DJ Journal of Engineering and Applied Mathematics, Vol. 1(1) 2015, pp. 23-29

RESEARCH ARTICLE

Parsing Techniques using Paninian Framework on Nepali Language

[*]Archit Yajnik[1], Dhanapati Sharma[1]
[1]Department of Mathematics, Sikkim Manipal University, India.

Received-19 October 2015, Revised-15 November 2015, Accepted-20 November 2015, Published-27 November 2015

ABSTRACT

This paper presents three parsing techniques for establishing karaka-vibhakti relations using Paninian framework for nepali language. It uses the concept of karaka relation between verbs and nouns in the sentence. The constraints problem reduces to bipartite graph matching problem because of the nature of constraints and bipartite graph reduces to hopfield network. Upon employing these three techniques viz. constraint based , bipartite graph matching problem and hopfield networks using the Paninial framework to the sentences of nepali language an elegant account of the relation between surface form (vibhakti) and semantic (karaka) roles is obtained. All the three techniques are demonstrated by forming parsing structure in various nepali sentences.

Keywords: Parsing, Paninian framework, Karaka-Vibhakti, Bipartite matching problem, Hopfield network.

1. INTRODUCTION

Parsing is a process by which an input sentence is analysed and assigned a suitable structure [1-3]. Parsing process makes use of two components: a parser which is the procedural component (a computer program), and grammar which is declarative [4-8].

Various techniques are employed for parsing in different natural languages like statistical dependency, graph based, transition based etc [9-15]. In this paper, we describe such a formalism, called the Paninian framework, that has been successfully applied to hindi language and the same is applicable to the nepali language also. It uses the notion of karaka relations between verbs and nouns in a sentence. From the given nepali sentence after the word groups are formed, karaka charts for the verb groups are created and each of the noun or verb group is tested against the karaka restrictions in each karaka chart. The notion of karaka relations is central to the Paninian model. As part of this framework, a mapping is specified between karaka relations and vibhakti (which covers collectively case endings, post-positional markers, etc.).

Consider following sentences:

S.1 नानीले हाथले केरा खान्छ |

Naani -Le haathle Kera khancha
The child with hand banana eats
(The child eats banana with his hand.)

S. 2 रामले विहान फोन गरयो |

Ram- Le Vihan phone garayo

Ram morning phone did
(Ram used the phone at morning)

S. 3 रामले खेत जोत्यो |

Rama -Le khet jotyo
Ram the field ploughed
(Ram ploughed the field)

The default karaka chart for three of the karakas is given in figure 1. This explains the vibhaktis in sentences S.1 to S.3. In S.1, ‘Naani’ (The Child) and ‘Kera’ (Banana) and ‘Haath’ (Hand) are Karta, Karma and optional karana respectively because of the presence of Le, ∅ and Le respectively. Moreover sentence never begins with an optional karana. Note that ‘Naani’ and ‘Haath’ are followed by Le postposition whereas ‘Kera’ is followed by ∅ or empty postposition. Similarly in S.2 and S.3

*Corresponding author. Tel.: +919635319656
Email address: archit.yajnik@gmail.com (A.Yajnik)
Double blind peer review under responsibility of DJ Publications
http://dx.doi.org/10.18831/djmaths.org/2015011004

'Ram' is Karta since it is followed by Le postposition whereas 'phone' and 'field' are Karma due to the presence of ∅ postposition.

KARAKA	VIBHAKTI	PRESENCE
Karta	Le or ∅	Mandatory
Karma	laYi or ∅	Mandatory
Karana	Le	Optional

Figure 1.A Default Karaka Chart

Now consider the following two sentences.

S. 4 शिकारीले भाग्दैगरेको सिंहलाई देख्यो |

Shikari -Le Bhagdaigareko sinh -LaYi dekhyo
Hunter running lion saw
(Hunter saw the lion while running)

S. 5 रामले फल खाएर मोहनलाई बोलाउंच |

Ram -Le pahila khaer Mohan -LaYi bolaunch
Ram fruit ate Mohan called
(Ram ate fruit and called Mohan)

2. SOLUTION USING CONSTRAINT GRAPH

As mentioned in [1], [2], the role of the core parser is to determine karaka relations and senses of words from the given local word clusters in a sentence.

