Seamless Tree Binarization for Interactive Predictive Parsing

Ricardo Sánchez-Sáez[†], *Luis A. Leiva*[‡], *Joan-Andreu Sánchez*[†], *José-Miguel Benedí*[†]

Instituto Tecnológico de Informática
Universidad Politécnica de Valencia, Spain
{rsanchez,luileito,jandreu,jbenedi}@{[†]dsic,[‡]iti}.upv.es

Abstract

This paper introduces a seamless method for tree binarization/debinarization that is employed within the Interactive Predictive Parsing framework for tree annotation. This novel method allows that, while the human annotator verifies and corrects standard non-binary trees, the parse engine can work with parsing algorithms that process and produce binary trees, such as a CYK-Viterbi based parser.

Within the Interactive Predictive Parsing framework the user is tightly integrated into the interactive parsing system, in contrast with the traditional post-editing approach. User feedback for tree correction and validation is provided by means of natural mouse gestures and keyboard strokes.

Index Terms: parsing, interactive predictive parsing, syntactic tree annotation, tree binarization

1. Introduction

Probabilistic parsing is a fundamental problem in Computational Linguistics. Probabilistic parsing has been greatly benefited in the past from the availability of annotated corpora. Perfectly annotated parsing trees are used in the training of new automatic parsing systems, and are needed for parser validation and experimentation. Therefore, there is a pressing need in efficiently constructing new perfectly annotated corpora.

The trees contained in these corpora must be manually annotated: either creating them from scratch, or based on automatically obtained error-prone parse trees. This results in a laborious and time-consuming task.

Several tools exist that can aid in easing the work of human annotators. Some examples are the TreeBanker [2], TrEd[1] or eBonsai [3], for structural annotation; or DepAnn [4], for dependency style annotation. The well-known Penn Treebank itself was annotated using automatically obtained basic skeletal parses, and an aid tool was used to finish the annotation of the parse trees [6].

The problem with using these tools for treebank creation is that they all typically introduce a two-step workflow: first, a chosen system generates the best tree for the sentence being annotated, and then the human annotator has to verify it and amend the errors within the proposed parse tree. This paradigm is rather inefficient and uncomfortable for the human annotator.

Recent work has introduced a new type of Web-based interactive predictive annotation tool, the Interactive Predictive Parsing Tree Annotator (or IPP-Ann) [10]. This tool follows the Interactive Predictive Parsing (IPP) paradigm, whose novelty is that it fully integrates the human annotator into the parsing loop, making him part of the system. The annotator interacts in real time with the IPP engine, and the system uses the readily available user feedback to make predictions about the parts of the tree that have not been validated by the corrector.

Experiments carried out to simulate user interaction with the IPP framework suggest figures ranging from 42% to 46% of effort saving compared to manually post-editing the trees without an interactive system, both for English [11] and Spanish [9] sentence annotation. Additionally, this kind of man-machine integration presents yet unexplored opportunities, such as the scenario in which the parsing system adapts its models, incorporating the new ground-truth data provided by the user.

IPP-Ann is a implementation of the IPP framework, and takes the form of a decoupled annotation system consisting in a parse engine and a Web client working together. IPP-Ann can be used online at `http://cat.iti.upv.es/ipp/`.

The parsing subsystem of IPP-Ann currently uses a CYK-Viterbi based parsing algorithm which works with a Probabilistic Context Free Grammar (PCFG) in Chomsky Normal Form (CNF) as its model. Algorithms that use grammars in CNF can only produce and process binary trees. An automatic process for binarization/debinarization of the trees going through the parse engine is needed, so they are presented to the human annotator in an usable non-binary form.

In this paper, after reviewing the IPP theoretical framework and the IPP-Ann system, we present a novel method for seamless tree binarization/debinarization. This method allows that, while the human annotator uses IPP-Ann to modify and annotate standard non-binary trees, the parsing subsystem of the annotation tool is able to internally work with parsing algorithms that process and generate binary trees, such as the CYK-Viterbi parsing algorithm. Parsing algorithms that use binary trees are widespread in the parsing world, as they are simpler to understand and more efficient.

2. Interactive Predictive Parsing Framework

In this section we review the IPP framework [11]. Interactive predictive methods have been successfully demonstrated to ease the work of transcriptors and translators in fields like Handwriting Text Transcription [8, 12] and Statistical Machine Translation [7, 13].

