

Augmenting the Automated Extracted Tree Adjoining Grammars by Semantic Representation

Heshaam FAILI

Department of ECE,
University of Tehran, Tehran, Iran
hfaili@ut.ac.ir

Ali BASIRAT

Department of Computer Engineering,
Islamic Azad University, Science and Research
Branch of Tehran, Iran
a.basirat@srbiau.ac.ir

Abstract--MICA [1] is a fast and accurate dependency parser for English that uses an automatically LTAG derived from Penn Treebank (PTB) using the Chen's approach [7]. However, there is no semantic representation related to its grammar. On the other hand, XTAG [20] grammar is a hand crafted LTAG that its elementary trees were enriched with the semantic representation by experts. The linguistic knowledge embedded in the XTAG grammar caused it to being used in wide variety of natural language applications. However, the current XTAG parser is not as fast and accurate as well as the MICA parser.

Generating an XTAG derivation tree from a MICA dependency structure could make a bridge between these two notions and gets the benefits of both models. Also, by having this conversion, the applications that use the XTAG parser, may get the helps from MICA parser too. In addition, it can enrich the MICA's grammar by semantic representation of XTAG grammar.

In this paper, an unsupervised sequence tagger that maps any sequence of MICA elementary trees onto an XTAG elementary trees sequence is presented. The proposed sequence tagger is based on a Hidden Markov Model (HMM) proceeded by an EM-based algorithm for setting its initial parameters values. The trained model is tested on a part of PTB and about 82% accuracy for the detected sequences is achieved.

Keywords—Semantic representation, HMM initializing, Grammar mapping, Automated extracted tree adjoining grammar (TAG), XTAG derivation tree,

1. Introduction

Tree Adjoining Grammar (TAG) formalism introduced by Joshi [11] is a mildly context sensitive grammar. The fundamental representing structure of TAG's is trees, namely elementary tree, rather than symbols as in Context Free Grammars. Lexicalized Tree Adjoining Grammars (LTAG) is a kind of TAG that assigns a finite set of lexicalized elementary trees to each word of language.

LTAG's could be crafted manually or extracted automatically. The manually crafted grammars are developed by some linguistic experts while the automatically extracted LTAG's are extracted from some

corpus based statistical approaches. In general, the manually crafting grammar processes are very costly and time consuming. Such difficulties inspire some researchers to develop some methods to extract LTAG's from a tree bank, automatically [19]. In [13], [7], [19], [16] and [14] some statistical models are proposed that extract LTAG's grammar from a tree bank. These kinds of grammars also suffer from the huge size of elementary tree set and sparse data problem [7]. Moreover, there is no semantic representation for these grammars. In fact enriching these huge size grammars by semantic representation needs a large manually effort. However, unlike to manually crafted LTAG's, due to statistical nature of automatically extracted LTAG's, some efficient statistical parsers such as MICA [1] and Statistical LTAG parser [16] are designed to work on them.

Mapping the elementary trees of these two kinds of grammars can provide a solution to combine the advantages of them. The usefulness of this mapping could be considered by three different ways. First, mapping elementary trees of automatically extracted grammars onto elementary trees of a hand crafted grammar can enrich the automatically extracted grammar with the semantic representation of hand written elementary trees. Second, this mapping can provide a way to use exist statistical parsers of automatically extracted grammar in applications that are based on an instance of manually crafted grammars. Third, this mapping can provide a way for evaluating the existence gap in the both kinds of grammars.

Here, we are going to introduce a solution for mapping the Chen's [7] automatically extracted LTAG used in MICA parser¹ onto the well known manually crafted English XTAG [20] grammar.

MICA (stand for Marseille-INRIA-Columbia-AT&T) is a dependency parser that returns the deep dependency representations have been created since 2009 [1]. MICA could be used as a supertagger too. In supertagging, MICA uses the set of supertags automatically extracted from Penn Treebank using the Chen's approach [7] as mentioned. This is not a TAG grammar but instead it is a limited TAG called TIG (Tree Insertion Grammar) [15]. This automatically extracted TIG then is converted into a specialized PCFG

¹ For simplicity, in the rest of the paper, we name the automatically extracted grammar, which is used by MICA parser by MICA's grammar and the elementary trees of such grammar as MICA's elementary tree.

that can model more finely some lexico-syntactic phenomena. An Earley-like parse based on PCFG generated by SYNTAX [4] comprises the MICA parser.

XTAG grammar is a well known and large scale manually crafted LTAG for English that has been hand-crafted at university of Pennsylvania since 1990. XTAG contains about 1,000 elementary trees. These trees provide good syntactic as well as semantic constraints over the words due to their manually creating. The linguistic knowledge embedded in XTAG grammar caused it widely has been used in many TAG based applications such as semantic interpretation and machine translation systems [20], [10].