For a given sentence after the word clusters have been formed, karaka charts for the verb clusters are created and each of the noun clusters is tested against the karaka restrictions in each karaka chart. When testing a noun group against a karaka restriction of a verb group, vibhakti information is checked, and if found satisfactory, the noun cluster becomes a member for the karaka of the verb group. The above can be shown in the form of a constraint graph (1). The abbreviation for karaka is shown in table 1.

Table 1.Abbreviation for karaka

Karaka	Abbreviation
Karta	K1
Karma	K2
Karana	K3

Following the rules indicated for constraint graph in [2] and considering the direction of the age from right to left, the constraint graph and its solution graph can be obtained. The constraint graph for S.1 is shown in figure 2.

A parse is a subgraph of the constraint graph. Two solution graphs are obtained from the subgraphs of figure 2 as shown in figure 3 and figure 4.

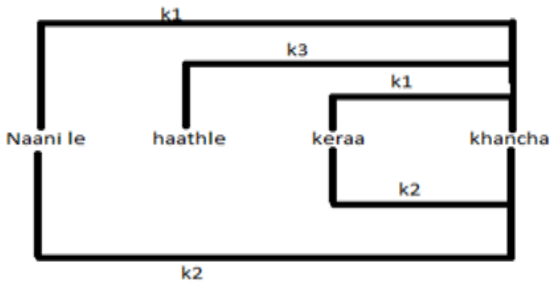

Figure 2.Constraint graph for S.1

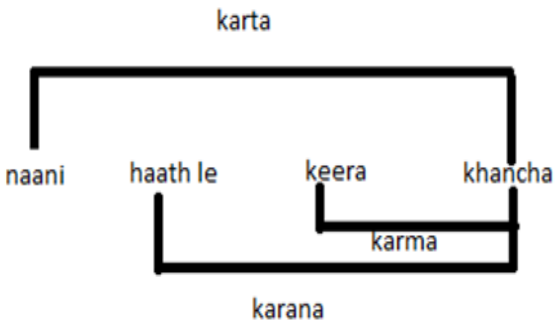

Figure 3.Solution graph (corresponding to "child eats banana")

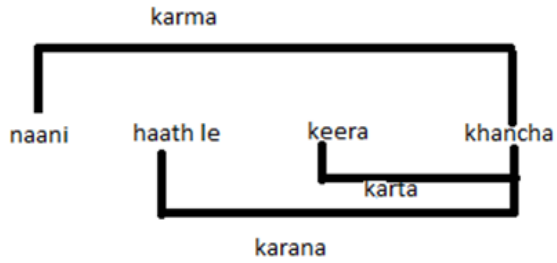

Figure 4.Solution graph (corresponding to the meaning "banana eats child")

Solution shown in figure 4 is incorrect because it violates the rules of default Karaka chart depicted in figure 1. Due to the presence of –Le before 'Naani' the noun 'Naani' must be Karta and cannot be karma.

3. SOLUTION USING BIPARTITE MATCHING GRAPH

Illustration 1: Consider the sentence S. 1

Naanile	haathle	keraa	khaancha
a	b	c	A

(Assuming that the optional karana karaka is mandatory) we have the bipartite graph

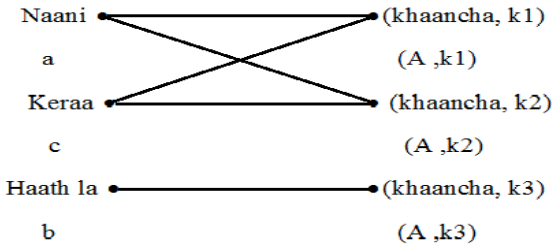

Figure 5. Bipartite graph for constraint graph of figure 2

A matching K of a bipartite graph G = (S, D, E) is a subset of edges with the property that no edges of K share the same node. The matching problem is to find a maximal matching of G that is matching with the largest number of edges. A maximal matching is called a complete matching if every node in S and D has an edge [2]. There are two maximal complete matching of the graph of the above sentence.