A tree t, associated to a string $x_{1|x|}$, is composed by substructures that are usually referred as constituents. A constituent c_{ij}^{A} is defined by the non-terminal symbol A (either a *syntactic label* or a *POS tag*) and its span ij (the starting and ending indexes which delimit the part of the input sentence encompassed by the constituent).

Here follows a general formulation for the non-interactive parsing scenario. Using a grammatical model G, the parser analyzes the input sentence $\boldsymbol{x} = \{x_1, \ldots, x_{|x|}\}$ and produces the

[1] `http://ufal.mff.cuni.cz/~pajas/tred/`

parse tree $\hat{t}$

$$\hat{t} = \arg\max_{t \in \mathcal{T}} p_G(t|\boldsymbol{x}), \quad (1)$$

where $p_G(t|\boldsymbol{x})$ is the probability of parse tree t given the input string $\boldsymbol{x}$ using model G, and $\mathcal{T}$ is the set of all possible parse trees for $\boldsymbol{x}$.

In the interactive predictive scenario, after obtaining the (probably incorrect) best tree $\hat{t}$, the user is able to individually correct any of its constituents c_{ij}^A. The system reacts to each of the corrections introduced by the human, proposing a new $\hat{t}'$ that takes into account the afore-mentioned corrections.

Within the IPP framework, the user reviews the constituents contained in the tree to assess their correctness. The action of modifying an incorrect constituent (either setting the correct span or the correct label) implicitly validates a subtree that is composed by the partially corrected constituent, all of its ancestor constituents, and all constituents whose end span is lower than the start span of the corrected constituent. We will name this subtree the validated prefix tree t_p (for analogy with the prefix sentence in the above mentioned interactive machine translation and interactive text transcription scenarios). When the user replaces the constituent c_{ij}^A with the correct one $c_{ij}^{\prime A}$, the validated prefix tree is:

$$\begin{aligned} t_p(c_{ij}^{\prime A}) = \{&c_{mn}^B \ : \ m \leq i, \ n \geq j, \\ &d(c_{mn}^B) \leq d(c_{ij}^{\prime A})\} \cup \\ \{&c_{pq}^D \ : \ p >= 1 \ , \ q < i\} \end{aligned} \quad (2)$$

with $d(c_{mn}^B)$ being the depth of constituent c_{mn}^B.

When a constituent correction is performed, the prefix tree $t_p(c_{ij}^{\prime A})$ is fixed and a new tree $\hat{t}'$ that takes into account the prefix is proposed:

$$\hat{t}' = \arg\max_{t \in \mathcal{T}} p_G(t|\boldsymbol{x}, t_p(c_{ij}^{\prime A})). \quad (3)$$

Given that we are working with context-free grammars, the only subtree that effectively needs to be recalculated is the one starting from the parent of the corrected constituent.

3. The IPP Tree Annotator

IPP-Ann has been previously adapted to parse both English (with a Penntreebank based model) [10] and Spanish text (with a UAM Treebank based model) [9], and can be accessed at `http://cat.iti.upv.es/ipp/`.

IPP-Ann can help users to efficiently annotate correct syntactic trees. For this, the user feedback (provided by means of keyboard and mouse operations) allows the system to predict new subtrees for unvalidated parts of the annotated sentence, which in turn reduces the human effort and improves annotation efficiency.

The IPP Tree Annotator uses the CAT-API library [1] as a communication backend between the server and client modules. This library allows for a clean application design, in which both the server side (the parsing engine) and the client interface (which draws the trees, captures and interprets the user feedback, and requests parsed subtrees to the server) are independent of each other. One of the features that steams from the CAT-API library is the ability for several annotators to work concurrently on the same problem-set, each in a different client computer sharing the same parsing server.

When working with IPP-Ann, the user is presented with the sentences from the selected corpus, and starts parsing them one by one. They can then make corrections in the trees and the user feedback is decoded on the client side which in turn requests subtrees to the parse engine.