Despite all advantages of XTAG grammar, there is no related efficient parser yet. The current available XTAG parser² uses the head-corner parsing method mentioned by Van Noord [17] that runs in $O(n^6)$ time complexity. A lot of ambiguity in output of this parser and the lack of effective disambiguation method caused many problems in the applications that use the XTAG parser [10]. Moreover, the low speed of this parser is another problem when using it in practical applications such as transfer-based MT's [9].

The low speed beside the ambiguities of XTAG parser encouraged some researchers to improve the parsing time and develop a disambiguation process. In [3], an idea based on statistical partial parsers which alleviate the TAG formalism in time complexity and ambiguity is proposed. They extend the notion of "tag" from a *part of speech* to a tag that represents rich syntactic information, and named it supertag. Each supertag can be considered as an element in TAG formalism and the whole method was named Supertagging that is almost parsing [3]. According to proposed idea, supertagging: almost parsing, the only remaining step after supertagging, is to combine the obtained supertags. This idea leads some researchers to create some efficient supertagging methods. In [6] the performance of supertagging by using some local techniques, which avoid parsing, was improved. In [2] also supertagging accuracy has been enhanced by training a Maximum Entropy model from contextual attribute of tagged word. Also, in [10] by embodying a supertagger within a hill climbing search strategy, the search space of XTAG parser was pruned dramatically. Some heuristics for detecting and correcting the erroneous supertags were proposed too.

In this paper, we have formulated the mentioned mapping as a sequence-tagging problem to give the best mapping solution depending on the local and non-local information of each elementary tree. Here an unsupervised sequence tagger based on Hidden Markov Model has been presented that produces an XTAG elementary tree sequence given an automatically extracted elementary tree sequence.

The layout of this paper is as follow. In section 2, we present a brief history of previous works which have been done on this subject. Section 3 illustrates an HMM to map a sequence of MICA's elementary trees onto the XTAG elementary trees. To use HMM, we need to specify three matrices *emission*, *transition* and *prior*. Initializing these matrices was done by comparing MICA and XTAG parser

results using an EM procedure. Finally, the experimental results are shown in section 4.

2. Previous Works

Sparseness and the lack of semantic representation for automatically extracted grammars are two main obstacles in using them in the real world applications. These weaknesses could be compensated by mapping these grammars onto a hand written grammar. This mapping provide a way to discover the redundant automatically extracted elementary trees and to enrich automatically extracted grammars with available semantic representation of hand written grammars such as English XTAG grammar [7]. Moreover, it provides a way to evaluating the coverage of the automatically extracted grammar or vice versa [18].

The outline of Chen's method [7] is based on two transformations, *node local transformation* and *structural transformation*. In node local transformation the minor changes between XTAG and automatically extracted elementary tree formatting is alleviated. In structural transformation phase by running three syntactic transformations, it attempted to diminish the syntactically differences between automatically extracted elementary trees and the result of node local transformation phase. These syntactic transformations are as follow:

1. Delete vacuous verb movement if it exists.
2. Insert NP object to subject movement if it is passive.
3. Eliminate PP sub trees immediately dominating P co-anchors.

Then the nodes of transformed XTAG elementary trees are compared to each of automatically extracted elementary trees. Using these transformations, many of most frequently used elementary tree of XTAG grammar that are about 4% of overall tree frames and about 30% of tree frames anchored by verb, were mapped onto the elementary trees of automatically extracted grammar correctly. The defined goal of the Chen's procedure is to map as many tree frames in the extracted grammar onto the XTAG tree frames as possible.

Xia and Palmer [18] also use a similar rule-based method, which their rules are heuristically designed to calculate the overlap between two grammars. They define two type of matching named *t-match* and *c-match* to map an XTAG elementary tree onto an automatically extracted elementary tree. According to their work, an elementary tree from automatically extracted grammar matches to an elementary tree from XTAG grammar, if they satisfy one of *t-match* or *c-match* rules. They reported 82.1% accuracy in the matching procedure regarding *t-match* and *c-match* types. The defined mapping goal in [18] is to estimate the coverage of XTAG grammar on English Penn Treebank. They reports that the grammar can covers 97.2% of template tokens in Treebank.

3. Our Approach

Both of the previous approaches just use the similarities between isolated elementary trees and none of them use any contextual information of the elementary tree sequences in mapping. This caused none of [7] and [18]'s approaches provide a one-to-one mapping between XTAG and automatically extracted elementary trees. It means, each elementary tree of source grammar was mapped to more

² Updated version available at <http://www.cis.upenn.edu/~xtag>.

than one elementary tree's of target grammar. The many-to-many mapping property of both mentioned methods is an obstacle in using them in applications. In fact, using either of [7] and [18] approaches for mapping an automatically extracted elementary tree onto an XTAG elementary tree needs a disambiguation phase due to many-to-many mapping property.

Another problem arises when using the mentioned mapping procedures is their agreement in the anchor of mapped elementary trees. It means the lexical items assigned to an elementary tree of source grammar may not be same as the lexical items assigned to its corresponding elementary tree in target grammar regarding the mapping procedure.