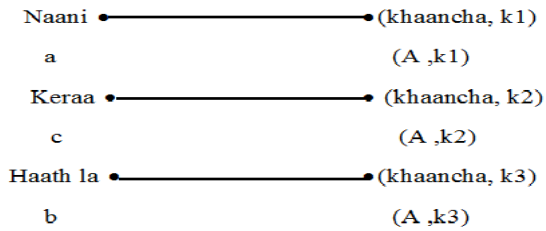

Figure 6.Solution graph corresponds to parse in figure 3

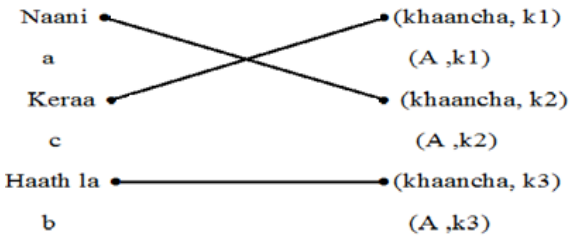

Figure 7.Solution graph corresponds to parse in figure 4

Illustration 2: Consider the sentence S.4,

Shikaarii bhaagdai garako sheya dhekyo

a A c B

Bipartite graph of the above sentence is given in figure 8.

Two maximal complete matchings of the graph (figure 8) of above sentence are depicted in figure 9 and figure 10.

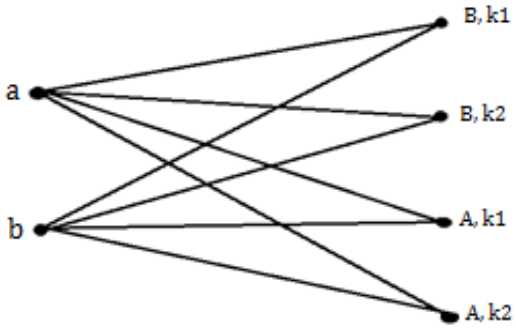

Figure 8.Bipartite graph for constraint graph

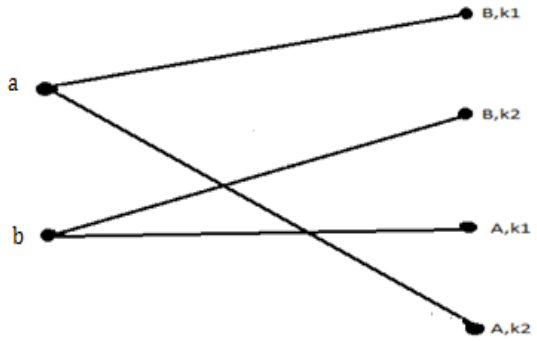

Figure 9.Bipartite Solution graph corresponds to parse

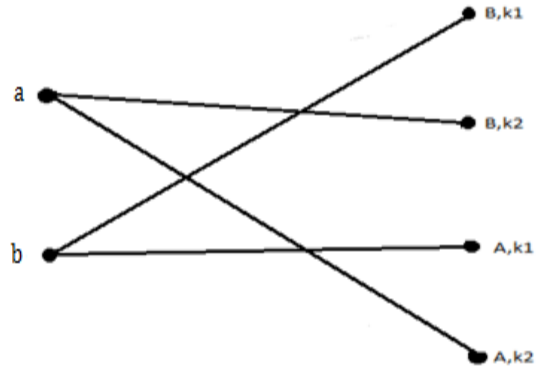

Figure 10.Solution graph corresponds to parse

4. SOLUTION USING HOPFIELD NETWORK LEARNING METHOD FROM BIPARTITE GRAPH

The constraint based problem can be converted into the hopfield network as shown in [4]. Equations (4.1) and (4.2) shows the details. In this representation, only N neurons are used for the $N - vertex$ bipartite sub-graph problem.

