This is a preprint for personal use only. The published paper may be subject to some form of copyright.

IPP-Ann: an Interactive Tool for Probabilistic Parsing

Ricardo Sánchez-Sáez, Luis A. Leiva, Joan Andreu Sánchez and José Miguel Benedí
Instituto Tecnológico de Informática
Universidad Politécnica de Valencia
Valencia, Spain
{rsanchez,luileito,jandreu,jbenedi}@dsic.upv.es

***Abstract*—This paper introduces IPP-Ann: the Interactive Predictive Parsing Tree Annotator. IPP-Ann is an interactive tool that can be used by an expert to effortlessly annotate syntactic trees. The tool shows an initial proposed annotation tree, and then allows the user to perform individual corrections on the tree constituents. These corrections implicitly validate a prefix subtree, and IPP-Ann reacts to such feedback in real time, proposing new trees that complete the modifications made by the user.**

***Keywords*-Interactive Predictive Parsing; IPP; Tree Annotation;**

I. Introduction

IPP-Ann is an annotation tool that is based on the Interactive Predictive Parsing (IPP) framework [1]. The objective of IPP is to be employed by linguists to construct correct syntactic trees in an interactive manner with little effort.

There exist additional works by other authors within the computer aid to annotation such as [2], [3], [4]. However, we feel that the fact of being based our tool on a sound theoretical framework that truly integrates the annotator within the parsing process, sets IPP-Ann apart.

Our tool comprises a thin Web client that operates in conjunction with an IPP server (using a Probabilistic Context-Free Grammar as a model) which provides annotated tree candidates. IPP-Ann can be accessed online at `http://cat.iti.upv.es/ipp/`.

We have adapted IPP-Ann for parsing both English (with a Penntreebank based model [5]) and Spanish text (with a UAM Treebank based model [6]). Figures 1 and Figures 2 show some screenshots: the IPP-Ann welcome and sentence selection screen.

When using IPP-Ann, the user is presented with the selected corpus, and can start parsing the sentences one by one. The user chooses a sentence, and the system proposes a complete annotated tree. The user then can make corrections in the trees using the mouse and the keyboard, and the system proposes new trees, completing the user validated data.

Architecturally, the user feedback is decoded on the client side, which in turn requests the new subtrees to the parse engine.

In user simulation experiments, we have observed high amounts of effort saving when using an IPP system to annotate sentences in an error-free fashion. User effort metrics show that the percentage of corrections needed using the IPP system is much lower than the rate of needed corrections when manually post-editing the output of a completely automatic parsing system: an estimated 46% of constituent corrections could be saved by a human linguist using IPP-Ann.

II. Interactive Predictive Parsing Theoretical Framework

In this section we review the Interactive Predictive Parsing Theoretical Framework.

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 nonterminal 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 parse tree $\hat{t}$

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

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.

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. When the user replaces the constituent c_{ij}^{A} with the correct one c'^{A}_{ij}, the validated prefix