Two kind of operations can be performed over constituents when using IPP-Ann: span modification (done by dragging a line from the constituent to the word that corresponds to the span's upper index), and label substitution (done by typing the correct one on its text field). Modifying the span of a constituent invalidates its label, so the server recalculates it as part of the suffix. Modifying the label of a constituent validates its span. Constituents can be deleted or inserted by adequately modifying the span of the left-neighbouring constituent. Figure 1 shows a span modification performed by a human annotator using.

Additionally, two unary-related operations can be performed over constituents: unary production insertion (done by drawing a line from the constituent node to the floating ball that appears below itself), and unary production removal (done by resetting the span of the constituent parenting the unary production).

As visual aid, when the user is about to perform an operation, the affected constituent and the prefix that will be validated are highlighted. The target span of the modified constituent is visually shown as well. When the user obtains the correctly annotated tree, they can accept it by by clicking on a new sentence.

3.1. Server side implementation

The server side of the system is a parsing engine based on a customized CYK-Viterbi parser, which uses a Probabilistic Context-Free Grammar in Chomsky Normal Form. The English grammar was obtained from sections 2 to 21 of the UPenn Treebank as a model [11], and the Spanish grammar was obtained from the first 1400 sentences of the UAM Treebank [9].

The client can send requests to the parsing server in order to obtain the best subtree for any given span of the input string. For each requested subtree, the client can either provide the root label or not. If the subtree root label is not provided, the server calculates the most probable label. The server also performs transparent tree debinarization/binarization, as presented in this work, and unary-rule expansion when communicating with the client.

3.2. Client side implementation

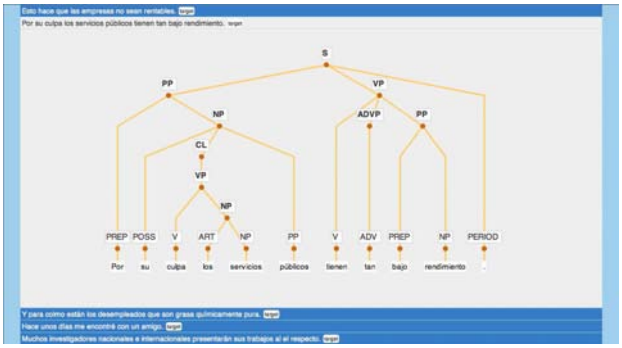
The client part is implemented in a combination of HTML, PHP and Flash ActionScript. As such, it is accessed through a Web browser, being the Flash plugin the only requisite. The hardware requirements are very low on the client side, as the parsing process is performed remotely on the server side: any computer (including netbooks) capable of running a modern Web browser is enough.

The Web client side of IPP-Ann communicates with the IPP engine through binary TCP sockets, resulting in very low response times.

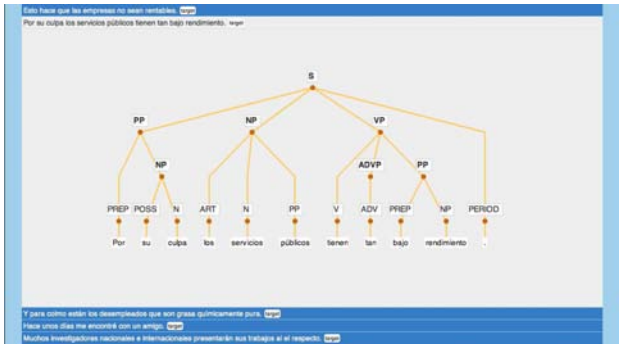
4. Transparent binarization

In this section we introduce a transparent debinarization/binarization process which is consistent with our Interactive Predictive strategy and with our parsing model.

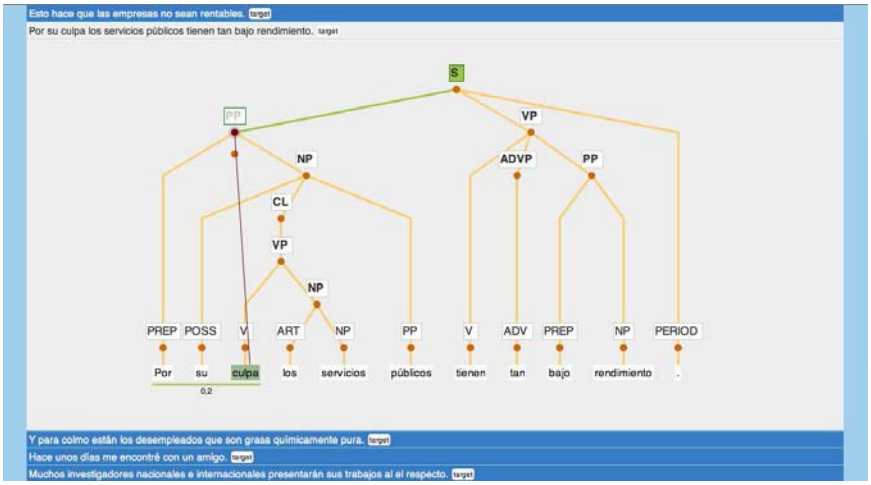
On the one hand, the CYK-Viterbi algorithm used within the parsing server works with Probabilistic Context-Free Grammars in Chomsky Normal Form. By the usage of CNF-PCFGs