Neuron i $(i = 1, \dots\dots, N)$ (4.1)

expresses the $i - vertex$, and the output

$(y_i \sim 1 \; or \; y_i \sim 0)$ (4.2)

of neuron i expresses that the $i - vertex$ is

partitioned into the subset $V_1 \, or \, V_2$, respectively [4]. Energy function is defined as

$$e = -\frac{1}{2}\sum_{i-1}^{N}\sum_{j-i}^{N} Z_{ij}\, y_i y_j - \sum_{i-1}^{N} h_i y_i + c \quad (4.3)$$

where c is constant, the weight from the j^{th} neuron to the i^{th} neuron, and the threshold are given by

$$Z_{ij} = -2A\left(1 - \delta_{ij}\right)d_{ij} \quad (4.4)$$

$$h_i = A\sum_{j-1}^{n} d_{ij}(1 - \delta_{ij}) \quad (4.5)$$

In (4) and (5), the notation δ_{ij} is 1 if $i = j, 0$ otherwise. The equations of motion is shown in (4.6)

$$\frac{dx_{\ i}}{dt} = \sum_{j}^{n} Z_{ij}\, y_j + h_i \quad (4.6)$$

Here the weights are symmetric i.e. $Z_{ij} = Z_{ji}$. The network converges to an infimum of energy function. Furthermore, if there are no self-connections ($Z_{ii} = 0$), by applying learning rule to the bipartite sub-graph problem, we can obtain the changes of the weights and the thresholds of bipartite sub-graph problem from the partial differential of the energy function equation (4.3) to weights and the thresholds.

$$\Delta z_{ij} = p\,\frac{\partial e}{\partial w_{ij}} = -p\, y_i y_j \quad (4.7)$$

$(For\ i, j = 1, \ldots\ldots, N \ and \ i \neq j)$

$$\Delta h_i = q\,\frac{\partial e}{\partial h_i} = -q\, y_{\ i} \quad (4.8)$$

$(For\ i\ =\ 1, \ldots\ldots\ldots, N)$

where, p and q are small positive constants. Now the following procedure gives solution of hopfield network learning method for bipartite sub-graph problem.

4.1. Working procedure

The following procedure describes the proposed algorithm for bipartite sub-graph problem of $N - vertex$ graph to be solved with hopfield network. Note that there are two kinds of conditions for the end of the learning. One has a very clear condition, for example, the $N - queen$ problem in which the energy is zero if the solution is optimum. Another one do not have a clear condition. For example, the travelling salesman problem and bipartite sub-graph problem in which the energy is not zero even when the solution is optimal. For the latter case, we have to set a maximum number of the learning in advance. Learning stops if the maximum number of learning is performed. If the learn limit is supposed to be the maximum number of learning times for the system termination condition, we have set learning time 0 and learn limit 20. "A" in equation (4.4) and (4.5) is set to 1.0. Constants p in (4.7), q in (4.8) is set to 0.01, temperature parameter T in (4.7) is set to 0.25.

The initial value of $x_i \quad for\ i = 1, \ldots\ldots, N$ are randomized to either $0\ or\ 1$. The updating procedure is performed on the hopfield network with original weights and thresholds until the network converge to be stable.

Recording the stable state, use equation (4.7) and (4.8) to compute the new weights and thresholds.

$$For\ i = 1, \ldots N,$$

Compute the $\Delta_{Z_{ij}}$using (7) $for\ j = 1, \ldots, N \ and \ j \neq i$;

Change the Z_{ij} for j = 1,…, N and $j \neq i.\ Z_{ij} = Z_{ij} + \Delta Z_{ij}$.

Compute the Δh_i using equation (4.8)

The updating procedure is taken on the hopfield network with the new weight and thresholds until the network reaches a stable state.

Illustration I: Consider S.1, the adjacency and weight matrix (using equation 4.4) of figure 5 given by

$$d = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

and weight matrix

$$w = \begin{bmatrix} -0 & -0 & -0 & -2 & -2 & -0 \\ -0 & -0 & -0 & -2 & -2 & -0 \\ -0 & -0 & -0 & -0 & -0 & -2 \\ -2 & -2 & -0 & -0 & -0 & -0 \\ -2 & -2 & -0 & -0 & -0 & -0 \\ -0 & -0 & -2 & -0 & -0 & -0 \end{bmatrix}$$

respectively. The Hopfield network is depicted in figure 11.

