Natural Language Engineering **22** (2): 215–255. © Cambridge University Press 2015
doi:10.1017/S1351324915000017

A statistical model for grammar mapping

A. BASIRAT[1,2], H. FAILI[1,3] and J. NIVRE[2]

[1]*School of Electrical and Computer Engineering, College of Engineering, University of Tehran, Tehran, Iran*
email: ali.basirat@lingfil.uu.se
[2]*Department of Linguistics and Philology, Uppsala University, Uppsala, Sweden*
email: joakim.nivre@lingfil.uu.se
[3]*School of Computer Science, Institute for Research in Fundamental Sciences (IPM), P. O. Box 19395-5746, Tehran, Iran*
email: h.faili@ut.ac.ir

(*Received 1 July 2012; revised 20 January 2015; accepted 22 January 2015; first published online 20 February 2015*)

Abstract

The two main classes of grammars are (a) *hand-crafted grammars*, which are developed by language experts, and (b) *data-driven grammars*, which are extracted from annotated corpora. This paper introduces a statistical method for mapping the elementary structures of a *data-driven grammar* onto the elementary structures of a *hand-crafted grammar* in order to combine their advantages. The idea is employed in the context of *Lexicalized Tree-Adjoining Grammars* (LTAG) and tested on two LTAGs of English: the hand-crafted LTAG developed in the XTAG project, and the data-driven LTAG, which is automatically extracted from the Penn Treebank and used by the MICA parser. We propose a statistical model for mapping any elementary tree sequence of the MICA grammar onto a proper elementary tree sequence of the XTAG grammar. The model has been tested on three subsets of the WSJ corpus that have average lengths of 10, 16, and 18 words, respectively. The experimental results show that full-parse trees with average F_1-scores of 72.49, 64.80, and 62.30 points could be built from 94.97%, 96.01%, and 90.25% of the XTAG elementary tree sequences assigned to the subsets, respectively. Moreover, by reducing the amount of *syntactic lexical ambiguity* of sentences, the proposed model significantly improves the efficiency of parsing in the XTAG system.

1 Introduction

Computational grammars are among the mathematical tools used for modeling natural languages. Information about the morphology, syntax, and semantics of a modeled language can be incorporated into a grammar to represent the structure of the language. These resources can be presented in various formalisms such as context-free grammars (CFGs) (Chomsky 1959), head-driven phrase structure grammars (HPSGs) (Pollard and Sag 1994), combinatory categorial grammars (CCGs) (Steedman 2000), lexical-functional grammars (LFG) (Kaplan and Bresnan 1982), and LTAGs (Joshi 1985).

Generally, there are two sources of knowledge for the development of a grammar: (1) linguistic theories and (2) the implicit knowledge embedded in treebanks (Xia 2001). A grammar resulting from the former source of knowledge gives a theoretical

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

linguistic perspective on the language. Aside from the syntactic descriptions, the elementary structures of these grammars are enriched with complex descriptions, such as semantic representation. The linguistically motivated properties of these grammars make them suitable for various NLP tasks that require complex descriptions of grammar elements (e.g., semantic interpretation, and machine translation) (Bangalore, Haffner and Emami 2005). On the other hand, a grammar resulting from the latter source of knowledge gives a data-oriented perspective on the language. The main focus of this kind of grammars is on the statistical distribution of co-occurring syntactic phenomena induced from training corpora. These grammars are suitable for use in the modern statistical approaches proposed for the NLP applications such as statistical parsing, and statistical machine translation.

This paper deals with the problem of linking these two perspectives together (*i.e.,* theory-oriented and data-oriented perspectives) by making a bridge between their related grammars. The bridge can provide a way to augment the theoretical perspective of a language with the data-oriented perspective of that language (and vice versa) in order to combine their advantages.

The idea is implemented in the context of LTAG. The main reason for this choice is the special and practical properties of LTAGs for representing the structural descriptions of syntactic phenomena (Kroch and Joshi 1985; Frank 2004). The power of this formalism is the direct result of the way it factors out recursions and dependencies from the domain of locality. Moreover, the virtue of having the extended domain of locality allows the formalism to impose syntactic and semantic constraints on the relevant arguments of the same elementary structure. The widespread use of this formalism in natural language processing applications such as summarization (Eddy, Bental and Cawsey 2001), discourse parsing (Forbes *et al.* 2003), syntactic parsing (Schabes and Joshi 1988; Bangalore and Joshi 1999; Bangalore *et al.* 2009), information retrieval (Chandrasekar and Bangalore 1997), and machine translation (Shieber and Schabes 1990; DeNeefe and Knight 2009; Ma and McKeown 2012) is another reason to concentrate on this formalism.

Formally, an LTAG is a tree-generating system that forms a language by a set of derived trees (Joshi and Schabes 1997). The elementary units of rewriting in this formalism are the basic syntactic structures, which are represented in the form of trees, called *elementary trees.*[1] Each elementary tree is associated with a lexical item, called the *anchor*, and defines a syntactic environment in which the anchor can appear. Accordingly, the elementary trees directly address the concept of the *supertag*, which is defined as a complex description of the syntactic environment of the lexemes (Bangalore *et al.* 2005; Bangalore *et al.* 2009).

In the terms of grammar development methods, LTAGs can be categorized into two groups: manually crafted LTAGs and automatically extracted LTAGs (Bangalore *et al.* 2005). The former group includes a set of detailed LTAGs whose elementary trees are hand-crafted by language experts. The elementary trees of these LTAGs are annotated with feature structures which serve to make them independent

[1] For the sake of brevity, the terms *elementary tree* and *e-tree* are hereafter used interchangeably.

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

of a certain textual domain, as opposed to extracted grammars, which depend on the textual domain of the data from which they have been extracted. The English XTAG grammar (XTAG-Group 2001), the Korean XTAG grammar (Han *et al.* 2002), the TAG-based grammar for German (Kallmeyer *et al.* 2008), and the French LTAG (Abeillé, Clément and Toussenel 2003) are some examples of manually crafted LTAGs.

The latter group includes a set of LTAGs whose elementary trees are automatically extracted from some annotated corpora. These LTAGs contain a large number of elementary trees and statistical information about their occurrence in syntactic environments. Unlike the manually crafted LTAGs, these grammars are dependent on the domain on which they were trained. Nonetheless, they can provide suitable coverage of the syntactic phenomena of their training corpora. Some examples of this kind of grammars are presented in Chen (2001), Xia (2001), and Chiang (2000) for English, Park (2006) for Korean, Kaeshammer (2012) for German.

This paper focuses on two LTAGs for English: the English XTAG grammar (XTAG-Group 2001), named *XTAG grammar*, and the automatically extracted LTAG used by the MICA parser (Bangalore *et al.* 2009), named *MICA grammar*. The XTAG grammar is a well-known manually crafted LTAG that is enriched with some semantic representations that enables the grammar to be used in semantics-related tasks such as discourse semantic parsing (Forbes *et al.* 2003). The semantic representation is modeled via the feature structures and is combined by a unification process. A downside of the XTAG grammar is that it does not provide statistical information for the elementary trees, and consequently it cannot benefit from the robust statistical LTAG parsing methods (Chen 2001).

On the other hand, the MICA grammar is a *Lexicalized Tree-Insertion Grammar* (LTIG) that is automatically extracted from Penn Treebank. A tree-insertion grammar (TIG) is a variant of tree-adjoining grammar (TAG) with some restrictions on both the structure of elementary trees and the tree attachment operations.[2] Unlike the XTAG grammar, the statistical nature of the MICA grammar makes it suitable for use in efficient statistical parsing methods. However, like most automatically extracted LTAGs, the usability of the MICA grammar in real world applications is limited because of the large number of elementary trees as well as the sparseness problem in the set of elementary trees (Chen 2001; Bangalore *et al.* 2005). More discussion about these LTAGs is provided in Section 3.

Considering all of these strengths and weaknesses of the XTAG and MICA grammars, the LTAG-mapping model is expected to provide a way to compensate the weaknesses of each grammar with the strengths of the other. For instance, as discussed in Section 6.2.3, the model could significantly improve the efficiency of parsing in the XTAG-based applications. In fact, by utilizing the statistical strength of MICA, the LTAG-mapping model makes it possible

[2] Since this paper does not deal with the distinctive features of the TAGs and TIGs, for the sake of simplicity we will refer to the MICA grammar as an LTAG. The differences between TIG and TAG are discussed in Section 3.

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

to map the output of the MICA parser onto a derivation tree in the XTAG grammar. In addition, it can provide the MICA grammar with the semantic representation of the XTAG grammar and allow it to appear in the semantics-related tasks.

The idea of linking LTAGs together is based on *corresponding elementary trees*, which are a set of paired elementary trees of the source and target grammars representing the same syntactic environments for their anchors. The main difficulties in finding the corresponding elementary trees are the *structural differences* and the *many-to-many* relationship between them. The structural differences refer to the differences between the grammars in modeling the syntactic structures. For instance, the elementary trees of the XTAG grammar related to phrase structures are larger than the corresponding elementary trees in the MICA grammar. The many-to-many relationship between the corresponding elementary trees refers to the fact that finding corresponding elementary trees is a context-dependent problem. This makes it difficult to use rule-based methods, which rely on the structural similarities of elementary trees, for finding corresponding elementary trees.

These difficulties led us to closely follow the approach proposed by Basirat and Faili (2013). In this work, the problem of finding *corresponding elementary trees* is formulated as a sequence tagging problem in which a *sequence of elementary trees* of the MICA grammar (instead of a single elementary tree) is mapped onto a *proper sequence of elementary trees* of the XTAG grammar, given the *structural* and *contextual* information of the MICA elementary trees. By the *proper elementary tree sequence*, we mean an XTAG elementary tree sequence that has the closest syntactic interpretation to the MICA elementary tree sequence and provides the best syntactic interpretation of the input sentence. More details about this concept are provided in Section 5.2.1.

The sequential formulation of the problem can be placed against the *context-independent* formulation presented by Chen (2001) and Xia and Palmer (2000), which relies only on the structural similarities of the elementary trees. In the sequential formulation of the LTAG-mapping model, it is assumed that the MICA elementary tree sequence is the output of supertagging an input sentence with the elementary trees of the MICA grammar. This sequence represents a syntactic interpretation of the input sentence from the MICA viewpoint. The XTAG elementary tree sequence, which is the direct translation of the MICA elementary tree sequence, also provides another syntactic interpretation of the sentence from the XTAG viewpoint.

Each of the corresponding XTAG and MICA elementary tree sequences that are assigned to a sentence can be considered an abstraction of both the MICA and XTAG grammars, one that simplifies the syntactic and semantic analysis of the original sentence. In fact, the set of elementary trees assigned to the words of the input sentence defines a subgrammar of the XTAG and MICA grammars which contains all of the essential syntactic and semantic information for analyzing the sentence. From this point of view, the LTAG-mapping model follows the general idea of lexical disambiguation introduced by Bonfante, Guillaume and Perrier (2004). In this regard, the LTAG-mapping model functions as a tool to reduce the amount

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

of *syntactic lexical ambiguity* of a sentence, which is the number of elementary trees selected by the words of the sentence (Sarkar 2007).

The sequence tagger proposed in this paper has two main steps. In the first step, the source MICA e-tree sequence is mapped onto a target e-tree sequence of the XTAG grammar using a sequence tagger that works based on a *Hidden Markov Model* (HMM). The HMM, which is responsible for modeling the hidden relationship between the LTAGs, is trained using the standard unsupervised HMM training algorithm called *Baum-Welch* (Baum *et al.* 1970). The initial state of the HMM is also estimated using a novel *EM-based semi-supervised* algorithm. This algorithm attempts to find the HMM parameters according to a parallel corpus annotated with the MICA and XTAG e-tree sequences.

In the second step, the resulting XTAG e-tree sequence is rectified to make it compatible with the XTAG grammar assumption based on the elementary trees that can be assigned to a word. After the incorrectly assigned elementary trees are identified, they will be replaced with the most probable elementary trees of the XTAG grammar. A statistical, word-based model that relates the lexical items of the language to the elementary trees of the XTAG grammar is used for this purpose.

The most important improvements of this method in relation to the previous work (Basirat and Faili 2013) are as follows:

- Improving the HMM training algorithm
- Incorporating more contextual features of the problem of training the model
- Using an error-correction phase to polish the output of the sequence tagger
- Increasing the size of the training data (number and length of sentences)

Our experimental results show that these changes could significantly improve the quality of the model, specifically when it is used as a parser.

Two major contributions of this work can be discerned. First, it addresses the shortcoming of the hand-crafted and data-driven LTAGs of a language and proposes a solution to combine their advantages. For instance, as will be discussed in Section 4, this solution can be used to improve the efficiency of parsing in the XTAG grammar, the main limiting factor of the XTAG grammar in real world applications. In this regard, by reducing the amount of *syntactic lexical ambiguity* of a sentence, the LTAG-mapping model can significantly improve the performance of parsing in the XTAG grammar. The syntactic lexical ambiguity of a sentence is a criterion that measures the syntactic complexity of the sentence and is the major contributor to the time complexity of parsing (Sarkar 2007).

Second, this work provides a solution to the problem of sparseness in the tree frames of the automatically extracted LTAGs. The tree frames in the TAG formalism are simply the elementary trees, where the lexical items have been excised. The data sparseness is an inevitable consequence of using statistical methods to extract TAGs from treebanks (Chen 2001). This problem mainly arises when the occurrence of zero probability elementary trees could not be predicted by statistical processing models that use these grammars. The large number of tree frames is the main reason for this weakness of the automatically extracted grammars. For example, there are 4,726 tree frames in the MICA grammar whereas the single-anchor tree

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

frames in the XTAG grammar number about 500. The LTAG-mapping model makes it possible to abstract a big LTAG with thousands of tree frames into a smaller LTAG containing few tree frames.

Other contributions of this work are briefly summarized below:

- It introduces a method for finding the initial state of an HMM, when there is partial information about the relationship between hidden states and observation symbols.
- It introduces a method for resolving the problem of how to disambiguate a set of elementary tree sequences assigned to a sentence.
- It could improve the performance of parsing in the XTAG grammar.

The layout of this paper is as follows. The next section reviews the previous work done in this field. Section 3 introduces the XTAG and MICA grammars in more detail. An applicable architecture of the LTAG-mapping model is proposed in Section 4. Section 5 deals with the training of the sequence tagger and the word-based model used in the error-correction phase. And, finally, the experimental results of this work are elaborated in Section 6.