(a) System: Output tree 1



(c) System: Output tree 2



(b) User: Span modification

Figure 1: Interaction example on the IPP Tree Annotation tool.

as the model to obtain syntactic trees, only binary trees can be obtained. In our case, the grammars used by the server were obtained using the Chomsky Normal Form transformation method from the Open Source Natural Language Toolkit[2] (NLTK) to calculate minimal right-factored binary grammars [5] from the corresponding regular vanilla reebank grammars.

On the other hand, manual tree annotation is generally performed using non-binary trees, as the underlying syntactic structures present in natural language sentences do not conform to binary trees. To sum up, the human annotator needs to work with non binary trees but the interactive predictive parsing algorithm uses and produces binary trees.

This fundamental necessity led us to devise and implement a seamless and automatic debinarization/binarization process for the trees produced by the parsing server. The debinarization/binarization process is performed in real time and in a completely oblivious form to the user. The human annotator working with regular non-binary trees never notices that the parsing server is internally working with binary trees.

Our seamless binarization/debinarization method employs the aforementioned Chomsky Normal Form transformation, which has been natively implemented within the parsing server. The debinarization process is performed when the algorithm calculates a new subtree, before sending it to the client. The tree binarization process is carried out when the server receives a new tree with the user corrections, before sending it to the interactive predictive parsing algorithm.

[2] http://www.nltk.org/

4.1. Linked tree structure

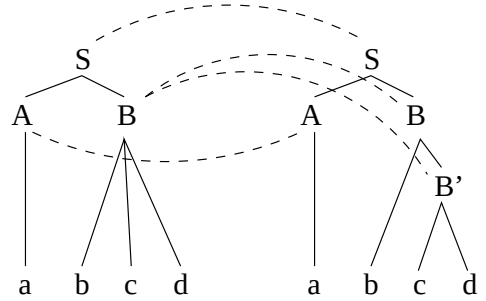
In a regular, non-binary tree, the non-terminal of each node (excluding POS tag nodes) corresponds to a syntactic label. Each represents a syntactic structure that relates to a sequence of words from the sentence being parsed. When such a tree is binarized by the CNF transformation, some new *dummy* nodes are introduced when one node has more than two descendants. These newly *introduced nodes* have non-terminals that do not carry new syntactic information by themselves. Instead, their only function is to propagate the syntactic label of the original non-binary ancestor through the structure of the binary tree.

When IPP-Ann is being employed by an annotator, the IPP algorithm needs to be informed about which tree constituent — the node with its syntactic label and span — was modified after each user interaction. Given that the user performs changes on regular trees via the client interface but the parsing algorithm works with their binarized counterparts, the binarization process must keep information about the correspondency between the introduced nodes in the binary tree and their original ancestor in the non-binary tree.

At binarization/debinarization time, for each introduced node in the binary tree, we note its corresponding ancestor node in the regular tree. Each *non-introduced node* in the binary tree

is also linked to the matching node in the regular tree. This generates a one-to-many relationship, in which each node of the regular tree relates to one or more nodes of the binary tree (either just one *non-introduced node*, or one *non-introduced* node plus several *introduced nodes*).

In order to keep the node correspondency information consistent over the binarization/debinarization process, we constructed a new tree structure which we call a *linked tree*. A linked tree consists on the binary and non-binary versions of the same tree, and the one-to-many relationships between the nodes of both trees. See Figure 2 for an example of a linked tree structure.