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

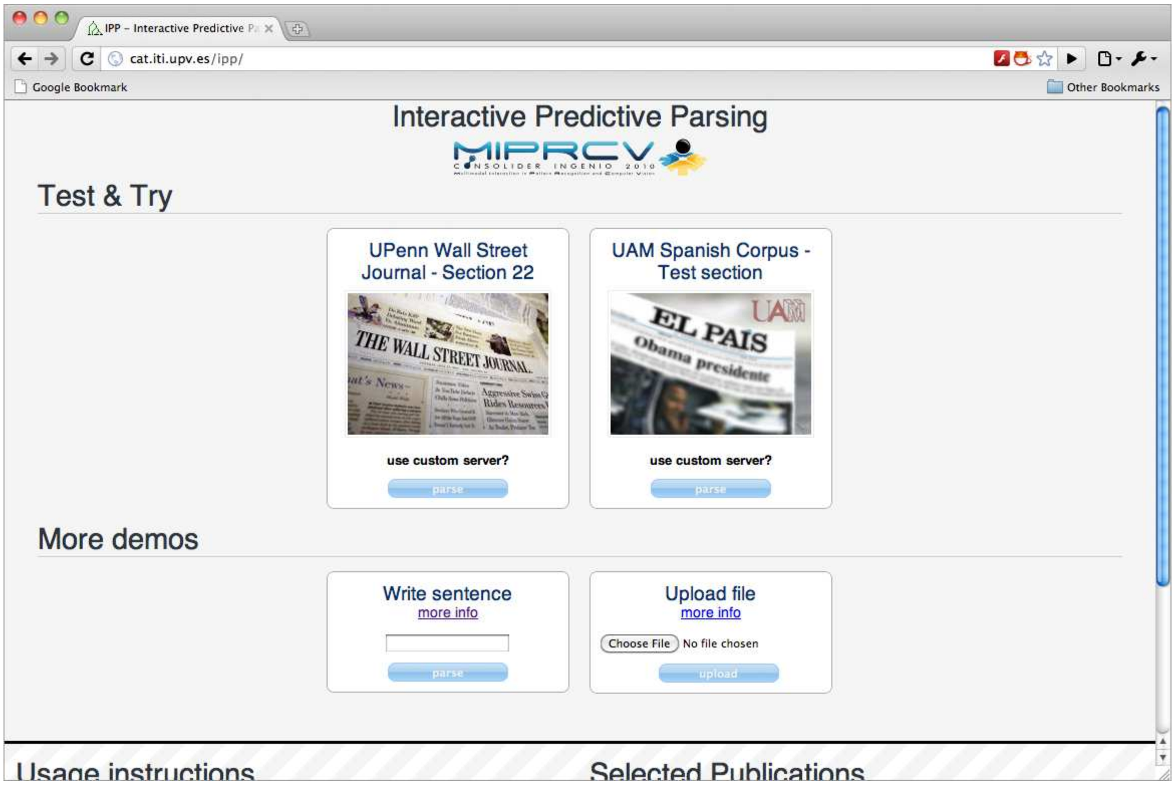


Figure 1. The IPP-Ann welcome screen.

tree is:

$$\begin{aligned} t_p(c'^A_{ij}) = \{c^B_{mn} &: m \le i,\ n \ge j, \\ d(c^B_{mn}) &\le d(c'^A_{ij})\} \cup \\ \{c^D_{pq} &: p >= 1\ ,\ q < i\} \end{aligned} \qquad (2)$$

with $d(c^B_{mn})$ being the depth of constituent c^B_{mn}.

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

$$\hat{t}' = \underset{t \in \mathcal{T}}{\arg\max}\, p_G(t|\boldsymbol{x}, t_p(c'^A_{ij})). \qquad (3)$$

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.

III. IPP-Ann Operation

The user can perform two kind of operations over constituents: span modification, and label substitution. Modifying the span of a constituent invalidates its label, so the server may recalculate and change it as part of the new tree. Modifying the label of a constituent validates its span. Constituents can be adequately inserted or deleted by modifying the span of their left-neighbouring constituents. Also, operations for manipulating unary productions is available: they can be inserted and deleted. See Figure 3 for an example of span modification.

Operations can be performed as follows:

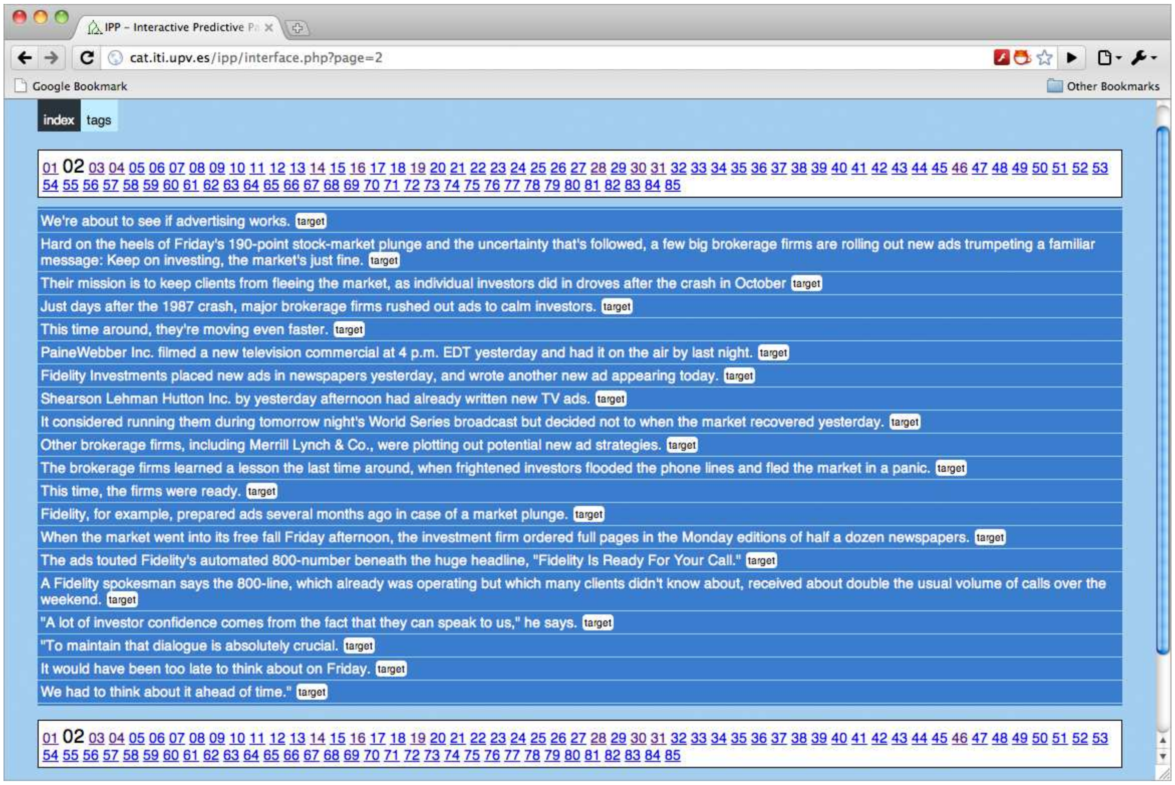
- **span modification**: Draw a line from the constituent node to the word that corresponds to the span's right index.
- **label substitution**: Select the text field and type the correct one.
- **unary production insertion**: Draw a line from the constituent node to the floating ball that appears below itself.
- **unary production removal**: Reset the span of the constituent parenting the unary production (just draw a line from the constituent node to the word that corresponds to the span's right index).