2 Related work

The widespread use of LTAG, alongside its ability to encapsulate syntactic and semantic structures of a language, have motivated many researchers to improve the usability of this formalism in applications for natural language processing. Bridging between two LTAGs of a language in order to combine their capabilities is a method commonly proposed by researchers for this aim. This section outlines the previous work that has been done in this regard.

In general, two major categories of methods have been proposed for bridging between LTAGs of a language: *context-independent* and *context-dependent* methods. The first category consists of methods based on structural similarities of elementary units of the grammars regardless of their contextual environment. The researches proposed by Chen (2001), Xia and Palmer (2000), and Ryant and Kipper (2004) belong to this category. The second category consists of methods that attempt to base the bridge on the behaviors of the grammars in similar syntactic environments. In these methods, the corresponding elementary structures of the grammars are mapped onto each other using a parallel corpus annotated with the elementary structures of both grammars. Applying these methods depends on the availability of a probability distribution that associates probabilities to the pairs of elementary structures with respect to their syntactic environment. The mapping approaches proposed by Chiang and Rambow (2006), Basirat and Faili (2011), and Basirat and Faili (2013) belong to this category.

Chen (2001) proposed a conversion method to map an individual XTAG elementary tree onto an automatically extracted elementary tree using a heuristic rule-based classification procedure. The method was intended to enrich the automatically extracted LTAG with the available semantic representations of the XTAG grammar. To this end, a function is proposed to find the elementary trees that are structurally

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

similar to each other. The function works based on two transformations on the XTAG elementary trees: *node local transformation* and *structural transformation*. The *node local transformation* alleviates the minor difference between the XTAG and the automatically extracted elementary trees. The *structural transformation* reduces the syntactic differences between the automatically extracted elementary trees and the output of the *node local transformation* phase. The main goal of the Chen's procedure is to map as many tree frames in the automatically extracted grammar onto the XTAG tree frames as possible. As mentioned in Section 1, tree frames are lexicalized trees whose anchors have been excised. Chen's mapping procedure could map a small number of XTAG tree frames, about 4%, onto the extracted grammar. In spite of the low value of the mapping result, Chen has argued that "it perhaps is not indicative of much effort for small gain." In fact, he has stated that the procedure could successfully map the most frequently occurring subcategorizations in the extracted grammar. Moreover, about 30% of the XTAG tree frames anchored by verbs could be mapped onto the extracted grammar using Chen's procedure.

Xia and Palmer (2000) also used a similar rule-based method to calculate the overlap between XTAG and the LTAG that has been automatically extracted from the Penn Treebank using the LTAG extraction tool proposed by Xia (1999). In their work, an XTAG elementary tree and an automatically extracted elementary tree are matched if they satisfy one of two types of matching called *t-match* and *c-match*. The t-match focuses on the structural similarities of the elementary trees regardless of any extra information that the grammars provide for the elementary trees, such as the feature structure in the XTAG grammar or the frequency information in the extracted grammar. In addition, since the extracted grammar does not have multi-anchor trees, the expanded subtrees of the XTAG elementary trees that lead to the multi-anchor trees are disregarded. The c-match is defined as a response to the weakness of the t-match that it does not provide any solution to minor annotation changes between two grammars. In c-match, each elementary tree is decomposed into three parts, *subcategorization frame*, *subcategorization chain* and *modification pair* and two elementary trees are considered to be c-matched if they are decomposed into the same tuple *(subcategorization frame, subcategorization chain, modification pair)*. The experiments of Xia and Palmer (2000) show that 59.9% of tree frames in the extracted grammar match a corresponding tree frame in the XTAG grammar. They have also shown that the percentage of matched tree frames increases from 59.9% to 82.1% if the annotation differences between the XTAG and the extracted grammar are discarded. These differences mostly concern how an adjective or an adverb modifies a noun or a verb, respectively. For instance, an adjective modifies a noun directly in the XTAG grammar, whereas in the extracted grammar, an adjective projects to an Adjective Phrase (AP) which in turn modifies a noun.

Ryant and Kipper (2004) extended the syntactic coverage of VerbNet (Dang, Kipper and Palmer 2000; Kipper, Dang and Palmer 2000) by incorporating the coverage of the XTAG grammar. VerbNet is a domain independent verb lexicon with explicit syntactic and semantic information for English. Due to the limitation that the VerbNet syntactic coverage only describes the declarative frame

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

for each syntactic construction or alternation, mapping syntactic information from this resource onto XTAG elementary trees could increase its robustness by capturing possible transformations of each frame (Ryant and Kipper 2004). Moreover, the semantic predicates presented in VerbNet could be used to disambiguate the XTAG verb senses (Ryant and Kipper 2004). Out of 196 VerbNet frames, all but 18 were found to correspond exactly to 16 of 57 XTAG tree families by doing manual mapping between XTAG trees and VerbNet frames, they reported.

Introducing a novel concept, called the *hidden TAG model*, Chiang and Rambow (2006) linked the various spoken dialects of Arabic with a standard written language for Arabic, called *Modern Standard Arabic* (MSA). They raised the question: *how can one create a statistical parser for an Arabic dialect when there is no large corpus of parsed data for training the parser?* In response, a model analogous to an HMM that couples a model of MSA trees with a translation model between MSA and another Arabic dialect was proposed. A sentence from the Arabic dialect could then be parsed by simultaneously building a forest of MSA trees and the corresponding trees of the dialect, the authors wrote. After testing the model on Levantine Arabic, one of the major varieties of Arabic, and on MSA, they reported 67% accuracy.

Basirat and Faili (2011), a statistical approach was proposed for enriching an automatically extracted LTAG used by MICA parser (Bangalore *et al.* 2009) with the semantic representations provided by the XTAG grammar. They introduced a new formulation of the problem in which a *sequence* of elementary trees of the MICA grammar was mapped onto a *sequence* of elementary trees of the XTAG grammar. Using a HMM, they were able to benefit from the contextual information of elementary trees to find the corresponding elementary trees of the grammars. Then, by improving the structure of the HMM, Basirat and Faili (2013) proposed a fully automated model that also could model the overall gap between the source and target grammars. The model was tested on three subsets of the ATIS (Hemphill *et al.* 1990), IBM manual, and WSJ (Marcus *et al.* 1993) corpora. They reported 80.00%, 88.30%, and 91.50% mapping accuracy between the corresponding MICA and XTAG elementary trees assigned to the subsets of ATIS, IBM, and WSJ, respectively. In the terms of parsing, they show that the full-parse trees could be built from 35.46%, 47.36%, and 61.73% of the XTAG elementary tree sequences assigned to the subsets of ATIS, IBM, and WSJ, respectively.

As mentioned in Section 1, this paper has closely followed the LTAG-mapping approach proposed by Basirat and Faili (2013) with some modifications in both the architecture of the model and the training algorithms. These changes make the model a more applicable, accurate, and reliable system for bringing two LTAGs together. The model has been tested on the three subsets of the WSJ corpus with average lengths of 10, 16, and 18 words. The experimental results show that full-parse trees with average F_1-scores of 72.49, 64.80, and 62.30 points could be built from 94.97%, 96.01%, and 90.25% of the XTAG elementary tree sequences assigned to the subsets, respectively. In addition, by reducing the *syntactic lexical ambiguity* of the sentences, the proposed model significantly improves the efficiency of parsing in the XTAG system.

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

3 XTAG and MICA

The English XTAG grammar (XTAG-Group 2001) is a well-known hand-crafted LTAG developed by the XTAG research group at the University of Pennsylvania. The grammar contains morphological and syntactic information on the words. It has a tree database containing 1,226 tree frames, anchored in around 115,000 words, providing good syntactic and semantic constraints for the words of English. The linguistically motivated descriptions of the lexical items of the grammar caused it to be considered a domain independent grammar that can be widely used in many TAG-based applications such as parsing, machine translation, information retrieval, and semantic interpretation.

In terms of syntactic coverage, the XTAG grammar can handle a wide range of syntactic phenomena in the English language (e.g., PRO-constructions, infinitives, passives, relative clauses, gerunds, etc.) (Bangalore *et al.* 1996). Able to detect correct grammatical constructions of the language, XTAG can mainly be considered as an acceptor rather than a generator of English sentences. Nevertheless, there are some truly acceptable English sentences that cannot be handled by the grammar. In fact, our experiments have shown that about 5% of WSJ sentences up to 20 words could not be parsed in the XTAG grammar.

From the semantic point of view, the XTAG grammar is well enriched with semantic representations. The elementary trees of the XTAG grammar are associated with semantic representations which can be used in other semantic related tasks, such as discourse semantic parsing (Forbes *et al.* 2003). These representations are modeled via the feature structures that have been in the XTAG project. The feature structures are combined by a unification process. Details on associating the semantics with a TAG elementary tree can also be found in Stone and Doran (1997), Gardent and Kallmeyer (2003), and Joshi and Vijay-Shanker (2001).

There is no efficient parser for the XTAG grammar. This is due to the fact that the problem of TAG parsing can be reduced to the problem of boolean matrix multiplication (BMM) (Satta 1994), and therefore its complexity depends on the complexity of algorithms for this problem. The algorithms with the lowest complexity for BMM, however, are impractical for parsing, *i.e.,* the practical lower bound for TAG parsing remains high in comparison with CFG (Lee 2002).

The currently available XTAG parser relies on a chart-based head-corner parsing algorithm running in $O(n^6)$ time complexity.[3] In practice, parsing in the TAG formalism strictly depends on both the size of the input grammar and the length of the sentence. In fact, in the worst case, the algorithm takes $O(\mid A \mid\mid A \cup I \mid Nn^6)$ time and $O(\mid A \cup I \mid Nn^4)$ space to parse a sentence of length n; A being the set of auxiliary trees, I being the set of initial trees, and N being the maximum number of nodes in an elementary tree (Joshi and Schabes 1997). The output of the parser for any input sentence is an ambiguous set of derived and derivation trees with no clear criteria of priority between them. Figure 1 represents the number of parse trees associated with sentences of maximum length of 22 words. As shown, by increasing

[3] The XTAG parser is available at http://www.cis.upenn.edu/∼xtag/.

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

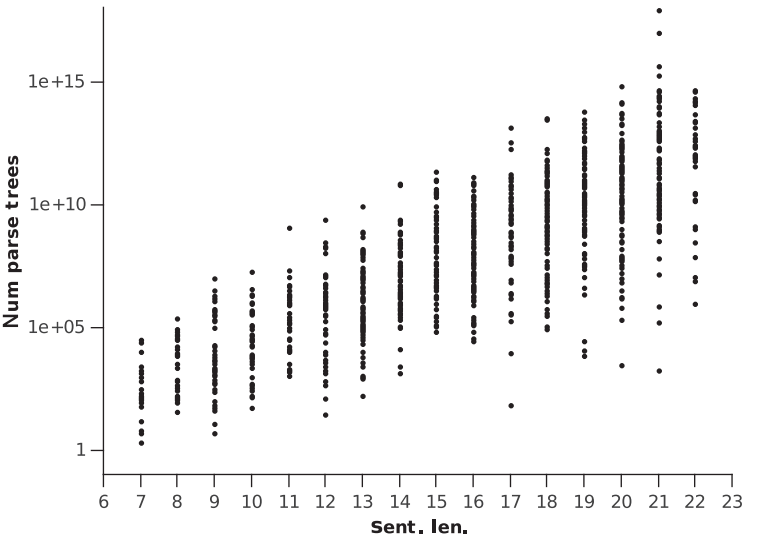


Fig. 1. Total number of parse trees generated by the XTAG parser.

the length of the sentences, the number of parse trees produced by the parser is drastically increased so that in some cases they reach a peak of 10^{15} parse trees for a sentence with a length of around 20 words. The lack of statistical information for the XTAG grammar is another cause of the problem of parsing in this grammar. This weakness of the grammar prevents the use of powerful statistical methods of parsing for it. Some efforts (e.g., part-of-speech (POS) filtering) have been made to improve the performance of parsing in the XTAG grammar, but the problem remains.

On the other hand, the MICA grammar is a LTIG which has been automatically extracted from the Penn Treebank using Chen's approach (Chen 2001). TIG is a limited variant of TAG which combines the efficiency of CFG with the lexicalizing power of the TAG formalism (Schabes and Waters 1995). The main difference between TIG and TAG is that in TIG, the adjunction operation is restricted. TIG therefore only generates context-free languages whose sentences can be parsed in $O(n^3)$ time. Moreover, unlike the TAG formalism, which does not impose any restrictions on the different types of auxiliary trees, TIGs only have *left* or *right* auxiliary trees for which every nonempty frontier node is to the left or right of the foot node, respectively.

MICA, standing for Marseille-INRIA-Columbia-AT&T, is itself a dependency parser that returns the deep dependency representations of the given sentence with 88.6% accuracy for unlabeled dependency trees on the Wall Street Journal part of the Penn Treebank. MICA contains two grammars for computing the *n*-best parse and dependency trees of an input sentence, a TIG and a *Probabilistic Context Free Grammar* (PCFG). The former grammar, here known as *MICA grammar*, contains 4,726 single-anchor tree frames as well as about one million words that are anchored in them. This grammar has almost full coverage of all syntactic structures in the Penn Treebank. The only sources of linguistic information used for extracting the

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

elementary trees of MICA grammar are a *head percolation table* for determining the head of the interior nodes of the treebank parse trees, and an *argument table,* and a *tagset table* used together to determine the complement or adjunct status of a given node.

The latter grammar, however, is a large scale PCFG which generates strings of elementary trees. This grammar is directly obtained from the TIG by doing some systematic transformations on its elementary trees. For each elementary tree in the TIG, there would be a related rule in the PCFG whose left-hand side refers to the root category of the elementary tree. The left-hand sides of the rules related to the initial elementary trees are exactly same as the root category of the trees. The left-hand sides of the rules related to the auxiliary trees are symbols combining both the type of the auxiliary tree (which is left or right) and the root category of the elementary tree. The right-hand sides of the rules are the string representation of the interior nodes of the elementary trees which are built during the top down traversal of them. More discussions about this transformation can be seen in Bangalore *et al.* (2009).