(a) Non−binary tree (b) Binary tree

Figure 2: Linked tree structure. Note how the *introduced node* B' in the binary tree is related to its original ancestor in the non-binary tree.

Within a linked tree, when either the binary tree or the regular tree is modified, the debinarization/binarization process operates automatically over the linked tree structure, recalculating the associated tree and updating the correspondency relationships. This action takes place either when the user modifies a constituent in the regular tree, or when the parsing algorithm recalculates a new binary subtree based on the user feedback. This method allows for the regular tree, the binary tree and their node correspondency information to remain synchronized and up-to-date at all times.

5. Conclusions and future work

We have introduced a seamless method for tree binarization/debinarization whithin the Interactive Predictive Parsing framework. This novel method allows that while the human annotator verifies and corrects standard non-binary trees, the parse engine can work with parsing algorithms that process and produce binary trees, such as the CYK-Viterbi parser.

The reviewed tool, by using a parse engine in an integrated manner, aids the user in creating correctly annotated syntactic trees. IPP-Ann greatly reduces the human effort required for this task compared to using a non-interactive automatic system.

Future work includes improvements to the client side (e.g., confidence measures as a visual aid, multimodality), as well as exploring other kinds of parsing algorithms for the server side (e.g., adaptative parsing).

6. Acknowledgements

Work partially supported by the Spanish MICINN under the MIPRCV “Consolider Ingenio 2010” (CSD2007-00018), MITTRAL (TIN2009-14633-C03-01), Prometeo (PROMETEO/2009/014) research projects, and the FPU fellowship AP2006-01363. The authors wish to thank Vicent Alabau for his invaluable help with the CAT-API library.

7. References

[1] V. Alabau, D. Ortiz, V. Romero, and J. Ocampo. A multimodal predictive-interactive application for computer assisted transcription and translation. In *Proc. of ICMI-MLMI*, pages 227–228, New York, USA, 2009. ACM.

[2] D. Carter. The TreeBanker. A tool for supervised training of parsed corpora. In *Proc. of ENVGRAM Workshop*, pages 9–15, 1997.

[3] I. Hiroshi, N. Masaki, H. Taiichi, T. Takenobu, and T. Hozumi. eBonsai: An integrated environment for annotating treebanks. In *Proc. of IJCNLP*, pages 108–113, 2005.

[4] T. Kakkonen. Depann - an annotation tool for dependency treebanks. In *Proc. of ESSLLI Student Session*, pages 214–225, Malaga, Spain, June 2006.

[5] D. Klein and C.D. Manning. Accurate unlexicalized parsing. In *Proc. of ACL*, volume 1, pages 423–430, Morristown, USA, 2003. ACL.

[6] M.P. Marcus, B. Santorini, and M.A. Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, 1994.

[7] D. Ortiz, L.A. Leiva, V. Alabau, and F. Casacuberta. Interactive machine translation using a web-based architecture. In *Proc. of IUI*, pages 423–425. Hong Kong, China, February 2010.

[8] V. Romero, L.A. Leiva, A.H. Toselli, and E. Vidal. Interactive multimodal transcription of text imagse using a web-based demo system. In *Proc. of IUI*, pages 477–478. Sanibel Island, Florida, February 2009.

[9] R. Sánchez-Sáez, L.A. Leiva, J.A. Sánchez, and J.M. Benedí. Interactive predictive parsing framework for the spanish language. In *Proc. of SEPLN*, Valencia, Spain, September 2010.

[10] R. Sánchez-Sáez, L.A. Leiva, J.A. Sánchez, and J.M. Benedí. Interactive predictive parsing using a web-based architecture. In *Proc. of NAACL-HLT*, Los Angeles, United States of America, June 2010.

[11] R. Sánchez-Sáez, J.A. Sánchez, and J.M. Benedí. Interactive predictive parsing. In *Proc. of IWPT'09*, pages 222–225, Paris, France, October 2009. ACL.

[12] A.H. Toselli, V. Romero, and E. Vidal. Computer assisted transcription of text images and multimodal interaction. In *Proc. MLMI*, volume 5237, pages 296–308. Springer, 2008.

[13] E. Vidal, F. Casacuberta, L. Rodríguez, J. Civera, and C. Martínez. Computer-assisted translation using speech recognition. *IEEE TASLP*, 14(3):941–951, 2006.