Here, the mapping problem is defined formally as a sequence tagging in which the result is depend on the position of each source elementary tree in a sequence as well as other local and non-local information. For the sake of simplicity we will use P as the MICA parser, G as the MICA grammar, P' as the XTAG parser and G' as the XTAG grammar. Formal definition of problem is as follow:

Given a sequence of elementary trees $T = (t_1, \dots, t_n)$ $t_i \in G$ related to the parse of sentence $S = (w_1, \dots, w_n)$ done by P, the mapping method is defined as finding a function F that tags each member of T by an individual elementary tree $t'_i \in G'$ $i = 1 \dots n$ such that the likelihood of $T' = (t'_1, \dots, t'_n)$ given T and S, $(L(T'|T, S))$, be maximized.

This sequence classification problem can be modeled simply using a Hidden Markov Model. In the rest of this section an HMM that models the mentioned tagging problem is presented. The initialization and training phase of proposed model also are illustrated in details.

3.1. Problem Modeling Using HMM

To model a problem using HMM, we need to specify the HMM parameters separately. By considering the elementary trees t_j belong to G' as the states of the HMM and the sequences of elementary trees t_k belong to G as the observation sequences then the problem can be modeled based on HMM as below:

1. N (Number of states in the model): number of single anchor elementary trees t'_j belongs to G'^3 .
2. M (Number of distinct observation symbols in the model): number of elementary trees t_k belongs to G^4 .
3. State transition probabilities matrix ($A=[a_{i,j}]$ where $a_{i,j}=P(q_{t+1}=S_j|q_t=S_i)$): $a_{i,j}$ is the probability of appearance of t'_j after t'_i in a sequence of XTAG elementary trees. Simply, it is the conditional probability of appearing t'_j given t'_i as below:

$$P(t'_j|t'_i) \quad \forall t'_i, t'_j \in G' \quad (1)$$

4. Observation probabilities matrix ($B=[b_j(m)]$ where $b_j(m)=P(O_t=v_m/q_t=S_i)$): $b_{j,i}$ is the conditional

probability of observing t_j given an XTAG elementary tree t'_i as below:

$$P(t_j|t'_i) \quad \forall t_j \in G \text{ and } \forall t'_i \in G' \quad (2)$$

5. Initial state probabilities matrix ($\Pi = \pi_i$ where $\pi_i = P(q_1 = S_i)$): Probability of seeing t'_i in the beginning of a sequence of XTAG elementary trees.

Given an HMM λ and MICA elementary tree sequence $T = (t_1, \dots, t_n)$, the Viterbi algorithm can find the most likely sequence $T' = (t'_1, \dots, t'_n)$ of XTAG elementary trees such that the probability $P(T'|T, \lambda)$ be maximized. Actually, combination of Viterbi and MICA parser as an observation sequence generator could be a replacement for XTAG parser. Figure 1, illustrates how to use the MICA parser in combination with the Viterbi to be replaced with XTAG parser.

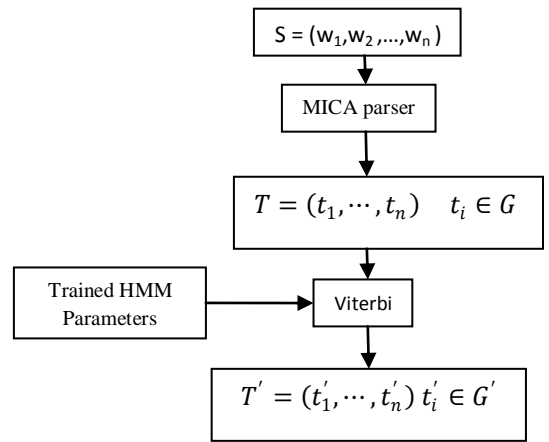


Figure. 1. Using MICA parser instead of XTAG parser

3.2. Training Data Set

Like the other statistical modeling tools HMM also needs a large-scale training corpora for both training and initialization. As we will say in this section, a part of these corpora is achieved by XTAG parser. The low speed and high ambiguity of XTAG parser was an important obstacle in this phase. In fact, the size of used corpora and the length of used sentences were limited just because of the mentioned weakness of XTAG parser. We have trained and also initialized the model by using 12000 sentences with maximum length 20 words of Penn Treebank. The average length of the sentences in the mentioned training data set (name it TRDB) is 12 words.

The whole sentences of TRDB were first parsed by both MICA and XTAG parsers. However, about half of this corpus was not parsed regarding XTAG parser. Let C and C' be two sets of elementary tree sequences obtained from parsing TRDB using MICA and XTAG parsers respectively. Each sequence of C or C' contains a set of elementary trees that are utilized in a derivation tree. The order of elementary trees in an elementary tree sequence is same as the order of their anchors in corresponding sentence.