Parsing in MICA is carried out in two main steps: *supertagging* and *actual parsing*. In *supertagging* (Bangalore and Joshi 1999), it attempts to associate the words of the input sentence with the elementary trees of the TIG using a maximum entropy model that is trained on the contextual attributes of the tagged words. Then, in the actual parsing, it derives the set of parse trees from the PCFG rules that are built from the set of elementary trees assigned to the input sentence by the supertagger. To this end, the SYNTAX (Boullier and Deschamp 1988) parser generator is used to generate an Earley-like parser for analyzing the PCFG. The Earley-like parser computes a shared parse forest containing all parse trees of the input sentence. Then, an *n*-best filtering module is used to filter the forest. The result is an *n*-best forest of derived/derivation trees with a clear boundary between the best parse trees (BPTs) and the others.

In this paper, we try to take advantage of both tree-based grammars described above, the XTAG grammar and the MICA grammar. By automatically mapping each MICA elementary tree sequence onto a proper XTAG elementary tree sequence, we aim to bring together the statistical properties of the MICA grammar and the semantic aspects of the XTAG grammar.

4 Architecture of the model

Figure 2 outlines a practical architecture of the LTAG-mapping model as a parser for the target grammar, XTAG grammar. As shown in the figure, the parsing process is carried out in three main steps:

(1) supertagging
(2) LTAG mapping
(3) LTAG parsing

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

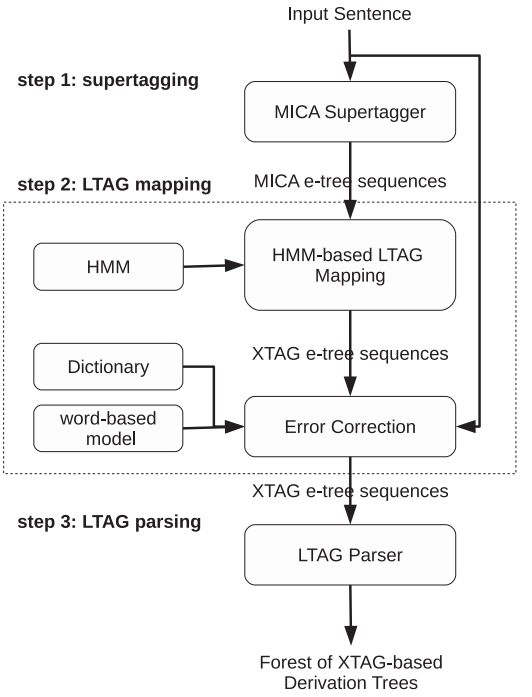


Fig. 2. Basic architecture of the model to be used as target parser.

First, the input sentence is annotated with a set of most probable MICA elementary tree sequences. This phase, *supertagging*, can be carried out using the MICA supertagger.[4]

In the second step, LTAG mapping, the resulting MICA elementary tree sequences are mapped onto a set of most probable XTAG elementary tree sequences. The LTAG mapping is done in two steps:

(1) HMM-based LTAG mapping
(2) error correction

In the HMM-based LTAG mapping, an HMM is used to map the MICA elementary tree sequences (i.e., observation sequences of the HMM) onto their related XTAG elementary tree sequences (i.e., the most probable hidden state sequences of the HMM). Then, in the error-correction phase, the XTAG elementary tree sequences generated by the HMM are fine-tuned to be compatible with the XTAG grammar. The available dictionary of the XTAG project is used for detecting elementary trees that are incorrectly assigned to the words of the input sentence. The dictionary relates each word of the language to a set of elementary trees. The erroneous elementary trees are then replaced by a set of XTAG elementary trees proposed by a probabilistic word-based model that assigns the most probable XTAG elementary trees to the words.

[4] The MICA package is available at http://mica.lif.univ-mrs.fr/.

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

Finally, in the third step, the resulting XTAG elementary trees are attached to each other in order to build the final set of derived/derivation trees. The attachment process is performed based on two standard tree operations defined in the TAG formalism, *substitution* and *adjunction*.

The main contribution of this paper concerns the second step of the proposed model, and consists of an attempt to find a set of most probable XTAG elementary tree sequences for an input sentence given its related set of MICA elementary tree sequences. The LTAG-mapping model can improve the efficiency of the XTAG parser by reducing the amount of *syntactic lexical ambiguity* of the input sentence (Sarkar 2007). The rest of this paper details the algorithms used for training the HMM and the word-based model used in the error-correction phase.

5 Training

As previously outlined, the sequential LTAG-mapping model is founded on two statistical models: an HMM, which maps a sequence of MICA elementary trees onto a proper sequence of XTAG elementary trees; and a word-based model that improves the output of the HMM according to the original sentence. The Baum–Welch algorithm can be used for training the HMM. The algorithm is an expectation maximization (EM) algorithm that attempts to find the HMM parameters in an unsupervised manner in order to maximize the likelihood of observing the observation sequences. As with most EM-based algorithms, the main weakness of the Baum–Welch algorithm is that it gets stuck in the local optima and there is no guarantee of finding the globally optimal solution. In this case, a proper initial state can significantly help the algorithm to converge to a better local optimum. Another EM-based algorithm is used to find the initial state of the HMM. This algorithm will be explained in detail in Section 5.2.1.

A semi-supervised algorithm is also used for estimating the probabilities of the word-based model. The algorithm looks for statistical relationships between the English words and the XTAG elementary trees. This algorithm will be described in detail in Section 5.3.

Before going into the algorithms in the next section, we first provide an outline of the data used for training the HMM and the word-based model.

5.1 Data preparation

As outlined in the previous section, training the HMM entails maintaining two kinds of data:

(1) A set of observation sequences (i.e., MICA elementary tree sequences) to be used by the Baum–Welch algorithm.
(2) A set of paired sequences of observations and hidden states (i.e., MICA and XTAG elementary tree sequences related to same sentences) to be used by the HMM initializing algorithm. This algorithm will be discussed in detail in Section 5.2.1.

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

The former dataset can be collected directly using the MICA supertagger. For any English sentence $s = (w_1, \ldots, w_n)$, the MICA supertagger generates a list of elementary tree sequences $\{T_1, \ldots, T_{l_s}\}$, where $T_i = (t_{i,1}, \ldots, t_{i,n})$. We will refer to the number of MICA elementary tree sequences assigned to sentence s as l_s. Since the elementary tree sequences are weighted by their correctness probability, the best ones are selected as the training data for the Baum–Welch algorithm.

Preparing the training data for the HMM initializing algorithm requires more effort. This algorithm looks for the HMM parameters to be used as the initial state of the Baum–Welch algorithm. It needs two sets of observation and hidden state sequences. The set of observation sequences, MICA elementary tree sequences, can straightforwardly be collected using the MICA supertagger. The unavailability of a suitable supertagger for the XTAG grammar, however, is the main obstacle in preparing the set of hidden state sequences. There is no supertagger for XTAG because such a tagger would need to rely on statistical information which is not available in XTAG. An elementary tree sequence based on the XTAG grammar can, however, be obtained from the output of an LTAG parser.

Using the XTAG parser introduced in Section 3, an input sentence $s = (w_1, \ldots, w_n)$ is associated with a *set* of XTAG elementary tree sequences $\{T'_1, \ldots, T'_{k_s}\}$, where $T'_i = (t'_{i,1}, \ldots, t'_{i,n})$. Each elementary tree sequence T'_i is extracted from a derivation tree of the sentence s. We will refer to the sentence s associated with the T'_i as $\eta(T'_i)$ and the number of XTAG elementary tree sequences assigned to the sentence s as k_s. Since the parser does not rank the derivation trees, any ambiguity in the derivation trees will remain in the set of elementary tree sequences. So we need to disambiguate them before using them in the LTAG-mapping model.

Equation (1) represents a weighting function based on the correctness ratio of the XTAG elementary tree sequences. The correctness ratio of an XTAG elementary tree sequence T' and sentence s is considered as the joint probability of observing $s = (w_1, \ldots, w_n)$ and $T' = (t'_1, \ldots, t'_n)$ given the HMM_{ex}. The HMM_{ex} models the relationship between an XTAG elementary tree sequence T' and sentence s, which has the same length n.

$$\omega(s, T') = p(s, T' \mid \mathrm{HMM}_{ex}) \tag{1}$$

The HMM is characterized as follows:

- N: the number of hidden states is equal to the number of XTAG elementary trees.
- M: the number of observation symbols is equal to the number of English words.
- state transition probability: the probability of seeing an XTAG elementary tree given the other one.
- observation probability: the probability of observing an English word being generated from a hidden state (XTAG elementary tree).
- prior probability: the probability of observing an XTAG elementary tree in the beginning of a sequence of elementary trees.

Algorithm 1 represents an EM-based algorithm for training the HMM_{ex}. The algorithm tries to estimate the HMM parameters from the values of ω which, in

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

turn, have been estimated from the HMM obtained in the previous step. At the beginning, all of the XTAG elementary tree sequences $\{T'_1, \ldots, T'_{k_s}\}$ associated with sentence s are weighted with uniform probability distribution and the initial value of HMM_{ex} is estimated from the uniform probabilities. Then, with each iteration of the algorithm, (1) the weight (correctness ratio) of the XTAG elementary tree sequences is updated based on the current value of HMM_{ex} (expectation), and (2) the parameters of HMM_{ex} are estimated from the weighted sequences (maximization). The objective function is the probability of observing the observation sequences (i.e., sentences) (2).

$$\log p(S \mid \text{HMM}_{ex}) = \sum_{s \in S} \log p(s \mid \text{HMM}_{ex}) \tag{2}$$

Using the *maximum likelihood estimation* (MLE), the state transition probability $t'_j \mapsto t'_i$ can be estimated from (3).

$$p(t'_i \mid t'_j) = \frac{C(t'_j, t'_i)}{C(t'_j)}. \tag{3}$$

The function $C(x)$ counts the number of times that the subsequence of elementary trees x is observed in the training data. The counts are weighted with the correctness ratio of the XTAG elementary tree sequences of which x is a subsequence (see (4)).

$$C(x) = \sum_{s} \sum_{T'} \omega(s, T') \delta(x, s, T'). \tag{4}$$

The Kronecker delta function $\delta(x, s, T')$ is 1 if $s = \eta(T')$ and x is a subsequence of T'; and 0 otherwise.

Similarly, the observation probability $p(w_i \mid t'_j)$ can be estimated from (5).

$$p(w_i \mid t'_j) = \frac{C(w_i, t'_j)}{C(t'_j)}. \tag{5}$$

Here also, the counts are weighted with the correctness ratio of the XTAG elementary tree sequences. For each word w and XTAG elementary tree t', the count function $C(w, t')$ is defined as below:

$$C(w, t') = \sum_{s} \sum_{T'} \omega(s, T') \delta(w, t', s, T'), \tag{6}$$

where the Kronecker delta function $\delta(w, t', s, T')$ is 1 if and only if $s = \eta(T')$, w is a word of s, and t' is an elementary tree in T' that is assigned to the word w.

The estimated HMM will then be used for updating the weight (correctness ratio) of the XTAG elementary tree sequences in the next iteration (expectation). The process is iterated until no change occurs in the value of the probability presented in (2) or the maximum number of iterations is reached.

The output of this process is the HMM_{ex} which models the relationship between the English words and XTAG elementary trees and a set of XTAG elementary tree sequences weighted with their correctness ratios. The HMM can be utilized in supertagging English sentences with XTAG elementary trees. As will be discussed later, the emission matrix of this HMM can also be used to estimate the probabilities for the word-based model used in the error-correction phase.

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

Algorithm 1 The EM-based semi-supervised algorithm used for disambiguation of the XTAG e-tree sequences

for all XTAG e-tree sequences T' and sentence $s = \eta(T')$ **do**
  $\omega(s, T') \leftarrow \frac{1}{k_s}$ ▷ k_s is the number of e-tree sequences associated with s
end for
compute initial values of HMM_{ex} parameters ((3) and (5))
while not termination criteria satisfied **do**
  for all XTAG e-tree sequences T' and sentence $s = \eta(T')$ **do**
    $\omega(s, T') \leftarrow p(s, T' \mid \text{HMM}_{ex})$
  end for
  update the values of HMM_{ex} parameters ((3) and (5))
end while
  return the weighted set of XTAG e-tree sequences

To summarize, for any English sentence $s = (w_1, \ldots, w_n)$ two sets of elementary tree sequences will be used in the training phase: a set of XTAG elementary tree sequences $\{T'_1, \ldots, T'_{k_s}\}$ which are weighted with their correctness ratios (see (1)), and a set of MICA elementary tree sequences $\{T_1, \ldots, T_{l_s}\}$, which are weighted by the MICA supertagger. The next section introduces a probabilistic model that models the relationships between the corresponding XTAG and MICA elementary tree sequences described in this section.

5.2 HMM-based LTAG mapping

As mentioned before, the relationship between two LTAGs can be modeled by an HMM in which the hidden states are the elementary trees of the target grammar and the observation symbols are the elementary trees of the source grammar. We will refer to this HMM as HMM_{mx} in order to distinguish it from the HMM_{ex} introduced in Section 5.1. The formal definition of HMM_{mx} is:

(1) N: the number of hidden states is equal to the number of elementary trees of the XTAG grammar.
(2) M: the number of the observation symbols is equal to the number of elementary trees of the MICA grammar.
(3) State transition probability: the probability of observing two elementary trees of the XTAG grammar jointly.
(4) Observation probability: the probability of observing an elementary tree of the source grammar being generated from an elementary tree of the target grammar.
(5) Prior probability: the probability of observing an elementary tree of the XTAG grammar in the beginning of a sequence of elementary trees.

The Baum–Welch algorithm (Baum *et al.* 1970) is used for training the HMM_{mx}. The algorithm runs on the set of observation sequences discussed in Section 5.1. In order to make the algorithm converge to a better local optimum, the initial state of the HMM is estimated by a semi-supervised EM-based training algorithm. This algorithm is detailed in the next section.

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

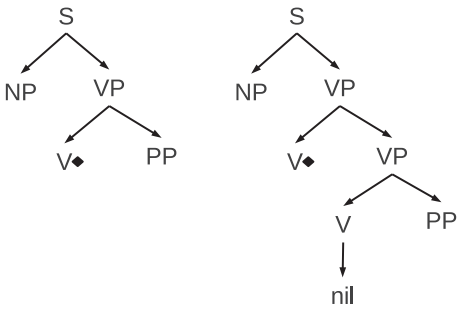


Fig. 3. Two structurally different elementary trees of two LTAGs describing the same syntactic environment for their anchors.

