under the framework of MSM, it becomes very easy to add these components.

With the increasing of layers the handle speed must decrease. So some heuristic search algorithms will be used to improve the speed of searching while enhancing the speed of each component. Under the MSM framework, we can do these easily.

The performance of each component should be improved in the future. At least, it is impossible for MSM to find out the correct analysis result if there is a component which cannot give a correct result within n-best results with a correct input. In addition, we are going to evaluate the performance of each component not just Word-Seg only.

6 Acknowledgements

We thank Liqi Gao and Zhuoran Wang provide the Word Segmentation tool, Wei He provide the POS Tagging tool and Jinshan Ma provide the Parser tool for us. We acknowledge Dekang Lin for his valuable comments on the earlier versions of this paper. This work was supported by NSFC 60203020.

References

Shan Gao, Yan Zhang, Bo Xu, ChengQing Zong, ZhaoBing Han, and RangShen Zhang. 2001. The research on integrated chinese words segmentation and labeling based on trigram statistic model. In *Proceedings of IJCL-2001*, Tai Yuan, Shan Xi, China.

Duan Huiming, Song Jing, Xu Guowei, Hu Guoxin, and Yu Shiwen. 2000. The development of a large-scale tagged chinese corpus and its applications. *Applied Linguistics*, (2):72–77.

Ting Liu, Yan Wu, and Kaizhu Wang. 1998. The problem and algorithm of maximal probability word segmentation. *Journal of Harbin Institute of Technology*, 30(6):37–41.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, Massachusetts.

Melanie Mitchell. 1996. *An Introduction to Genetic Algorithms*. The MIT Press, Cambridge, Massachusetts.

Masaaki Nagata. 1994. A stochastic japanese morphological analyzer using a forward-dp backward-A* n-best search algorithm. In *Proceedings of the 15th International Conference on Computational Linguistics*, pages 201–207.

Stuart Russell and Peter Norvig. 1995. *Artificial Intelligence: A Modern Approach*. Prentice Hall Series in Artificial Intelligence, Englewood Cliffs, NJ, USA.

Matthew Wall. 1996. GAlib: A C++ Library of Genetic Algorithms components. http://lancet.mit.edu/ga/.

Andi Wu and Zixin Jiang. 1998. Word segmentation in sentence analysis. In *Proceedings of the 1998 International Conference on Chinese Information Processing*, pages 169–180, Beijing, China.