Neither of MICA and XTAG parsers produces a unique derivation tree for a sentence. However, due to its probabilistic architecture, MICA has the ability to score its resulted derivation trees. Among the whole elementary tree

³ Number of single anchor XTAG elementary trees are about 490

⁴ The exact number of MICA elementary trees is 4726

sequences achieved from parsing a sentence, C contains the most probable derivation tree. But on the other hand, as mentioned before, XTAG does not provide any solution for ambiguity problem of parser results. In fact, related to each sentence of TRDB, there are multiple ambiguous derivation trees in C' . Due to the G 's limitation, the whole derivation trees which contain at least one multi-anchor elementary tree are eliminated from C' .

To summarize, given a sentence S_i with length n in TRDB, there is a sequence of MICA elementary trees T_i in C and a set of XTAG elementary tree sequences ξ_i as a subset of C' . Each member of ξ_i corresponds to a derivation tree achieved from parsing S_i by XTAG parser. Figure 2 illustrates the whole process.

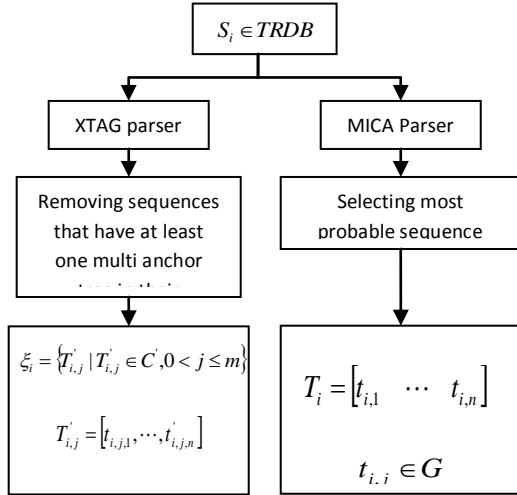


Figure. 2. Corresponding to each sentence in TRDB there is a unique MICA elementary trees sequence in C and a set of XTAG elementary tree sequences in C'

3.3. Initializing the HMM Parameters

The simplest and most intuitive way to estimate the mentioned HMM parameters is using Maximum Likelihood Estimation (MLE) method on the generated annotated corpora C and C' . However, the existence ambiguity in the C' would mislead the MLE method. To reduce the ambiguities size of C' , we have used a specialized kind of expectation maximization (EM) algorithm [8]. During the proposed algorithm we are going to keep the XTAG elementary tree sequences that are more compatible with their related MICA elementary tree sequences and pruning the C' .

Formally, we are going to estimate HMM parameters λ such that the probability

$$P(C', C | \lambda) = P(T'_1, \dots, T'_n, T_1, \dots, T_m | \lambda) \quad (3)$$

be maximum. Eq. (3) have been estimated by partitioning C' onto m partitions $\xi_i = \{T'_{i,j} | t'_{i,j} \in G', 0 < j \leq k\}$, each one contains all sequences in C' corresponding to a sentence in TRDB (and consequently corresponding to a sequence in C). So eq. (3) could be estimated as shown in formula (4).

$$P(C', C | \lambda) = P(\xi_1, \dots, \xi_m, T_1, \dots, T_m | \lambda) \quad (4)$$

According to eq. (3), EM algorithm can be applied by the following formulation. E-step is defined as the

estimation of the HMM parameters that can be done using MLE by some consideration. M-step is maximizing the value of probability eq. (3) using the latest λ .

Counting in MLE has been done by assigning a score to each sequences of C' according to the joint probability of accruing that sequence and its related sequence in C given λ . We have defined the score of sequence $T'_{i,j} \in \xi_i$ as the joint probability of observing that sequence and its related sequence $T_i \in C$ given the current values of HMM parameters (λ) normalized by the number of sequences in ξ_i . Due to the variation of HMM parameters in each iteration, the scores of each sequences in C' are varied during each iteration of EM algorithm. At begin, the initial score of each sequence in C' is considered to be same and they will be varied according to the variation of HMM parameters. Eq. (5) shows the mentioned definition.

$$\mathfrak{Z}(T'_{i,j}) = \begin{cases} 1 & \text{iter} = 0 \\ \frac{P(T'_{i,j}, T_i | \lambda)}{|\xi_i|} & \text{iter} \neq 0 \end{cases} \quad (5)$$

Given $T_i = (t_{i,1}, \dots, t_{i,n})$ in C there is a matrix of XTAG elementary trees related to ξ_i as shown in eq. (6).

$$\begin{bmatrix} T'_{i,1} \\ \vdots \\ T'_{m,1} \end{bmatrix} = \begin{bmatrix} t'_{i,1,1} & \dots & t'_{i,1,n} \\ \vdots & & \vdots \\ t'_{i,m,1} & \dots & t'_{i,m,n} \end{bmatrix} \quad (6)$$

The observation probabilities could be estimated by taking weighted-count from the set

$$\tau = \{(t_{i,k}, t'_{i,j,k}) | i \leq |C|, j \leq |\xi_i|, k \leq |T_i|\} \quad (7)$$

and normalizing them by the sum of all observed pair $(t_x, t'_{i,j,k}) \in \tau$ that share the same second XTAG elementary tree. State transition and initial state probabilities also have been estimated by taking weighted-count according to the score of each sequence from C and C' .