5.2.1 HMM initializing algorithm

The HMM initializing algorithm estimates the initial parameters of HMM_{mx}. The algorithm is analogous to Algorithm 1. The main difference between them is the number of observation sequences associated with hidden state sequences. Algorithm 1 relates a *certain observation sequence* (i.e., a sentence) to a set of hidden state sequences (i.e., a set of XTAG elementary tree sequences). The HMM initializing algorithm, however, discovers the relationship between *a set of observation sequences* (i.e., a set of MICA elementary tree sequences) and the corresponding set of hidden state sequences (i.e., a set of XTAG elementary tree sequences).

The algorithm runs on the training data described in Section 5.1. For each sentence s in the training data, the algorithm needs to weight pairs of the MICA and XTAG elementary tree sequences (T, T') associated with s. We will refer to sentence s as $s = \eta(T, T')$, where T and T' are the MICA and XTAG elementary tree sequences assigned to sentence s, respectively. The weights are computed based on the closeness of syntactic interpretations that the sequences provide for their generating sentence s. We will refer to these weights as the *syntactic compatibility ratio* (or CR for short) between the XTAG and MICA elementary tree sequences.

The two main reasons for introducing the syntactic CR between the sequences of elementary tree are as follows: First, in some cases, the grammars use different tree structures to represent the same syntactic environment. They also use different notations for internal nodes of some elementary trees that model the same syntactic environment of the language. Figure 3, originally presented in Chen (2001), shows two structurally different elementary trees of two LTAGs that represent the same syntactic environment for their anchors.

Second, the grammars can explain sentences syntactically in different ways. For instance, Figure 4 represents different syntactic explanations for the sentence *John is giving books to Mary*. The elementary tree sequence T is provided by the MICA grammar, and the T'_is $(i = 1, 2, 3)$ are provided by the XTAG grammar. All of the sequences assign the same elementary trees to the words *John*, *is*, *book*, and *Mary*. The differences are found in the elementary trees assigned to the words *giving* and *to*. The verb *giving* is classified as a basic transitive verb in T, T'_1, and T'_3, but as a ditransitive verb in T'_2. Both T'_1 and T'_2 suggest that the word *to* is a preposition

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

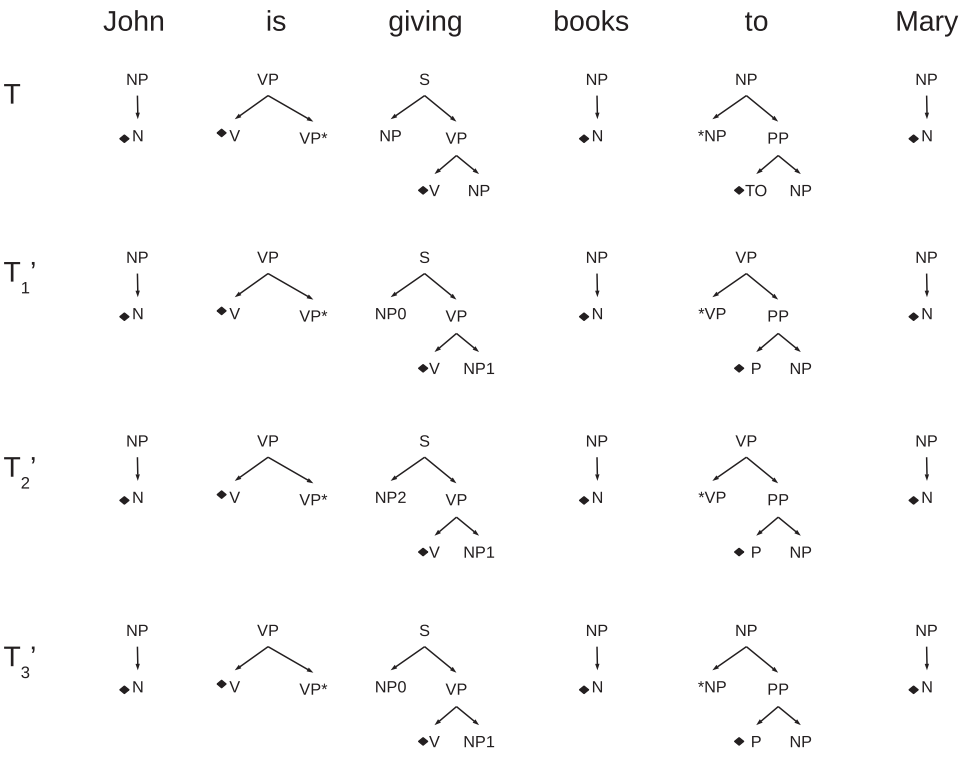


Fig. 4. Different e-tree sequences associated with the sentence "John is giving books to Mary." T is a MICA e-tree sequence and $T'_i, i = 1, 2, 3$ are XTAG e-tree sequences.

depending on a *verb* to its left side. T'_3, however, suggests that the word *to* is a preposition depending on a *noun* to its left side. Similarly, T suggests that the word *to* is a preposition depending on a noun to its left side but the word is categorized as *TO* rather than preposition (P) as in the T'_3.

In this example, the syntactic CR is expected to give the highest value to the pair (T, T'_3) and the lowest value to the pair (T, T'_2). An accurate estimation of syntactic compatibility requires that many parameters to be taken into consideration (e.g., the structure of the elementary trees, the syntactic category of the words, and the dependency structure that the elementary trees suggest for the words). Another approach is to estimate the syntactic CR between two elementary tree sequences $T = (t_1, \ldots, t_n)$ and $T' = (t'_1, \ldots, t'_n)$ assigned to sentence $s = (w_1, \ldots, w_n)$ based on the relative frequency of seeing the corresponding elementary trees t_r and t'_r together. This is based on the assumption that the elementary trees that often appear together are more likely to be related to the same syntactic environment of the language. We estimate the syntactic CR between T and T' as a value proportional to the joint probability of the sequences given the HMM_{mx}. Equation (7) estimates the syntactic CR between T and T' assigned to sentence $s = \eta(T, T')$ respectively. The function ω returns the correctness ratio of its arguments (see Section 5.1).

$$CR(s, T, T') = \omega(s, T)\omega(s, T')p(T, T' \mid \mathrm{HMM}_{mx}). \tag{7}$$

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

Algorithm 2 outlines the EM-based HMM initializing algorithm. The algorithm starts by estimating the syntactic CR between all of the XTAG elementary tree sequences $T'_i \in \{T'_1, \ldots, T'_{k_s}\}$ and MICA elementary tree sequences $T_j \in \{T_1, \ldots, T_{l_s}\}$ assigned to sentence $s = (w_1, \ldots, w_n)$. In this step, since the algorithm has no clue about the HMM_{mx}, the probability $p(T_j, T'_i \mid \text{HMM}_{mx})$ in 7 is replaced with the uniform probability distribution. The initial value of HMM_{mx} is then estimated from the weighted pairs of MICA and XTAG elementary tree sequences. In the EM loop, the CR of the MICA and XTAG elementary tree sequences are updated based on the current value of HMM_{mx} (expectation), and the new weights are used to estimate the new value of HMM_{mx} (maximization). The objective function is the likelihood of observing the observation sequences (i.e., MICA elementary tree sequences).

Both of the transition and observation probabilities of HMM_{mx} are estimated by the MLE. For each pair of XTAG elementary trees (t'_j, t'_i), the transition probabilities $t'_j \mapsto t'_i$ can be estimated by dividing the frequency of seeing (t'_j, t'_i) by the frequency of seeing t'_j. The frequency of seeing a sub-sequence of XTAG elementary trees x is computed from 8, where the Kronecker delta function $\delta(x, s, T, T')$ is 1 if and only if $s = \eta(T, T')$ and x is a subsequence of T'.

$$C(x) = \sum_{s} \sum_{T} \sum_{T'} CR(s, T, T')\delta(x, s, T, T'). \tag{8}$$

For each XTAG elementary tree t' and MICA elementary tree t, the observation probability $p(t \mid t')$ can also be estimated by dividing the frequency of seeing the pair (t, t') assigned to the same word by the frequency of seeing t'. The frequency of seeing the pair (t, t') is computed from 9, where the Kronecker delta function δ is 1 if and only if (1) $s = \eta(T, T')$, (2) t is in T and t' is in T', and (3) t and t' refer to the same word in s.

$$C(t, t') = \sum_{s} \sum_{T} \sum_{T'} CR(s, T, T')\delta(t, t', s, T, T'). \tag{9}$$

Algorithm 2 The EM-based semi-supervised algorithm for initializing HMM_{mx}

for all MICA and XTAG e-tree seq. T and T', and sentence $s = \eta(T, T')$ **do**
  $CR(s, T, T') = \frac{\omega(s,T)\omega(s,T')}{l_s k_s}$ ▷ l_s ans k_s are the number of MICA and XTAG e-tree sequences assigned to s
end for
compute initial values of HMM_{mx} parameters (MLE)
while not termination criteria satisfied **do**
  for all MICA and XTAG e-tree seq. T and T' and sentence $s = \eta(T, T')$ **do**
    $CR(s, T, T') \leftarrow \omega(s, T)\omega(s, T')p(T, T' \mid \text{HMM}_{mx})$
  end for
  update the values of HMM_{mx} parameters (MLE)
end while
  return HMM_{mx}

The output of Algorithm 2 is the initial value of HMM_{mx}. This model will then be trained by the Baum–Welch algorithm. The output of the Baum–Welch

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

algorithm would be the desired model to map MICA elementary tree sequences onto the most probable XTAG elementary tree sequences using the HMM decoding algorithm, called the *Viterbi algorithm*. This process is carried out in the *LTAG-mapping phase* presented in Figure 2. As shown in Figure 2, the XTAG elementary tree sequences resulting from the LTAG-mapping phase will then be processed in the *error-correction phase*. The following section details the error-correction phase.

5.3 Error correction

The output of the HMM-based LTAG-mapping phase needs to be fine-tuned to be compatible with the syntactic interpretation of the target grammar of the input sentence. This process will be performed as an *error-correction* phase in two main steps. At first, the wrongly assigned elementary trees will be detected using some heuristics designed for the set of elementary trees that can be assigned to the words. Then, in the second step, the erroneous elementary trees will be replaced with a set of most probable elementary trees given the contextual information of the words.

There may be several heuristics for finding the erroneous elementary trees assigned to the words of the input sentence. POS agreement between the words and the anchor of the elementary trees is a simple heuristic that can be used for this purpose. In POS agreement, errors are defined as elementary trees whose anchor POS tag differ from the POS of the words to which they are assigned.

In an alternative model, the erroneous elementary trees could be detected using the *word-supertag dictionary* provided by the XTAG grammar. This dictionary contains about 45,000 words, each of which is associated with a set of XTAG elementary trees that can be assigned to the words regardless of their syntactic environment. Equation (10) defines the binary error-detection function ε for word w and XTAG elementary tree e. E_w in this equation is the set of single anchor XTAG elementary trees defined by the *word-supertag dictionary* for word w.

$$\varepsilon(w, e) = \begin{cases} 0 & \text{if } e \in E_w \\ 1 & \text{otherwise.} \end{cases} \tag{10}$$

It is clear that none of the aforementioned heuristics can be guaranteed to find all erroneous elementary trees. This is due to this fact that both of these error detection methods rely only on the lexical information of the individual words while the assignment of elementary trees to the words is a sophisticated process that entails incorporating much contextual information about the word. For instance, POS agreement does not necessarily entail the correct assignment of elementary trees to words, since many elementary trees of grammar can be anchored by the same POS tag. The same situation can also be imagined for the dictionary-based error detection.

After error detection, the erroneous elementary trees are replaced with the correct elementary trees with respect to the input words to which they are assigned. In a blind mode, each erroneous elementary tree can be replaced with all elementary trees of the XTAG word-supertag dictionary associated with the input word. Figure 5 represents the number of elementary trees associated with the 200 most frequently used words

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

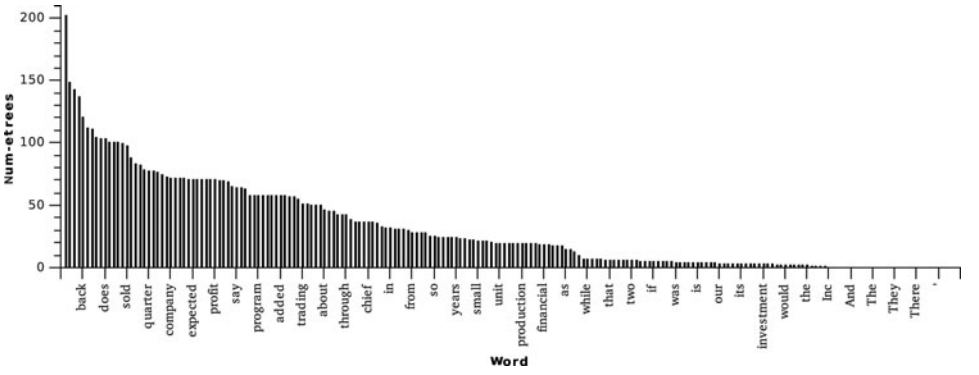


Fig. 5. Number of elementary trees associated with the 200 most frequently used words not satisfying the word-supertag relationship defined in the XTAG word-supertag dictionary.

that do not satisfy the word-supertag relationship defined in the dictionary. As shown, in many cases, more than 50 elementary trees are associated with a word. By increasing the number of erroneous elementary trees per sentence, the blind model will create large number of elementary tree sequences for each sentence, which makes the model impractical.

In addition, it is obvious that some of the errors are due to gaps in the dictionary's coverage of words. For instance, it does not define any single anchor elementary tree for the words belonging to the syntax category *particle* (PL).