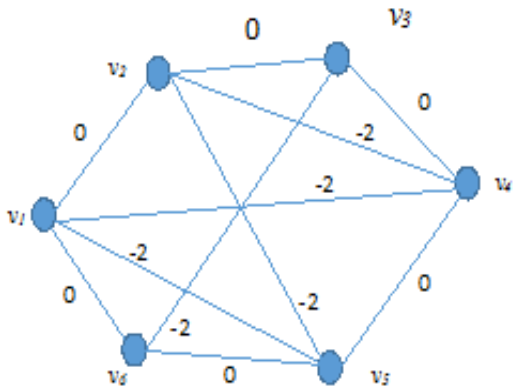

Figure 11.Hopfield network with initial weights

Using equation (4.5), threshold vector $h = [2,2,1,2,2,1]^T$ .Initial energy is turned out to be -4 using equation (4.2). Vertices and their meanings for S.1 are shown in table 2.

Table 2.Vertices and their meanings for S.1

Vertex	Meaning
v_1	Naanile
v_2	Kera
v_3	Haathle
v_4	Khancha, K_1 (A, K_1)
v_5	Khancha, K_2 (A, K_2)
v_6	Khancha, K_3 (A, K_3)

Taking initial vector $v = [0,0,0,1,0,0]^T$. In order to get steady state of i^{ih} $(for\ i = 1,2,3,4,5,6)$ component of v, the linear combination (say s) of vector v and i^{ih} row of w is compared with the i^{th} component of h. If $s \geq hi$ then replace v_i by 1 and 0 otherwise. Updation of weights w and threshold h continue until the error between the two consecutive $v's$ reaches the desired accuracy. After 9th iteration, the steady state solution $(say\ y1)$ is attained as shown in equation (4.9).

$$y_1 = [1,1,0,1,1,0]^T \qquad (4.9)$$

Similarly, taking another initial vector $v = [0,0,0,0,0,1]^T$, another steady state solution $(say\ y2)$ is obtained as shown in equation (4.10).

$$y_2 = [0,0,1,0,0,1]^T \qquad (4.10)$$

Here, the first three components of yi ($i = 1\ and\ 2$) $say\ v_1, v_2, v_3 \in V_1$ and last three components say $v_4, v_5, v_6 \in V_2$. From figure 5 in a bipartite graph, there is no edge among the elements of $Vi, for\ i = 1,2$.

Non zero entries in y_1 equation (4.9) assumes that there is an edge among $v_1, v_2, v_4 or\ v_5$. Since $v_1, v_2 \in V_1$ and $v_4, v_5 \in V_2$, there can't be any age between v_1 & v_2 and $v_4 or\ v_5$. Therefore, v_1 can be connected $with\ v_4\ and\ v_5$. But it violates the rule 3 of constraints graph because there are two incoming edges into the source group v_1. Hence v_1 can either be connected with $v_4 or\ v_5$

Case: I

Let us take v_1 is connected with v_4 then with the same argument v_2 is connected with v_5 .Output y_2 is unambiguous, there is an age between $v_3\ and\ v_6$. In this situation, the solution graph mentioned in figure 6 is obtained.

Case: II

Let us take v_1 is connected with v_5 then with the same argument v_2 is connected with v_4. Again the output y_2 is unambiguous, there is an age between $v_3\ and\ v_6$. In this situation, the solution graph mentioned in figure 7 is obtained.

Case II does fails since it violates the conditions of default karaka chart shown in figure 1.

Illustration II:

Considering S.5, the adjacency

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

with 9 vertices. As shown shown in illustration I, the two sets $V_1\ and\ V_2 with\ |V_1| = 3\ and\ |V_2| = 6$ forms a bipartite graph. Applying the same procedure, we obtain the steady state solutions as shown in equation (4.11) and (4.12).

$$y_1 = [1,1,0,1,1,0,1,1,0]^T \qquad (4.11)$$

$$y_2 = [0,0,1,0,0,1,0,0,1]^T \qquad (4.12)$$

The corresponding solution subgraph is shown in figure 12.