In M-step by trashing some sequences of C' that are not compatible with current achieved HMM parameters it tries to maximize the eq. (3). These are some sequences of C' that their score are less than the pre-defined threshold value θ , and are considered as redundant sequences. An intuitive suggestion for $\theta_{i,j}$ related to sequence j of ξ_i is presented by eq. (8).

$$\theta_{i,j} = \text{mean}(\overline{\mathfrak{Z}(\xi_i)}) \quad 1 \leq j \leq |\xi_i| \quad (8)$$

Figure 3 summarizes the main steps of initialization phase. In each iteration it estimates the HMM parameters using weighted-MLE. Trashing the redundant sequences from C' is according to the estimated threshold value for each XTAG sequences in C' . It iterates until no change in C' or it exceeded predefined maximum number of iterations.

During the EM-based initialization iterations, we can expect that the value of eq. (3) in each iteration increase. Figure 4 shows the logarithm of value of eq. (3) normalized by the number of remain derivation trees in C' in each iteration. Figure 5 also shows the variation of values of eq. (3) during proposed EM-based HMM initialization

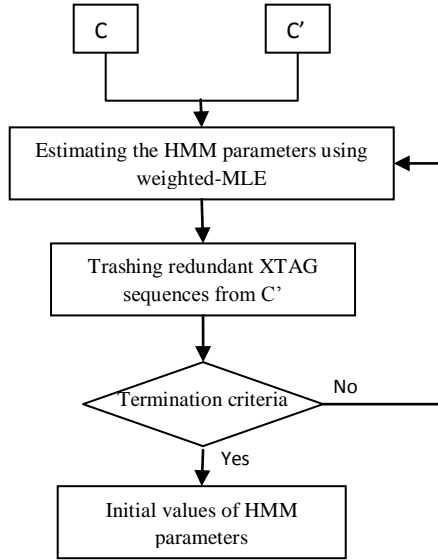


Figure 3. Initialization, estimating the initial values of HMM

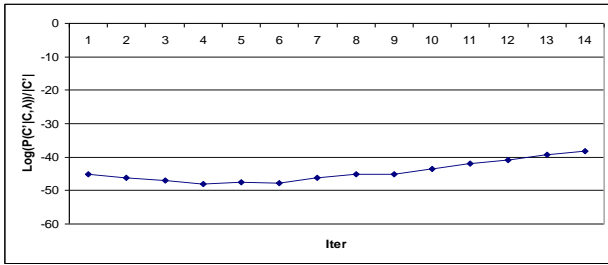


Figure 4. The variation of the logarithm of value of eq.(3) normalized by the number of remain derivation trees in C' with respect to the iterations of EM based initialization procedure.

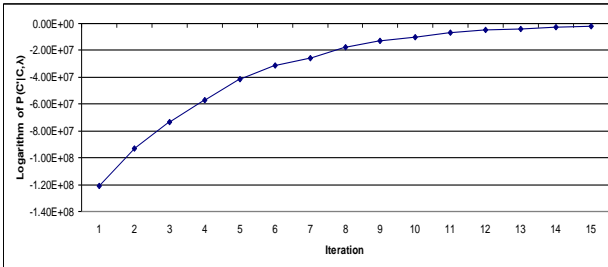


Figure 5 The variation of the value of eq.(3) with respect to different iterations of EM based initialization procedure.

3.4. Smoothing

Same to other corpus-based approaches, due to limitation of available corpus, sparseness problem is unavoidable. Among the three matrices A , B and Π , B is more sparse than the others are. Our experiments on TRDB are clarified that from the whole 2,325,192 (4,726*492) elements of matrix B , the mentioned initialization EM method could fill just 672 elements. Also, among the all of 4,726 elementary trees used by MICA, the EM method faced with only 129 elementary trees (2.7 percent of all MICA elementary trees). This shows a very huge amount of sparseness in B .

To overcome the sparseness problem in matrix B , a new smoothing method based on interpolation approach is proposed. Here, four kind of emission matrix, named B_{parser} , B_{stagger} , B_{dict} and B_{chen} , are developed and mixed by using interpolation method as shown in formula 9.

$$B = \lambda_1 B_{\text{paresr}} + \lambda_2 B_{\text{stagger}} + \lambda_3 B_{\text{dict}} + \lambda_4 B_{\text{chen}} \quad (9)$$

$$\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 1$$

B_{parser} is the observation probability matrix achieved by the initialization phase and smoothed using Laplace smoothing method with a few modifications. The Laplace smoothing method have been applied just on some elements of B that their related MICA elementary trees are not an unused⁵, instead of applying it on all of B elements. With this restriction, we guarantee that the un-used MICA elementary trees never being selected during mapping process. However, it may seem as a cause that the un-used MICA elementary trees remain un-used.