These difficulties led us to use a statistical model of error-correction instead of the aforementioned blind model. The model attempts to find the most probable replacement elementary trees for the erroneous ones using some extra contextual information about the word. Equation (11) shows a word-based model that suggests a replacement for the erroneous elementary tree assigned to word w. The probabilities in the model can be estimated from the observation matrix of HMM_{ex} introduced in Section 5.1.

$$t = \arg\max_{\tau \in E_w} p(w|\tau). \tag{11}$$

For each word w that is not defined in the dictionary ($E_w = \phi$), the replacement candidate can be selected from the set of XTAG elementary trees whose anchor POS tags match the POS tag of the word w. We have used another model for finding the most probable elementary trees whose anchor POS tag matches the POS tag of the word (12). The model replaces an erroneous elementary tree assigned to word w with POS tag ρ with the most probable XTAG elementary tree selected from the set T_ρ, which contains all of the XTAG elementary trees whose anchor POS tags are ρ.

$$t = \arg\max_{\tau \in T_\rho} p(\rho|\tau). \tag{12}$$

A simple extension of the proposed error-correction model can be the n-best replacement model in which each erroneous elementary tree would be replaced with an ordered list of elementary trees instead of a single elementary tree. In this case, instead of just having a single XTAG elementary tree sequence, the model would be able to generate a set of XTAG elementary tree sequences for each MICA elementary

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

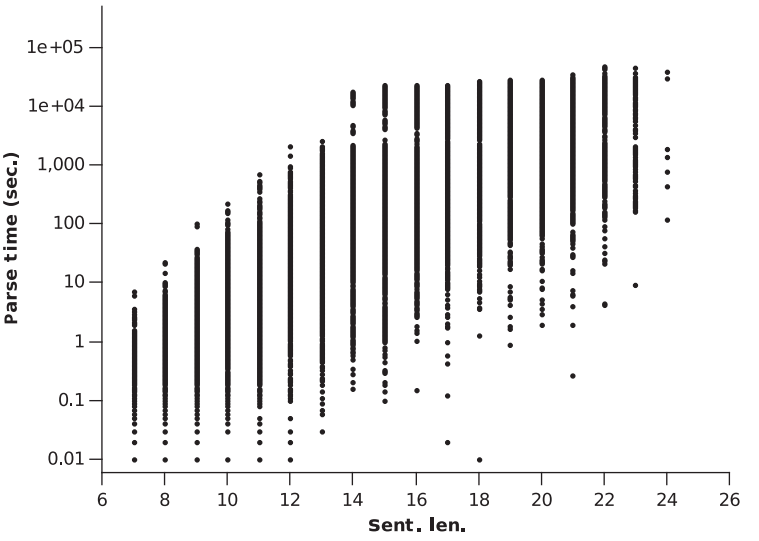


Fig. 6. The amount of parse time taken by the XTAG parser.

tree sequence. Increasing the value of parameter n would be expected to increase the full-parse rate of the model when using it with an LTAG parser (as shown in Figure 2). This would however increase the search space of the parser and consequently increase the time required to parse the input sentence. The effects of this parameter on both the performance and the quality of the model are numerically described in the Section 6.2.3.

6 Experimental evaluation

This section focuses on the numerical illustration of the proposed LTAG-mapping model. First in Section 6.1, the numerical behavior of the training algorithms used for HMM_{ex} and HMM_{mx} will be studied. Then, in Section 6.2, the LTAG-mapping model will be evaluated with respect to different test sets. To this end, three different evaluation criteria have been proposed to deal with different aspects of the model.

All of the experiments, including training and testing the model, have been done on a shared high performance computing (HPC) cluster consisting of around 400 Pentium 4 2.6 GHz processors running Linux 2.6. The system has been used for about four months, including waiting time in the job scheduler queues. Most of the computing resources have been devoted to running the XTAG parser on the train and test data sets. Figure 6 represents the amount of CPU time taken by the XTAG parser to parse the sentences. As shown, the time taken to parse sentences longer than 14 words reaches a maximum of 50,000 seconds per sentence. Sarkar (2007) reported a similar experiment on a Beowulf cluster of Pentium Pro 200 MHz running Linux 2.2. He showed that the time required to parse the sentence longer than 14 words can even reach 10^9 seconds per sentence.

In Section 6.2.3, it is shown that the LTAG-mapping model could reduce the parse time in the XTAG grammar to a few seconds.

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

6.1 Evaluation of the training algorithms

As mentioned before, the proposed sequential LTAG-mapping model needs to train two HMMs, named HMM_{ex} and HMM_{mx}. The former HMM is used in the data preparation phase to disambiguate the elementary tree sequences generated by the XTAG parser. The latter is used in the HMM-based LTAG-mapping phase, which relates the MICA elementary tree sequences to the XTAG elementary tree sequences.

These HMMs and their related training algorithms can be evaluated according to the following criteria:

(1) The probability of observing the observation sequences.
(2) The amount of uncertainty involved in the HMM parameters for observing the observation sequences.
(3) The quality of disambiguation of the XTAG elementary tree sequences. (This criterion is used for evaluating the HMM_{ex}.)

According to the first criterion, the success of an HMM training algorithm entails increasing the likelihood of observing the observation sequences, as shown in (13). In this equation, O is the set of all observation sequences o.

$$\log p(O \mid \text{HMM}) = \sum_{o \in O} \log p(o \mid \text{HMM}). \tag{13}$$

The second criterion refers to the entropy of the distribution of all possible hidden state sequences that explain an observation sequence, as shown in (14). This equation, introduced by Hernando *et al.* (2005), gives the entropy of the distribution of all possible solutions S^T for a given observation sequence O^T and the HMM λ. In the worst case, with N^T possible hidden state sequences, this value would be $T \log N$ and in the best case, with just one hidden state sequence, it would be zero. The lower values for entropy show the lower uncertainty in tracking the hidden states and hence the greater ability of the model to model the hidden Markov process.

$$H(S^T \mid O^T = o^T, \lambda) = -\sum_{S^T} \left[P(S^T \mid O^T = o^T, \lambda) \log P(S^T \mid O^T = o^T, \lambda) \right]. \tag{14}$$

Finally, the last criterion can be used for evaluating the HMM_{ex} used by Algorithm 1 to disambiguate the set of XTAG elementary tree sequences. To this end, the elementary tree sequences proposed by the disambiguation algorithm will be assessed based on the quality of the parse trees that can be built from them. In this case, it is expected that the sequences with the highest values for correctness ratio could lead to the BPTs for their sentences. Using the *PARSEVAL* (Abney *et al.* 1991) evaluation technique, the BPT of an elementary tree sequence can be defined as a parse tree with the highest F_1-score. This definition is similar to the maximization criterion introduced by Goodman (1996). Equation (15) gives a mathematical definition for the BPT of elementary tree sequence T assigned to sentence s. In this equation, set ρ contains all parse trees that can be derived from

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

Table 1. *Statistical information about the training datasets*

Model	Sents.	Avg. Len.	Max Len.	Words
HMM_{ex}—word-based	14,167	13.2	18	187,000
HMM_{mx} Init.	13,428	13	17	174,564
HMM_{mx} Train	26,943	16.3	25	440,556

T, and $F(p)$ stands for the F_1-score of parse tree $p \in \rho$.

$$\mathrm{BPT}(s, T) = \arg\max_{p \in \rho} F(p). \tag{15}$$

The rest of this section deals with the evaluation of the HMM_{ex} and HMM_{mx} with respect to the three aforementioned criteria. First, a brief description of the datasets used for training each of these HMMs is given. Then, the results of the evaluations of the HMMs are presented in detail.

6.1.1 Training sets

The quality of the LTAG-mapping model closely depends on three datasets over which the statistical parts of the model are trained. Table 1 provides some information about the datasets used for training these parts of the model, including the HMM_{ex}, the HMM_{mx}, and the word-based model used in the error-correction phase. The HMM_{ex} and the word-based model are merged into a single row, since both of these models have been trained on the same dataset. *HMM_{mx} Init.* and *HMM_{mx} Train* give information about the training datasets used for *initializing* and *training* HMM_{mx}, respectively.

All of the sentences in these datasets have been randomly sampled from the set of all sentences in WSJ with a maximum length of 25. The columns *Avg Len.* and *Max Len.* show the average and maximum length of the sentences, respectively. The number of sampled sentence (column *Sents.*) and their maximum length were limited by computational resources available to us for parsing the sentences with the XTAG parser. Information about the computing resources used in this work can be found at the beginning part of this section.

6.1.2 Evaluation of HMM_{ex}

HMM_{ex} has been trained using the EM-based HMM training algorithm, presented in Section 5.1. The HMM is used to reduce the ambiguity of the set of XTAG elementary tree sequences by weighting them according to their correctness ratio, as defined in (1). Table 1 describes the training dataset over which the HMM has been trained. On average, 1,300 XTAG elementary tree sequences were related to each sentence in this dataset, in some cases reaching a peak of 70,000 elementary tree sequences per sentence.

The quality of this HMM was measured according to the first and third evaluation criteria presented in Section 6.1. Figure 7, which concerns with the first criterion,

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

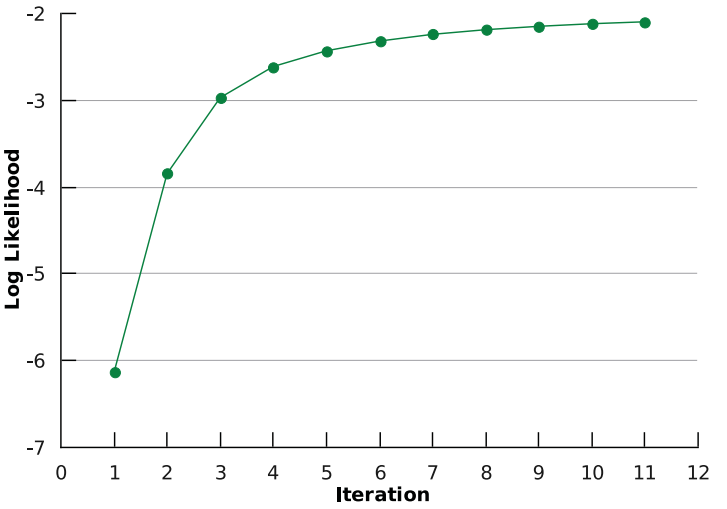


Fig. 7. (Colour online) Logarithmic scale of likelihood of observing English sentences.

represents the value of (13) during training of the HMM_{ex}. The ascending behavior of this diagram is evidence for success of the training algorithm. As shown in this figure, the HMM resulting from each iteration is more likely to model the observation sequences than the other HMMs resulting from previous iterations of the training algorithm.

In terms of the disambiguation of the XTAG elementary tree sequences, the success of the model can be measured according to the quality of the BPTs defined in Section 6.1. Table 2 shows the results of this assessment with respect to the gold parse trees of the Penn Treebank. The table shows the *precision*, *recall*, and F_1-score of the bracketing matches of the unlabeled *BPT*s built from the elementary tree sequences with the highest correctness ratios. The upper bound, in this table, represents the values for parse trees built by the non-statistical Earley-style parser of the XTAG.

The upper bound in Table 2 shows that the XTAG grammar can provide parse trees of the same quality for both short and long sentences. Column BPT, however, shows that using the disambiguation algorithm, the parse trees built from the short sequences are better than the parse trees built from the long sequences. This is mainly because the accuracy of the disambiguation algorithm depends on the length of the sequence. In addition, the differences between the values of the BPT and the upper bound show that the disambiguation algorithm could not find the best elementary tree sequences for all sentences. In fact, our experiments confirmed that the BPTs could be built from around 60% of the elementary tree sequences suggested by the disambiguation algorithm. In the remaining sequences, the number of erroneous elementary trees are at most two.

6.1.3 Evaluation of HMM_{mx}

The quality of HMM_{mx} obtained from Algorithm 2 was measured according to the first and second evaluation criteria presented in Section 6.1. The experiments have

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

Table 2. *Evaluation of unlabeled best parse trees (BPT) built from the XTAG e-tree sequences with the highest correctness ratios*

	BPT		Upper bound	
Max. Sent. Len.	13	18	13	18
Precision	47.97	45.36	54.95	54.78
Recall	78.29	73.95	89.02	88.80
F_1-score	59.49	56.23	67.96	67.76

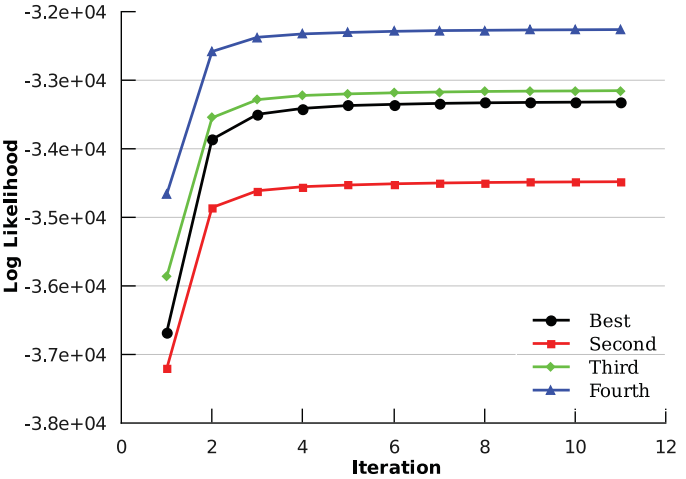


Fig. 8. (Colour online) Logarithmic scale of likelihood of observing MICA elementary tree sequences. Each curve is related to a series of MICA elementary tree sequences including most probable, second most probable, and so on.

been performed on four series of observation sequences assigned to the sentences described in Table 1. The first series contains the most probable MICA elementary tree sequences, the second contains the second most probable MICA elementary tree sequences, and so on.

Regarding the first criterion, the quality of an HMM can be evaluated through the likelihood of observing the observation sequences. Figure 8 represents the logarithm of likelihood of observing each series of MICA elementary tree sequences, given HMM_{mx} resulting from each iteration of the HMM initializing algorithm.

As shown, after just four steps, the algorithm converges to a stable state after which the rate of increase falls rapidly. The ascending behavior of the curves shows that the HMM resulting from each iteration of the initializing algorithm is more likely than its predecessors to model the observation sequences. It also confirms that the HMM can satisfactorily model all series of observation sequences. Nevertheless, the figure shows that the results does not preserve the order of the series of observation sequences. For instance, the first category, containing the MICA elementary tree sequences with the highest weights, is placed below the third and fourth series.