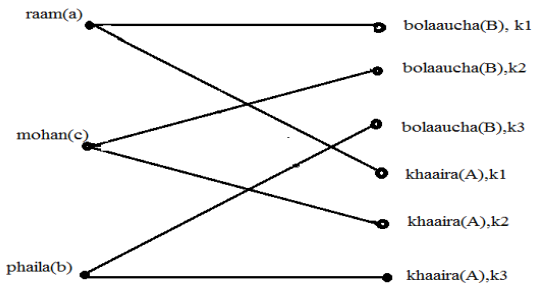

Figure 12.Solution graph of S.5

5. CONCLUSION

In summary, this paper makes several contributions:

- It shows that the Paninian framework applied to nepali language gives an elegant role of the relation between vibhakti and karaka roles. A default karaka chart is demonstrated in section 1.
- It shows in section 2 and 3 how a constraint based parser can be built using the framework. The constraints problem reduces to bipartite graph matching problem because of the nature of constraints. Efficient solutions are known for these problems.
- The method of obtaining a solution subgraph by converting a Bipartite graph to a hopfield network is demonstrated in section 4. Solution is obtained by implementing this method.

REFERENCES

[1] Akshar Bharti and Rajeev Sangal, Parsing Free Word Order Languages in the Paninian Framework, Proceedings of the 31st Annual Meeting on Association for Computational Linguistics (ACL), USA, 1993, pp. 105-111, http://dx.doi.org/10.3115/981574.981589

[2] R.Tsarfaty and K.Sima'an, Relational - Realizational Parsing, Proceeding of the 22nd International Conference on Computational Linguistics Coling, Manchester, UK, 2008, pp. 889-896.

[3] Akshar Bharti, Vineet Chaitanya and Rajeev Sangal, Natural Language Processing: A Paninian Perspective, Prentice-Hall of India, New Delhi, 1995.

[4] Rong Long Wang, Zheng Tang and Qi Ping Cao, A Hopfield Network Learning Method for Bipartite Subgraph Problem, IEEE Transactions on Neural Networks, Vol. 15, No. 6, 2004, pp.1458-1465, http://dx.doi.org/10.1109/TNN.2004.836234.

[5] James Allen, Natural Language Understanding, Addison-Wesley Publication, Boston, Massachusetts, 1995.

[6] Jan Van Bakel, Automatic Semantic Interpretation: A Computer Model of Understanding Natural Language, Journal of Computational Linguistics - Special Issue on Machine Translation, Vol. 11, No. 2-3, 1985.

[7] Russell Suereth, Developing Natural Language Interfaces: Processing Human Conversations, McGraw-Hill, Inc. New York, NY, USA, 1996.

[8] Frederick Jelinek, Statistical Methods for Speech Recognition, Pearson Education, 2001.

[9] Christopher D Manning and Hinrich Schuetze, Foundation of Statistical Natural Language Processing, Computational Linguistic, Vol. 26, No. 2, USA, 2000.

[10] Ashwini Vaidya, Samar Husain, Prashanth Mannem and Dipti M Sharma, A karaka-Based Dependency Annotation Scheme for English. In Proceedings of the CICLing, Mexico City, Mexico, 2009.

[11] Chaitanya Vempaty, Viswanatha Naidu, Samar Husain, Ravi Kiran, Lakshmi Bai, Dipti M Sharma and Rajeev Sangal, Issues in Analyzing Telugu Sentences Towards Building a Telugu Treebank. In Proceedings of CICLing, Iasi, Romania, 2010, pp. 50-59, http://dx.doi.org/10.1007/978-3-642-12116-6_5

[12] Sriram Venkatapathy, Preeti Agarwal and Aravind K. Joshi, Relative Compositionality of Noun+Verb Multiword Expressions in Hindi, In

Proceedings of ICON, Kanpur, India, 2015.

[13] Manindra K Verma, Complex Predicates in South Asian Languages. Manohar Publications, New Delhi, 1993.

[14] Hiroyasu Yamada and Yuji Matsumoto, Statistical Dependency Analysis with Support Vector Machines, In Proceedings of the 8th IWPT, Nancy, France, 2003, pp. 195-206.

[15] Yue Zhang and Stephen Clark, A Tale of Two Parsers: Investigating and Combining Graph- Based and Transition-Based Dependency Parsing. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Hawaii, USA, 2008, pp.562-571.