B_{stagger} is estimated by comparing the supertagging of TRDB using XTAG⁶ and MICA supertaggers. Both of MICA and XTAG supertaggers have the ability to estimate the probability of assigning an elementary tree (Supertag) to a word, hence scoring the supertagging result. Given any supertag sequence $ST = (st_1, \dots, st_n)$ related to sentence $S = (w_1, \dots, w_n)$ eq. (10) can estimate the probability of ST according to distributions of any supertag given a word.

$$P(ST|S) = \prod_{i=1}^n P(st_i|w_i) \quad (10)$$

As in B_{parser} , here we have provided two corpus of most probable XTAG and MICA supertags (elementary tree) sequences for each sentence S in TRDB, namely ζ' and ζ respectively, using their corresponding supertaggers (instead of parser). So, related to each word $w_k \in TRDB$ there is a pair (st, st') produced by MICA and XTAG supertaggers respectively. The corresponding conditional probability st and st' given w_k also are provided by MICA and XTAG supertaggers. For each corresponding pair (st, st') , the conditional probability $P(st|st')$ could be estimated using Bayesian rule as shown in eq. (11).

$$P(st|st') = \frac{P(st, st')}{P(st')} \quad (11)$$

Each of individual probabilities $P(st|w)$ or $P(st'|w)$ are provided by supertaggers. $P(w)$ also could be estimated by taking count from TRDB words normalized by the total number of words in TRDB.

As mentioned, XTAG is a hand-crafted English grammar based on LTAG model. The associations of each lexical word to their suitable elementary trees are stored in a hand-written dictionary. For simplicity we name this dictionary as syn_dic . In fact, each entry in syn_dic , indexed by lexical word w , contains all elementary trees which are related to w .

B_{dict} is computed same as the B_{stagger} but instead the words of TRDB it uses all of the words of the syn_dic dictionary. Considering n_k as the number of elementary trees assigned to word w_k , eq. (11) could be rewritten as follow.

$$P(t_j|t'_i) = \frac{\sum_k \frac{P(w_k)P(t_j|w_k)}{n_k}}{\sum_k \frac{P(w_k)}{n_k}} \quad (12)$$

B_{chen} is the emission matrix achieved by Chen's [7] proposed mapping procedure from XTAG elementary trees

⁵ Un-used elementary trees are some MICA elementary trees that their frequency usage during parsing all of PTB is zero.

⁶ Refer to <ftp://ftp.cis.upenn.edu/pub/xtag/release-1.15.99/>

onto his extracted grammar. Given any XTAG elementary tree t' with its feature structure the proposed mapping procedure would be summarized in tree step:

1. Doing node local transformation such that t' became compatible with extracted grammar (G) format.
2. Doing structural transformation to alleviate the syntactic phenomena differences between XTAG and automatically extracted elementary trees.
3. Searching the extracted elementary tree to find some elementary trees that are similar to modified t' .

Using the mentioned procedure, an XTAG elementary trees t' will mapped onto a set of MICA elementary trees. $T = \{t_1, \dots, t_n\}$ Assuming the uniform distribution of T over its corresponding XTAG elementary tree the probability $P(t_j | t_i)$ could be estimated simply as shown in equation 13.

$$P(t_j | t_i) = \begin{cases} 0 & |T| = 0 \\ \frac{1}{n} & |T| = n \neq 0 \end{cases} \quad (13)$$

The values of coefficients $\lambda_1, \lambda_2, \lambda_3$ and λ_4 have been sets manually as 0.4, 0.3, 0.2 and 0.1 respectively. In fact these values shows the confidence in emission matrices. B_{parser} is a matrix achieved from direct comparison between MICA and XTAG parser, Hence we gave it the higher weight. Either of MICA and XTAG supertagger uses the local information of words in supertagging. In addition, they assign a probability to each of suggested supertags. However, there is no guarantee that the combination of elementary trees of a sequence of elementary trees achieved from supertagging process could construct a derivation and hence a derived tree. Hence, B_{stagger} is placed at the second rank. B_{dict} just uses the knowledge embedded in *syn_dic*, B_{chen} just uses the structural information of elementary trees, and they give a uniform probability to all of potentially mapped elementary trees given their anchors.

All of $B_{\text{parser}}, B_{\text{stagger}}, B_{\text{dict}}, B_{\text{chen}}$ and B have been normalized such that the zero elements of them remain with no change. This restriction is due to respecting the EM-based initializing guess regarding the XTAG and MICA elementary pairs that never had been seen in initialization.