The second evaluation criterion deals with the amount of entropy involved in the HMM parameters for tracking the observation sequences. Figure 9 represents the

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

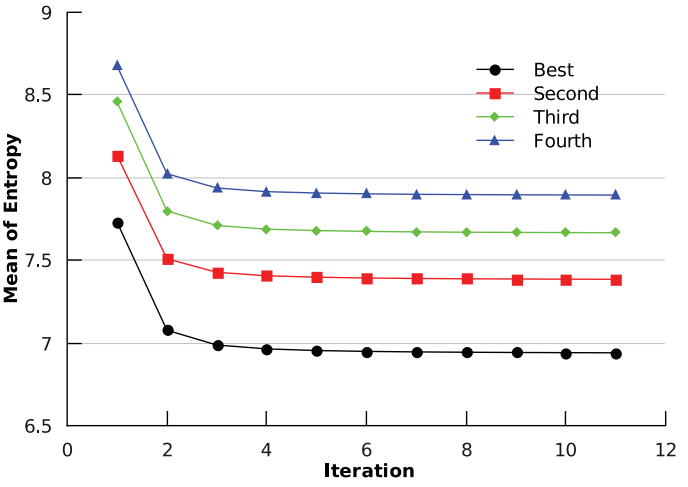


Fig. 9. (Colour online) The mean value of entropy of the distribution of all XTAG elementary tree sequences, given a MICA elementary tree sequences.

mean value of the entropy of distribution of all XTAG elementary tree sequences given the MICA elementary tree sequences and the earlier HMM resulting from the initializing algorithm. In this figure, the curves are related to the different series of MICA elementary tree sequences with respect to the probability assigned to them by the MICA parser. As shown, unlike the first criterion, this criterion maintains the order of the series of observation sequences.

It can clearly be seen that the HMM resulting from the last iteration is more certain to model the MICA elementary tree sequences than the HMMs resulting from the previous iterations. The descending behavior of the curves is evidence for this claim that the distribution of the MICA elementary trees at each hidden state of the HMM is being skewed toward the most likely MICA elementary trees. This property would make the HMM more powerful than HMMs in which observation sequences are uniformly assigned to the hidden states (Johnson 2007). In addition, unlike the previous criterion, which violates the order of the observation series, this criterion shows that the level of uncertainty regarding the best elementary tree sequences is less than for the other sequences. This means that the observation sequences with the highest weights can be modeled more accurately by the HMM. This behavior of the model is also observable for the other series of observation sequences.

6.2 Experiment descriptions

Table 3 describes three different subsets of the WSJ corpus (Marcus *et al.* 1993), *DTS-CSL*, *DTS-MED*, and *DTS-LNG*, that are used for evaluating the proposed LTAG mapping model. In order to compare the LTAG-mapping model with the previous work of Basirat and Faili (2013), we have used the test set *DTS-CSL* which contains a set of short sentences just like the sentences used by Basirat and Faili (2013) in evaluating their LTAG-mapping model. This test set includes around

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

Table 3. *Three datasets used in the evaluation of the LTAG-mapping model.*

	Sents.	Avg. Len.	Max. Len.	Words	*X-P*
DTS-CSL	197	10.0	15	1,029	90.85%
DTS-MED	219	16.8	25	3,698	87.02%
DTS-LNG	1163	18.1	25	21,154	88.99%

200 sentences with an average length of 10 words. The *DTS-MED* and *DTS-LNG*, however, contain longer sentences, which allows us to test the behavior of the model when facing long sentences. Column *X-P* shows the percentage of sentences in each of these test sets that can be parsed in the XTAG grammar.[5]

In order to provide a complete evaluation of different aspects of the LTAG-mapping model, three kinds of evaluation criteria have been proposed:

- part-of-speech agreement
- grammar compatibility
- parsing quality

The first two criteria are exactly like the error-detection heuristics introduced in Section 5.3. Hence, both of them are valid before the application of error-correction phase.

Given an input sentence $s = (w_1, \ldots, w_n)$ and its related XTAG e-tree sequence $T' = (t'_1, \ldots, t'_n)$, the value for POS agreement shows the percentage of the agreement between the POS of the anchor of t'_i and the POS of w_i. The grammar compatibility criterion, however, is founded on the *word-supertag dictionary* provided by the XTAG grammar and introduced in Section 5.3. This criterion measures the percentage of XTAG elementary trees t'_i that are correctly assigned to their related word w_i, with reference to the *word-supertag dictionary*. In this criterion, the XTAG elementary tree t'_i assigned to word w_i is considered correct, if $t'_i \in E_w$. Since neither the *POS agreement* nor the *grammar compatibility* criteria are guaranteed to find all of the erroneous elementary trees, their results can only be considered as an estimation of the accuracy of LTAG-mapping model.

The *parsing quality* criterion deals with the application of the LTAG-mapping model in parsing, as mentioned in Section 4. Due to the critical role of parsing in many natural language processing tasks, this criterion is expected to provide some useful information about the usability of LTAG-mapping model in other applications. To this end, three kind of evaluation criteria have been proposed to measure both the performance and the quality of the model when it is used alongside an LTAG parser. These criteria are as below:

- full-parse rate
- accuracy of parse tree
- parsing speed

[5] *X-P* stands for XTAG-Parseable.

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

The first criterion, *full-parse rate*, measures the percentage of the sentences that could be parsed using the LTAG-mapping model. This criterion is only concerned with the connectivity of elementary trees assigned to the words of the input sentence. In fact, it shows whether or not a parse trees could be built from a sequences of XTAG elementary trees assigned to the input sentences. The second one, *accuracy of parse tree*, quantifies the quality of parse trees with respect to *precision*, *recall*, and F_1-score of matched bracketing. Unlike the *full-parse rate*, which only checks the connectivity feature of elementary trees, *accuracy of parse tree* shows the closeness of parse tree to the desired parse tree of input sentence. Finally, *parsing speed* shows the average time required to parse a sentence. It shows how much the model could improve the parsing performance in the XTAG grammar compared to the original parser of the XTAG grammar. The results of these experiments are discussed in the next section.

6.2.1 Part-of-speech agreement

In order to compute the POS agreement of the model, we need to annotate the test sets with their desired POS tags and compare them with the POS tags defined by XTAG elementary trees. To this end, the statistical POS tagger introduced by Ratnaparkhi (1996) was used. The tagger uses a tag set which consisting of 45 POS tags defined in the Penn Treebank project. In comparison with the small tag set of the XTAG grammar, which contains 12 POS tags, the relatively large number of POS tags used by the tagger makes it difficult to find the corresponding POS tags in each tag set. This problem is due to the fact that the Penn POS tags are more fine-grained than those defined in the XTAG grammar. In fact, most of the POS tags defined in the XTAG grammar can somehow be considered abstractions of a subset of POS tags used in the Penn tag set. For instance, the POS tag *N* in the XTAG grammar can be considered an abstraction of all the POS tags *NN*, *NNS*, *NNPS*, and *NNP* in the Penn tag set.

The main obstacle to finding the corresponding POS tags, however, is the difference between the *definition* of the POS tags provided by each tag set. As an example, the word *to* in XTAG grammar can be tagged as *CONJ* (conjunction), *P* (preposition), or *V* (verb), but in the Penn POS tag set, it has its own tag *TO* regardless of whether it is a preposition or an infinitive marker. The other Penn POS tags that can be mapped onto different XTAG POS tags are *IN*, *WP*, and ‘,’. Table 4 summarizes the relationship between the POS tags in each of these tag sets.

Table 5 shows the amount of POS tag agreement for each of the test sets, with respect to the POS relationship mentioned in Table 4. The results are related to the output of the HMM-based LTAG-mapping phase, before applying the error-correction phase. In comparison with the values reported in Basirat and Faili (2013), the table shows a somewhat smaller value for POS agreement for the DTS-CSL, 92.50% as against 94.84%. This does not however convincingly demonstrate the superiority of the LTAG-mapping model proposed in Basirat and Faili (2013) over this other work. This is mainly due to the fact that the POS agreement provides only an estimation of the true quality of the system, and the higher values for

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

Table 4. *The relationship between the Penn and XTAG part-of-speech tags*

XTAG POS tag	Penn POS tag
A	JJ JJR JJS
Ad	RB WRB RBR RBS
Cmp	IN
Conj	CC IN TO
D	PRP$ WP$ DT CD PDT
G	POS
I	UH
N	N NNS NNPS NNP SYM EX FW WDT PRP WP
P	IN TO
PL	RP
Punct	, . $:) (# “ ” \ -RRB- -LRB- LS
V	VBP VBG VBD VBZ VB VBN MD TO

Table 5. *The results for POS agreement*

	DTS-CSL	DTS-MED	DTS-LNG
POS tag agreement	92.50%	88.66%	80.61%
Average errors per sentence	0.78	1.91	3.56
p-value of significance test	0.01	0.03	0.02

this criterion do not necessarily confirm the high quality of the sequences. A better criterion is the *parsing ability* of the models discussed in Section 6.2.3.

The second row of Table 5 shows the average number of POS errors per sentence. The third row shows the *p-value* of the significance test for the null hypothesis claiming that the reported averages are true. The table shows that number of errors in the test sets containing the longer sentences is higher than that of the test sets with short sentences. This is because the errors mostly occur in the most frequently used POS tags. Figure 10 shows the distribution of POS agreement errors over the desired POS tags of the words. As shown, most of the errors occur in the syntax categories *nouns* (NN), *adjectives* (JJ), and *prepositions* or *subordinating conjunctions* (IN), which are considered the most frequently used POS tags.

Figure 11 represents the probability of POS agreement errors given the desired POS of the words. As shown, the distribution of POS errors does not follow a regular pattern in the different test sets. For instance, despite the fact that *coordinating conjunctions* are the most probable sources of error in the DTS-CSL, they are ranked as the fourth most probable sources of error in the DTS-LNG after the syntax categories *RB*, *JJ*, and *IN*. So, it can be concluded that the HMM-based LTAG-mapping does not show a certain and predictable response to the syntax categories of the words. This failure of the model is mainly caused by the fact that the model does not provide any clear way to benefit from the POS tags of the words for assigning the XTAG elementary trees to words. The only syntactic information the model uses for this assignment is the words themselves, and their related MICA

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

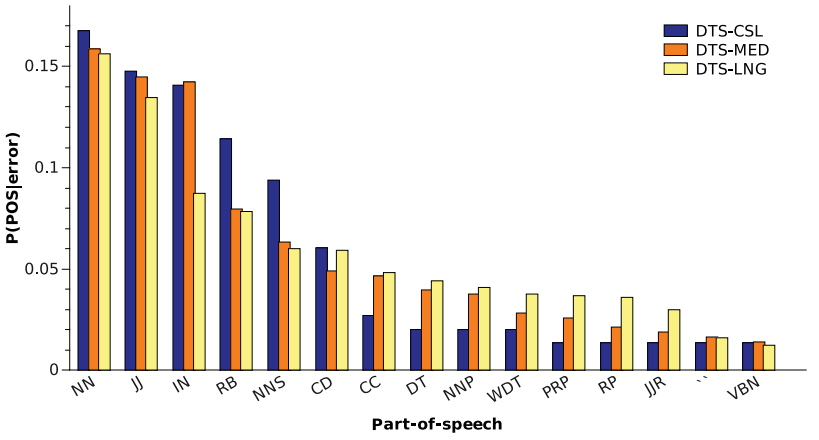


Fig. 10. (Colour online) Distribution of POS agreement error over the POS tags.

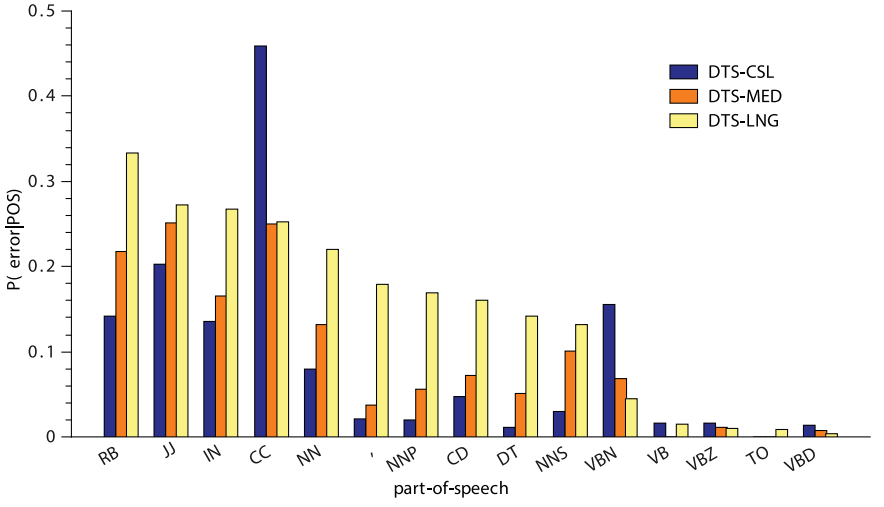


Fig. 11. (Colour online) The probability of part-of-speech agreement errors given the desired POS tags.

elementary trees. This weakness of the model, however, is compensated in the error-correction phase in which the probability distribution of POS tags is used to detect and correct the erroneous XTAG elementary trees.

6.2.2 *Grammar compatibility*

The grammar compatibility criterion measures the degree of compatibility between the output of the HMM-based LTAG-mapping phase and the *word-supertag dictionary* provided by the XTAG grammar. According to this criterion, an XTAG elementary tree e assigned to word w is assumed to be correct if e is a member of the set of XTAG elementary trees that the dictionary defines for word w, T_w. This criterion provides an upper estimate of the model's accuracy and cannot be considered an accurate metric for evaluation of the model. This is mainly because

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

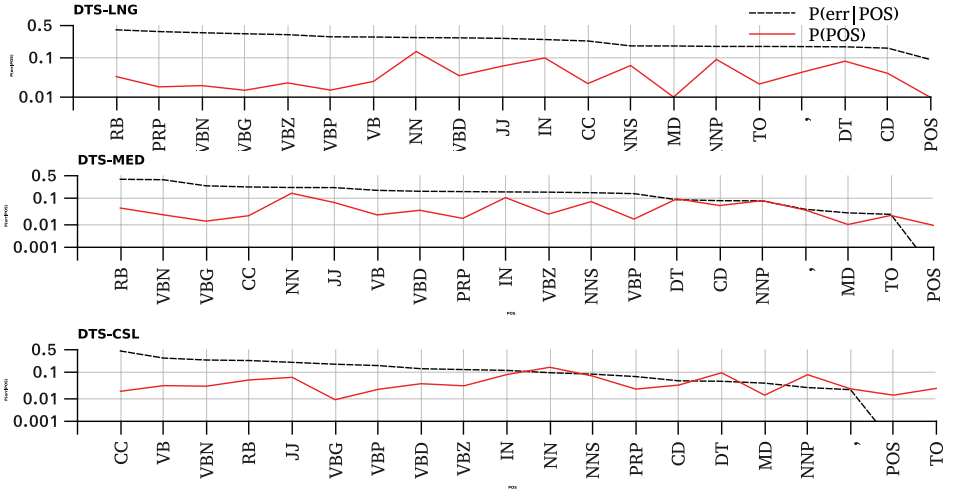


Fig. 12. (Colour online) The probability of grammar compatibility errors given part-of-speech tag of the words.

the dictionary only indicates whether or not an elementary tree can be assigned to a word, regardless of the syntactic environment. This problem can be more crucial for words that can be placed in multiple syntactic environments, such as *verbs*. In fact, it can be expected that the larger the size of the T_w is, the lower the amount of the reliability of the criterion will be for w.

Figure 12 represents the probability of occurrence of grammar compatibility errors for the 20 most frequently observed POS tags, together with the probability of observing the POS tags. The graphs are related to the different test sets described in Table 3, and the POS tags are arranged in ascending order of probability of generating error.

As shown in the figure, the most probable sources of error are the POS tags with relatively low frequencies of occurrence. For instance the four most probable sources of error in DTS-LNG are *adverbs* (RB), *personal pronouns* (PRP), *past participle verbs* (VBN), and *gerund verbs* (VBG), which have frequencies of occurrence of 0.03, 0.01, 0.02, and 0.01, respectively. This confirms the success of the HMM-based LTAG-mapping phase in annotating most of the words in the test sets with the XTAG elementary trees, with reference to the grammar compatibility criterion.

Although the distribution of grammar compatibility errors is not biased toward the most frequently used words, the nature of the errors can make it difficult for the error-correction process to find a proper replacement for the erroneous elementary trees. This is mainly because the errors mostly occurred in the words associated with a large number of elementary trees in the grammar dictionary. Table 6 provides some information about the number of single-anchor elementary trees in the dictionary used by the XTAG grammar. According to this table, one of the most probable sources of error, verbs, can be associated on average with around 45 elementary

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

Table 6. *Distribution of part-of-speech tags in XTAG word-supertag dictionary*

POS	Num words	Sum e-trees	Avg e-trees per word
A	8,714	145,803	16.73
Ad	1,019	5,241	5.14
Cmp	4	4	1.00
Conj	12	89	7.42
D	73	138	1.89
G	2	4	2.00
I	28	29	1.04
N	27,261	201,347	7.39
P	98	1,706	17.41
PL	0	0	0.00
Punct	8	35	4.38
V	17,344	788,306	45.45

Table 7. *The results of grammar compatibility*

	DTS-CSL	DTS-MED	DTS-LNG
Grammar compatibility	89.52%	84.12%	74.59%
Average errors per sentence	1.08	2.68	4.65
p-value of significance test	0.04	0.02	0.03

trees regardless of the syntactic environment. The same situation can be observed for the nouns and adjectives.

Table 7 summarizes the results achieved by the grammar compatibility evaluation before proceeding to the error-correction phase. The first row of this table shows the amount of grammar compatibility over all of the words observed in the test sets. The second and third rows respectively represent the average number of grammar compatibility errors per sentence and the *p-value* of the significance test for the null hypothesis claiming the reported averages are true. As shown, the degree of grammar compatibility depends on the length of the sentences.

6.2.3 Parsing quality

The quality of parse trees is studied based on three criteria introduced in Section 6.2, *full-parse rate*, *accuracy of parse tree*, and *parsing speed*.

The *full-parse rate* of the LTAG-mapping model refers to the percentage of the XTAG elementary tree sequences that can result in a parse tree for their sentences. In fact, the full-parse rate measures the amount of consistency between the elementary trees assigned to words of a sentence with respect to the tree attachment operations defined in LTAG formalism, *substitution* and *adjunction*. Unlike the POS agreement and grammar compatibility criteria, which assess the elementary trees *individually*, this criterion attempts to assess the quality of a *sequence* of elementary trees.

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

Table 8. *The percentage of parsed sentences*

	DTS-CSL		DTS-MED		DTS-LNG	
n	All	*X-P*	All	*X-P*	All	*X-P*
0	73.60	81.00	58.00	66.65	42.56	47.82
5	73.60	81.00	70.77	81.35	69.17	77.72
10	85.78	94.41	81.27	93.42	70.38	79.08
15	86.29	94.97	82.64	95.00	70.58	79.31
20	86.29	94.97	83.10	96.42	80.00	89.88
40	86.29	94.97	83.48	96.01	80.33	90.25
Model-CSL	56.82	61.73	28.97	35.66	24.23	27,22
Upper bound	90.85	100	87.02	100	88.99	100

Table 8 represents the *full-parse rate* of the test sets with respect to parameter *n* of the *error-correction* phase. As mentioned before, each of the test sets contains some sentences that cannot be parsed in the XTAG grammar (see Table 3). The column *X-P* represents the full-parse rate of the sentences that can be parsed in the XTAG grammar. The column *All*, however, represents the full-parse rates regardless of whether the sentences can be parsed in the XTAG grammar or not. The first row of the table, where *n* is 0, shows the values for full-parse rate when the error-correction phase is completely bypassed. In this case, output of the HMM-based LTAG-mapping phase would be directly passed onto the LTAG parser. Hence, it would properly measure the quality of the HMM-based LTAG-mapping phase. In order to compare our model with the previous LTAG-mapping, we have also computed the full parse rate values of the LTAG-mapping model introduced in Basirat and Faili (2013) (row *Model-CSL*). The *upper bound* has also been computed as the percentage of sentences that can be parsed in the XTAG grammar using the original XTAG parser.

In an overall view, in comparison with the values for Model-CSL, Table 8 shows dramatic improvements in the LTAG-mapping model. This can be seen in both the HMM-based LTAG-mapping phase and the error-correction phase. Hence, it can be concluded that not only the error-correction model, but also the HMM training method could be helpful for this success.

The increase of the full parse rate shown in Table 8 can be considered as evidence for the success of the error-correction phase in rectifying the output of the HMM-based LTAG-mapping phase. As shown, the error-correction phase could significantly compensate the main weakness of the HMM-based LTAG-mapping phase in processing the long sentences. Despite the large differences between the values related to the HMM-based LTAG mapping (first row), the table shows similar results for the test sets after the error-correction phase. In addition, it shows a direct relationship between the parameter *n* of the error correction and number of parsed sentences.

The accuracy of parse trees is examined with respect to the gold bracketing available in the Penn Treebank. The phrase structure bracketing produced by the

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

XTAG-based parse trees has been compared with the bracketing in the treebank according to the *PARSEVAL* evaluation technique (Abney *et al.* 1991). This method is based on the *precision* and *recall* of the matched brackets. The precision and recall of matched brackets are defined as the total number of bracketing matches M with respect to number of bracketings returned by the XTAG-based parse trees X (M/X) and with respect to the number of treebank bracketings C (M/C), respectively. A merged description of these metrics is their F_1-score which is defined as $(2 \times \text{precision} \times \text{recall})/(\text{precision} + \text{recall})$. Because of the differences between the internal nodes of the elementary trees in the XTAG and MICA grammars, the parse trees have been compared in unlabeled mode.

Table 9 represents the *precision*, *recall*, and F_1-score of the BPTs of the XTAG elementary tree sequences with respect to the gold parse trees in the Penn Treebank. As mentioned in Section 6.1, among the parse trees that can be built from a sequence of elementary trees, the one with the highest F_1-score is assumed to be the BPT of that sequence. We have also computed the quality of parse trees resulting from the LTAG-mapping model introduced in Basirat and Faili (2013) (row *Model-CSL*). *Upper bound* represents the accuracy of the BPTs obtained from the original XTAG parser. Since the parser has the ability to generate *all* parse trees for a given sentence, the accuracy of the best one is considered as the upper bound for parsing in the XTAG grammar.

As shown in Table 9, the values for recall are remarkably higher than the values for precision. This is mainly because the structures of the XTAG-based parse trees are relatively larger and deeper than those of the Penn bracketing. The table also shows that the error-correction phase could successfully improve the quality of the parse trees generated from the HMM-based LTAG-mapping phase (when n is 0). However, it can be seen that increasing the value of n does not necessarily improves the quality of the parse trees, though it does increase the values for *full-parse rate* (see Table 8).

The row Model-CSL in the table shows that our model could result in better parse trees than the previous work. However, unlike the full parse rate criterion, which clearly confirms the success of both the HMM-based LTAG-mapping phase and the error-correction phase, Table 9 shows that in some cases (DTS-MED, and DTS-LNG) the parse trees obtained from the HMM-based LTAG-mapping phase are worse than the previous model. This weakness of the HMM-based LTAG-mapping phase is mitigated in the error-correction phase. So, having both the HMM-based LTAG-mapping phase and the error-correction phase, the LTAG-mapping model could outperform the previous LTAG-mapping model with respect to full parse rate and the accuracy of parse trees.

In comparison with the *upper bound*, surprisingly, it can be seen that in some cases the unlabeled parse trees generated by the LTAG-mapping model are better than the parse trees generated by the XTAG parser. For instance, in DTS-CSL, when n is 15, the F_1-score is higher than the upper bound. This shows that some of the elementary trees suggested by the LTAG-mapping model are structurally more appropriate for the input words than the set of elementary trees associated with them in the XTAG grammar. This is mainly because of the XTAG grammar's

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

Table 9. *The precision (p), recall (r), and F_1-score (f) of the best parse trees (BPT) generated from XTAG elementary tree sequences*

	DTS-CSL			DTS-MED			DTS-LNG		
n	*p*	*r*	*f*	*p*	*r*	*f*	*p*	*r*	*f*
0	55.84	87.76	68.25	52.03	83.10	63.99	51.39	83.73	63.69
5	57.89	92.07	71.09	53.95	85.83	66.25	53.32	85.86	65.79
10	58.36	92.35	71.53	54.34	85.34	66.40	53.68	86.05	66.11
15	60.10	93.63	73.21	54.55	85.49	66.61	53.11	85.12	65.41
20	58.76	92.70	71.94	54.49	86.14	66.75	52.01	83.51	64.10
40	59.18	93.53	72.49	52.87	83.69	64.80	50.23	82.01	62.30
Model-CSL	54.85	88.49	67.36	54.14	86.17	66.09	51.98	85.98	64.45
upper bound	57.92	97.45	72.65	53.08	94.51	67.98	51.28	91.22	65.65

failure to cover all of the syntactic environments in which the words can appear. Nevertheless, using the word-based model, which relies on the POS tag of the words (see (12)), the LTAG-mapping model could somewhat compensate this weakness of the grammar. In the model, by focusing on the POS tags instead of the grammar rules defined in the grammar dictionary, the model could extend the set of XTAG elementary trees that can be assigned to the word.

In this part, we study the parsing speed of the LTAG-mapping model, which is one of the main contributions of this work. Using the parsing system introduced in Section 4, the amount of time required to parse a sentence is the sum of the time required to perform the three main steps of *supertagging*, *LTAG mapping*, and *LTAG parsing*. The basic algorithms running in the first two steps are (1) the maximum entropy decoder, which is used in the supertagging step, (2) the Viterbi algorithm, and (3) the word-based decoder, which are used in the LTAG mapping step. The most important factor in the time complexity of these algorithms is the sentence length. The LTAG parsing, however, is affected by both the grammar size (i.e., number of elementary trees assigned to the words and maximum number of nodes in an elementary tree) and sentence length. Sentence length is an external factor that an input sentence imposes on the parsing system. We have no control over the sentence length. The grammar size, however, can be controlled by the parameter *n* of the error-correction phase.

Figure 13 represents the average time required to parse each sentence of the test sets with respect to the parameter *n* of the *error-correction* phase. As the figure shows, the sentences of DTS-CSL can almost be parsed in a reasonable time regardless of the values of parameter *n*. It shows that parameter *n* has little effect on the parsing speed of small sentences. On the other hand it shows large increases in the parse time of the DTS-MED and DTS-LNG, which contain longer sentences, when the value of parameter *n* is increased. This is the direct consequence of the fact that increasing the length of the sentences will also increase the probability

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

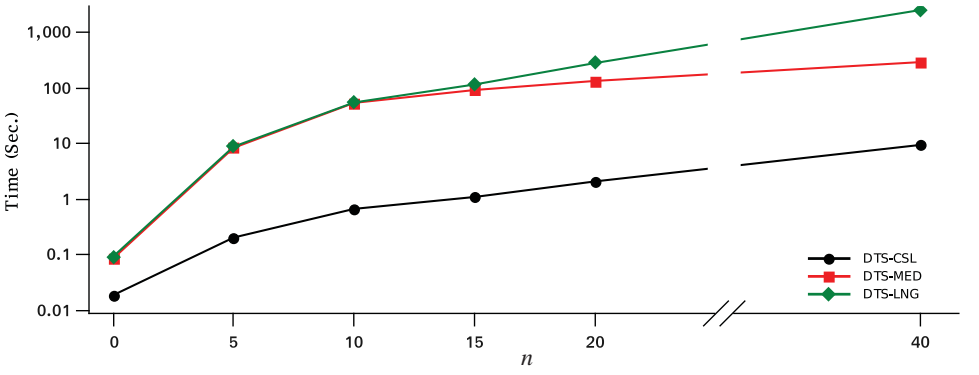


Fig. 13. The relative speed of parsing for each sentence.

of occurrence of error. This, in turn, will increase the amount of *syntactic lexical ambiguity*, the main contributor to parsing speed (Sarkar 2007).

We have also computed the average time required to parse the test sets using the parsing system introduced in Basirat and Faili (2013). In order to preserve the compatibility of this part with the previous discussions, we will refer to this system as Model-CSL. The parser cannot strictly be compared with the parsing system introduced in Section 4 because of two reasons: First, the error-correction phase is not included in the Model-CSL parsing system. The error-correction phase has negative effects on the parsing speed. In addition to the extra step that it imposes on the process of parsing, it increases the size of the grammar that is analyzed by the LTAG parser. So, it would be sensible to expect the Model-CSL parsing system to be faster than the current one, because the input words in the former model are associated with fewer elementary trees. Second, there are many sentences in the test sets that could not be parsed by the Model-CSL parsing system but could by the new model. We have shown in the Table 8 that a large portion of the test sets could not be parsed by the Model-CSL parsing system. The average lengths of these sentences in DTS-CSL, DTS-MED, and DTS-LNG are 10.65, 18.46, and 18.86, respectively.