By Considering \vec{r}_j as the j th row of emission matrix B_{xx} , normalized vector \vec{R}_j related to \vec{r}_j could be calculated as shown in eq. (14). In this equation 'pinv' refers to Moore-Penrose pseudo inverse of a matrix and operator ' \cdot ' shows the elements multiplication.

$$\vec{R}_j = \vec{r}_j \cdot [\text{pinv}(\vec{r}_j)]^T \quad (14)$$

3.5. Training

As mentioned before the standard algorithm for HMM training is Baum-Welch [5] algorithm, a special kind of Expectation Maximization (EM) [8] algorithm that its convergence is strongly depends on the initial state. The Baum-Welch algorithm is only guaranteed to converge to a local optimal. This algorithm will let us train both transition and emission matrices such that the achieved HMM will maximizes the probability of occurring the train data (TRDB). In fact, given the observation sequence O and the set of possible states in the HMM, it learn the HMM parameters A, B and Π .

The observation sequences used in training step was C corpus, the MICA elementary tree sequences achieved from parsing TRDB. The model have been trained using MATLAB HMM toolbox written by [12] in five different initial state. One, the initial state proposed by init procedure and four randomly selected initial state. Table 1 shows the improvement of logarithm of likelihood of occurring train data (TRDB) during the Baum-Welch iterations starts in five different initial states.

The results show that the initial state founded by proposed EM-based method can improve the likelihood significantly better than randomly selected initial states. The other likelihood values achieved from training HMM started at a randomly selected state are very near to each other such that some of rows of table 1 seem to have the same numbers.

4. Evaluation

Unlike the two previous approaches that just were going to map as many tree frames from the source grammar onto the destination grammar, here we proposed a model to map a MICA elementary tree onto the best candidate of all of XTAG elementary trees regarding the contextual information of the MICA elementary tree. So, this model can map every MICA elementary tree onto a XTAG elementary tree. Hence, due to differences between the defined goal of our solution and both of [7] and [18] solutions, any comparison between them is incorrect.

By the way, accuracy of the proposed model could be measured by the average of correct tagged elements. However, lake of a tagged (supertagged) gold corpus of XTAG elementary trees is an obstacle to identify the correct tagged elements. To compute the count of correct tagged elements, we have provided a test corpus (TSDB) contains about 120 sentences randomly selected from PTB and checked the tagged results manually. Moreover, a comparison between suggested tags of our model and XTAG supertagger have done, named supertagger comparison. The supertagger comparison has been done on both TSDB and Section 24 of WSJ. In addition, we have applied another evaluation named, counting fully correct tagged, by taking count from all fully correct tagged sequences. Combination of XTAG elementary trees assigned to a fully correct tagged sequence would provide a parse tree. The fully correct tagged evaluation was done on both of TSDB and section 24 of WSJ.

To find more statistical information we have divided TSDB into 2 parts; P-1 contains some sentences of TSDB with length smaller than 12 and P-2 the remain sentences of TSDB. Table 2 shows the result of manually checking of the model response for TSDB. As it shows, from all of 1468 words comprises the TSDB, in 82% of times the assigned XTAG elementary trees to their corresponding MICA elementary trees are correct.

Table 3 gives some information about the distribution of error counts in TSDB's sentences. For instance, pair (P-1, 1) shows the number of sentences in P-1 that only one word of them is mapped incorrectly. As it shows the number of incorrect tagged elements of most of sentences of PTB are less than 4. In addition, it shows that 26 sentences in TSDB are fully correct tagged (about 21% of all of TSDB's sentences).

Table 1: Variation of logarithm of likelihood of occurring TRDB during the training using Baum-Welch

	iter 1	iter 2	iter 3	iter 4	iter 5	iter 6	iter 7	iter 8	iter 9	iter 10
Random	-1196903	-549700	-548403	-546527	-543793	-540190	-536046	-531482	-525520	-515840
Random	-1196423	-549681	-548330	-546368	-543521	-539797	-535568	-530921	-524692	-515591
Random	-1196423	-549681	-548330	-546368	-543521	-539797	-535568	-530921	-524692	-514319
Random	-1196423	-549681	-548330	-546368	-543521	-539797	-535568	-530921	-524692	-514319
EM-Based	-842913	-433001	-414553	-406367	-401633	-397559	-394055	-391571	-389615	-388241

Table 4 shows the supertagger comparison over TSDB and section 24 of WSJ. As it shows in 61% of times our model gives the same answer as the XTAG supertagger gives. In addition, according to our experiments, about 24% of WSJ-24 sentences were tagged without any wrong tagging such that the combination of achieved elementary [3] trees could create at least a parse tree (fully correct tagged).

Table 2: Manually evaluation of HMM on TSDB

	Part1	Part2	TSDB
Incorrect	102	152	254
Correct	398	816	1214
All of Observations	500	968	1468
Error rate	20.40%	15.70%	17.30%
Accuracy	79.60%	84.20%	82.70%

Table 3: Distribution of error counts on TSDB sentences

	0	1	2	3	>3	sum
P-1	14	16	12	10	4	56
P-2	12	16	15	12	11	66
TSDB	26	32	27	22	15	122

Table 4: Supertagger comparison over TSDB and Sec. 24 of WSJ

	TSDB	WSJ 24
Incorrect	562	10637
Correct	833	16747
All of observations	1395	27384
Error rate	40%	39%
Accuracy	60%	61%

5. Conclusion

In this paper, we have tackled the problem of mapping Mica's output to XTAG elementary tree sequence as a sequence-tagging problem. An unsupervised sequence tagger based on Hidden Markov Model has been illustrated which produces an XTAG elementary tree sequence given an automatically extracted elementary tree sequence.

Our focus was on the Chen's (Chen 2001) automatically extracted LTAG used in MICA parser. MICA uses the set of supertags automatically extracted from Penn Treebank using the Chen's approach and tries to generate a derivation tree based on that formalism.

The initial values of HMM's parameters are themselves learned by using an EM-based unsupervised method and then the whole parameters are trained by forward-backward algorithm. Also, the trained emission matrix of HMM is

smoothed by an interpolation method on 4 different simpler models

References

- [1] S. Bangalore, P. Boullier, A. Nasr, O. Rambow, and B. Sagot, "MICA: A probabilistic dependency parser based on Tree Insertion Grammar," *North American Chapter of the Association for Computational Linguistics (NAACL)*, 2009.
- [2] S. Bangalore, P. Haffner, and G. Emami, "Factoring global inference by enriching local representations," *Technical report, AT&T Labs - Reserach*, 2005.
- [3] S. Bangalore, and A. Joshi, "Supertagging: An approach to almost parsing," *Computational Linguistics*, 25(2):237-266, 1999
- [4] P. Boullier, and P. Deschamp, *Le syst'eme SYNTAXTM - manuel d'utilisation et de mise en oeuvre sous UNIXTM*. <http://syntax.gforge.inria.fr/syntax3.8-manual.pdf>, 1988.
- [5] L. E. Baum, "An inequality and associated maximization technique in statistical estimation for probabilistic functions of Markov processes," In Shisha, O. (Ed.), *Inequalities III: Proceedings of the Third Symposium on Inequalities*, University of California, Los Angeles, 1972, pp. 1-8. Academic Press.
- [6] J. Chen, S. Bangalore, and K. Vijay-Shanker, "New models for improving supertags disambiguation," in *Proc, EACL'99*, 1999, *Pp. 188-195. Bergen*
- [7] J. Chen, "Towards Efficient Statistical Parsing Using Lexicalized Grammatical Information," *Ph.D. thesis, University of Delaware*, 2001.
- [8] A.P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from in-complete data via the em algorithm," *Journal of the Royal Statistical Society: Series B*, 39(1):1-38, 1977.
- [9] H. Faili, and G. Ghassem-Sani, "An application of Lexicalized Grammars in English-Persian Translation," in *Proceedings of 16th European Conference on Artificial Intelligence (ECAI, 2004, 596-600*
- [10] H. Faili, "From partial toward full parsing," in *Recent Advances In Natural Language Processing (RANL)*, 2009, 71-75
- [11] A. Joshi, L. Levy, and Takahashi, "Tree Adjunct Grammars," *Journal of Computer and System Sciences*. 10(1):136-163, 1975.
- [12] K. Murphy, 1998. A Hidden Markov Model (HMM) Toolbox for Matlab. Available at: <http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html>
- [13] G. Neumann, "Automatic extraction of stochastic lexicalized tree grammars from tree banks," in *forth international workshop on TAG and related frameworks (TAG+4)*, 1998
- [14] J. Park, "Extraction of tree adjoining grammar from a tree bank for Korean," in *Proceedings of the COLING/ACL 2006 Student Research Workshop*, 2006, pages 73-78
- [15] Y. Schabes, and C. Richard, "Tree Insertion Grammar" *Computational Linguistics*, 1995.
- [16] L. Shen, and A. Joshi "Incremental ltag parsing," in *HLT-EMNLP'05*, 2005.
- [17] G. Van Noord, "Head-corner parsing for TAG," in *Computational Intelligence*, 1994, 10(4), pp. 525-534.
- [18] F. Xia, and M. Palmer, "Evaluating the Coverage of LTAGs on Annotated Corpora," in *Proceeding of Workshop on Using Evaluation within HLD Programs. Results and Trends at Second International Conference on Language Resources and Evaluation (LREC)*, 2001, 1-6
- [19] F. Xia, "Automatic grammar generation from two different perspectives," *Ph D. thesis, University of Pennsylvania*, 2001.
- [20] XTAG-group, "A Lexicalized Tree Adjoining Grammar for English", *University of Pennsylvania, Technical Report IRCS 98-18, Institute for Research in Cognitive Science*, pp. 5-10, 1998.