Given all of this, it can be concluded that the parsing systems are comparable if (1) the error-correction phase is bypassed in the new model (i.e., n is 0), and (2) the test sets only include those sentences that can be parsed by both of the parsing systems. In this situation, the average time required for parsing the sentences in the test sets using the Model-CSL parsing systems is 0.01 which is equal to the new model. From the viewpoint of the end user, however, the models are compared regardless of their internal differences. In this case, the parsing speed in Model-CSL, which is 0.01 for all test sets, can be compared with the parsing speed in the new model, represented in Figure 13. It can be clearly seen that the Model-CSL is faster than new model.

Our experiments confirm that the parsing system introduced in this paper is faster than the original XTAG parser. Using the XTAG parser, the average time required for parsing the sentences in DTS-CSL, DTS-MED, and DTS-LNG is around 900

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

seconds. As shown in Table 3, the sentence pool contains 1579 sentences with a maximum length of 25 words.

To summarize, according to three evaluation criteria *full-parse rate*, *accuracy of parse trees*, and *parsing speed*, the best result of the LTAG-mapping model is related to values of n between 5 and 15. In fact, in this range, we would have a trade-off between the quality of parse trees and the time required to generate them.

7 Conclusion

Grammars can be categorized into two main groups: *hand-crafted grammars* and *data-driven grammars*. The integrating these two types of grammars might be a useful way to combine the theory-oriented and data-oriented perspectives on a language.

This paper introduces a statistical model to create such a merged description of English based on two LTAGs. The problem is formulated as a sequence-tagging problem and a *hidden Markov model* (HMM) is proposed to find the best sequence of elementary trees of a target LTAG for a given sequence of elementary trees of a source LTAG.

The model is tested on the XTAG (XTAG-Group 2001) and MICA (Bangalore *et al.* 2009) grammars in order to link the linguistic perspective of the former LTAG with the statistical nature of the latter one. Our experiments show that, using this model, around 80% of the sentences in our test sets could be parsed in the XTAG grammar. In addition, it has been shown that the model could successfully improve the efficiency of parsing in the XTAG grammar.

Although the model is founded on two LTAGs of a language, it would be interesting to apply it to different grammars of different languages. Instead of using HMM, it would also be interesting to examine other statistical approaches like *Maximum Entropy Models* and *Neural Networks* to improve the quality of the grammar mapping model.

Acknowledgments

This research was in part supported by a grant from IPM (No. CS1393-4-42). The authors would like to thank the unknown reviewers for the care and patience with which they reviewed the original manuscript. We would also like to thank Hamid Reza Taremian for the valuable contribution he has made in editing the paper.

References

Abeillé, A., Clément, L., and Toussenel, F. 2003. Building a treebank for French. In A. Abeill (ed.), *Treebanks*, pp. 165–187. Text, Speech and Language Technology, vol. 20. Netherlands: Springer.

Abney, S., Flickenger, S., Gdaniec, C., Grishman, C., Harrison, P., Hindle, D., Ingria, R., Jelinek, F., Klavans, J., Liberman, M., Marcus, M., Roukos, S., Santorini, B., & Strzalkowski, T. 1991. Procedure for quantitatively comparing the syntactic coverage of English grammars. In *Proceedings of the Workshop on Speech and Natural Language*, *ACL 1991*, Association for Computational Linguistics. California, USA, pp. 306–311

Bangalore, S., and Joshi, A. 1999. Supertagging: an approach to almost parsing. *Computational Linguistics* **25**(2): 237–266.

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

Bangalore, S., Doran, C., Hockey, B., and Joshi, A. 1996 (August). An approach to robust partial parsing and evaluation metrics. In *Proceedings of the 8th European Summer School In Logic, Language and Information*, Prague, The Czech Republic, pp. 70–82

Bangalore, S., Haffner, P., and Emami, G. 2005. Factoring global inference by enriching local representations. Technical Report, AT&T Labs - Research.

Bangalore, S., Boulllier, P., Nasr, A., Rambow, O., and Sagot, B. 2009. MICA: a probabilistic dependency parser based on tree insertion grammars. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, NAACL-Short 2009. Association for Computational Linguistics. Boulder, Colorado, pp. 185–188

Basirat, A., and Faili, H. 2011. Constructing linguistically motivated structures from statistical grammars. In *Proceedings of International Conference on Recent Advances in Natural Language Processing*, RANLP 2011. Association for Computational Linguistics. Hissar, Bulgaria, pp. 63–96.

Basirat, A., and Faili, H. 2013. Bridge the gap between statistical and hand-crafted grammars. *Computer Speech and Language* **27**(5): 1085–1104.

Baum, L. E., Petrie, T., Soules, G., and Weiss, N. 1970. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *The Annals of Mathematical Statistics* **41**(1): 164–171.

Bonfante, G., Guillaume, B., and Perrier, G. 2004. Polarization and abstraction of grammatical formalisms as methods for lexical disambiguation. In *Proceedings of the 20th International Conference on Computational Linguistics*, COLING 2004. Association for Computational Linguistics. Geneva, Switzerland, pp. 303–309.

Boullier, P., and Deschamp, P. 1988. Le systeme SYNTAX^{TM} manuel d'utilisation et de mise en oeuvre sous UNIX^{TM}. http://syntax.gforge.inria.fr/syntax3.8-manual.pdf.

Chandrasekar, R., and Bangalore S. 1997. Gleaning information from the web: using syntax to filter out irrelevant information. In *Proceedings of AAAI Symposium on NLP for the World Wide Web*, California, USA, pp. 27–34.

Chen, J. 2001. *Toward efficient statistical parsing using lexicalized grammatical information*, Ph.D. thesis, University of Delaware.

Chiang, D. 2000. Statistical parsing with an automatically-extracted tree adjoining grammar. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL 2000. Association for Computational Linguistics. Hong Kong, pp. 456–463

Chiang, D., and Rambow, O. 2006. The hidden TAG model: synchronous grammars for parsing resource-poor languages. In *Proceedings of the Eighth International Workshop on Tree Adjoining Grammar and Related Formalisms*, Association for Computational Linguistics. Sydney, Australia, pp. 1–8.

Chomsky, N. 1959. On certain formal properties of grammars. *Information and control* **2**(2): 137–167.

Dang, H. T., Kipper, K., and Palmer, M. 2000. Integrating compositional semantics into a verb lexicon. In *Proceedings of the 18th Conference on Computational Linguistics*, COLING 2000, Association for Computational Linguistics. Saarbrücken, Germany. vol. 2, pp. 1011–1015

DeNeefe, S., and Knight, K. 2009. Synchronous tree adjoining machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, EMNLP 2009, Association for Computational Linguistics. Singapore. vol. 2, pp. 727–736.

Eddy, B., Bental, D., and Cawsey, A. 2001. An algorithm for efficiently generating summary paragraphs using tree-adjoining grammar. In *Proceedings of the 8th European workshop on Natural Language Generation*, EWNLG 2001, Association for Computational Linguistics. Toulouse, France. vol. 8, pp. 1–8.

Forbes, K., Miltsakaki, E., Prasad, R., Sarkar, A., Joshi, A. K., and Webber, B. 2003. D-LTAG system: discourse parsing with a lexicalized tree-adjoining grammar. *Journal of Logic, Language and Information* **12**(3): 261–279.

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

Frank, R. 2004. *Phrase Structure Composition and Syntactic Dependencies*, vol. 38, The MIT Press.

Gardent, C., and Kallmeyer, L. 2003. Semantic construction in Feature-based TAG. In *Proceedings of the 10th Conference on European Chapter of the Association for Computational Linguistics*, EACL 2003, Association for Computational Linguistics. Budapest, Hungary. vol. 1, pp. 123–130.

Goodman, J. 1996. Parsing algorithms and metrics. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, ACL 1996. Association for Computational Linguistics. California, USA, pp. 177–183.

Han, C., Han, N., Ko, E., and Palmer, M. 2002. Development and evaluation of a Korean treebank and its application to NLP. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, LREC 2002. European Language Resources Association (ELRA). Canary Islands, Spain, pp. 1635–1642.

Hemphill, C. T., Godfrey, J. J., and Doddington, G. R. 1990. The ATIS spoken language systems pilot corpus. In *Proceedings of the Workshop on Speech and Natural Language*. HLT 1990. Association for Computational Linguistics. Pennsylvania, USA, pp. 96–101.

Hernando, D., Crespi, V., and Cybenko, G. 2005. Efficient computation of the hidden Markov model entropy for a given observation sequence. *Information Theory, IEEE Transactions* **51**(7): 2681–2685.

Johnson, M. 2007. Why doesn't EM find good HMM POS-taggers? In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL 2007. Association for Computational Linguistics. Prague, Czech Republic, pp. 296–305.

Joshi, A. K., and Schabes, Y. 1997. Tree-adjoining grammars. In G. Rozenberg and A. Salomaa (eds.) *Handbook of formal languages*, 69–123. Springer Berlin Heidelberg.

Joshi, A. K., and Vijay-Shanker, K. 2001. Compositional semantics with lexicalized tree-adjoining grammar (LTAG): how much underspecification is necessary? In H. Bunt et al. (eds.), *Computing Meaning*, . Springer Netherlands, vol. 77, pp. 147–163.

Joshi, A. K. 1985. Tree adjoining grammars: how much context-sensitivity is necessary for characterizing structural descriptions? In: David R. Dowty et al. (eds.) *Natural language parsing: Psychological, computational and theoretical perspectives*, 206–250. Cambridge: Cambridge University Press.

Kaeshammer, M. 2012. *A German Treebank and Lexicon for Tree-Adjoining Grammars*, Master thesis. Saarland University.

Kallmeyer, L., Lichte, T., Maier, W., Parmentier, Y., and Dellert, J. 2008. Developing a TT-MCTAG for German with an RCG-based parser. In *Proceedings of the 6th International Conference on Language Resources and Evaluation*, LREC 2008. European Language Resources Association (ELRA). Marrakech, Morocco, pp. 782–789.

Kaplan, R. M., and Bresnan, J. 1982. Lexical-functional grammar: a formal system for grammatical representation. *The mental representation of grammatical relations*, pp. 173–281, Cambridge, MA: MIT.

Kipper, K., Dang, H., and Palmer, M. 2000. Class-based construction of a verb lexicon. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI-2000) and Twelfth Conference on Innovative Applications of Artificial Intelligence*, American Association for Artificial Intelligence (AAAI) Press. Texas, USA, pp. 691–696.

Kroch, A. S., and Joshi, A. K. 1985. The linguistic relevance of tree adjoining grammars. Technical Report MS-CIS-85-16. Department of Computer and Information Science, University of Pennsylvania.

Lee, L. 2002. Fast context-free grammar parsing requires fast boolean matrix multiplication. *Journal of the ACM (JACM)* **49**(1): 1–15.

Ma, W.-Y., and McKeown, K. 2012. Detecting and correcting syntactic errors in machine translation using feature-based lexicalized tree adjoining grammars. *International Journal of Computational Linguistics and Chinese Language Processing (IJCLCLP)* **17**(4): 1–14.

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. 1993. Building a arge annotated corpus of English: the penn treebank. *Computational Linguistics - Special issue on using large corpora* **19**(2): 313–330.

Park, J. 2006. Extraction of tree adjoining grammars from a treebank for Korean. In *Proceedings of the 21st International Conference on computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, ACL 2006. Association for Computational Linguistics. Sydney, Australia, pp. 73–78.

Pollard, C., and Sag, I. A. 1994. *Head-Driven Phrase Structure Grammar*, University of Chicago Press.

Ratnaparkhi, A. 1996. A maximum entropy model for Part-Of-Speech tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP 1996, Association for Computational Linguistics. Philadelphia, USA. vol. 1, pp. 133–142.

Ryant, N., and Kipper, K. 2004. Assigning XTAG trees to VerbNet. In *Proceedings of the 7th International Workshop on Tree Adjoining Grammar and Related Formalisms*, TAG+ 7. Vancouver, BC, CA, pp. 194–198.

Sarkar, A. 2007. Combining Supertagging and Lexicalized Tree-Adjoining Grammar Parsing. *Complexity of Lexical Descriptions and its Relevance to Natural Language Processing: A Supertagging Approach*. MIT Press, Boston, MA, USA.

Satta, G. 1994. Tree-adjoining grammar parsing and boolean matrix multiplication. *Computational Linguistics* **20**(2): 173–191.

Schabes, Y., and Joshi, A. K. 1988. An Earley-type parsing algorithm for tree adjoining grammars. In *Proceedings of the 26th Annual Meeting on Association for Computational Linguistics*., ACL 1988. Association for Computational Linguistics. Buffalo, New York, USA, pp. 258–269.

Schabes, Y., and Waters, R. C. 1995. Tree insertion grammar: a cubic-time, parsable formalism that lexicalizes context-free grammar without changing the trees produced. *Computational Linguistics* **21**(4): 479–513.

Shieber, S. M., and Schabes, Y. 1990. Synchronous tree-adjoining grammars. In *Proceedings of the 13th Conference on Computational linguistics*, COLING 1990. Association for Computational Linguistics. University of Helsinki, Finland. vol. 3, pp. 253–258.

Steedman, M. 2000. *The Syntactic Process*. Cambridge, MA, USA: MIT Press.

Stone, M., and Doran, C. 1997. Sentence planning as description using tree adjoining grammar. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, EACL 1998. Association for Computational Linguistics. Madrid, Spain, pp. 198–205.

Xia, F. 1999. Extracting tree adjoining grammars from bracketed corpora. In *Proceedings of the 5th Natural Language Processing Pacific Rim Symposium*. NLPRS 1999. Beijing, China, pp. 398–403.

Xia, F. 2001. *Automatic grammar generation from two different perspectives*, Ph.D. thesis, University of Pennsylvania.

Xia, F., and Palmer, M. 2000. Evaluating the Coverage of LTAGs on Annotated Corpora. In *Proceedings of LREC satellite workshop Using Evaluation within HLT Programs: Results and Trends*, Athens, Greece, pp. 1–6.

XTAG-Group. 2001. A Lexicalized Tree Adjoining Grammar for English. Technical report IRCS 01–03, Institute for Research in Cognitive Science, University of Pennsylvania.

https://doi.org/10.1017/S1351324915000017 Published online by Cambridge University Press