When a correction operation is performed over a constituent, the tree prefix is validated. The tree prefix consists of *a*) the validated constituent, *b*) all of its ancestors, and *c*) all constituents to the left of the corrected one.

The aforementioned operations result in a very straightforward operation protocol that is performed by the manual corrector, in which she validates or corrects the successive output parse trees:

1) The parsing system proposes a full parse tree for the input sentence.

This is a preprint for personal use only. The published paper may be subject to some form of copyright.



We're about to see if advertising works.

Hard on the heels of Friday's 190-point stock-market plunge and the uncertainty that's followed, a few big brokerage firms are rolling out new ads trumpeting a familiar message: Keep on investing, the market's just fine.

Their mission is to keep clients from fleeing the market, as individual investors did in droves after the crash in October

Just days after the 1987 crash, major brokerage firms rushed out ads to calm investors.

This time around, they're moving even faster.

PaineWebber Inc. filmed a new television commercial at 4 p.m. EDT yesterday and had it on the air by last night.

Fidelity Investments placed new ads in newspapers yesterday, and wrote another new ad appearing today.

Shearson Lehman Hutton Inc. by yesterday afternoon had already written new TV ads.

It considered running them during tomorrow night's World Series broadcast but decided not to when the market recovered yesterday.

Other brokerage firms, including Merrill Lynch & Co., were plotting out potential new ad strategies.

The brokerage firms learned a lesson the last time around, when frightened investors flooded the phone lines and fled the market in a panic.

This time, the firms were ready.

Fidelity, for example, prepared ads several months ago in case of a market plunge.

When the market went into its free fall Friday afternoon, the investment firm ordered full pages in the Monday editions of half a dozen newspapers.

The ads touted Fidelity's automated 800-number beneath the huge headline, "Fidelity Is Ready For Your Call."

A Fidelity spokesman says the 800-line, which already was operating but which many clients didn't know about, received about double the usual volume of calls over the weekend.

"A lot of investor confidence comes from the fact that they can speak to us," he says.

"To maintain that dialogue is absolutely crucial.

It would have been too late to think about on Friday.

We had to think about it ahead of time."

Figure 2. IPP-Ann showing sentences to be annotated.

2) Then, the user finds the first incorrect constituent exploring the tree in a certain ordered manner (preorder in our case, given by the tree prefix definition) and amends it, by modifying its span and/or label (implicitly validating a prefix tree).
3) The parsing system produces the most probable subtree that is compatible with the validated prefix tree.
4) Steps 2-3 are iterated until a final, perfect parse tree is produced by the server and validated by the user.

Validation can be performed by pressing CTRL+Enter inside a text field. The system then validates the current sentence and goes on to the next one.

IV. IPP-Ann Architecture

IPP-Ann has a modular architecture: it comprises a web client implemented by a combination of PHP and ActionScript (where the annotated trees are drawn and the user interaction is performed), and a parse server implemented in C++ (which does the probabilistic parsing iterations). They both communicate by using the socket-based CAT-API library [7].

A. Parse server

The server implements customized CYK-Viterbi parser, which uses a Probabilistic Context-Free Grammar (PCFG) in Chomsky Normal Form (CNF). Currently we have available two instances of the server, each running with a different grammar as the parse model. The grammar for English parsing were obtained from sections 2 to 21 of the UPenn Treebank. The Spanish server uses a grammar obtained from the first 1400 sentences of the UAM Spanish Treebank.

The server can provide the most probable subtree for any given span of the input string. For each subtree request, the subtree root label can be optionally provided. If the subtree root label is not provided, the server calculates the most probable label.

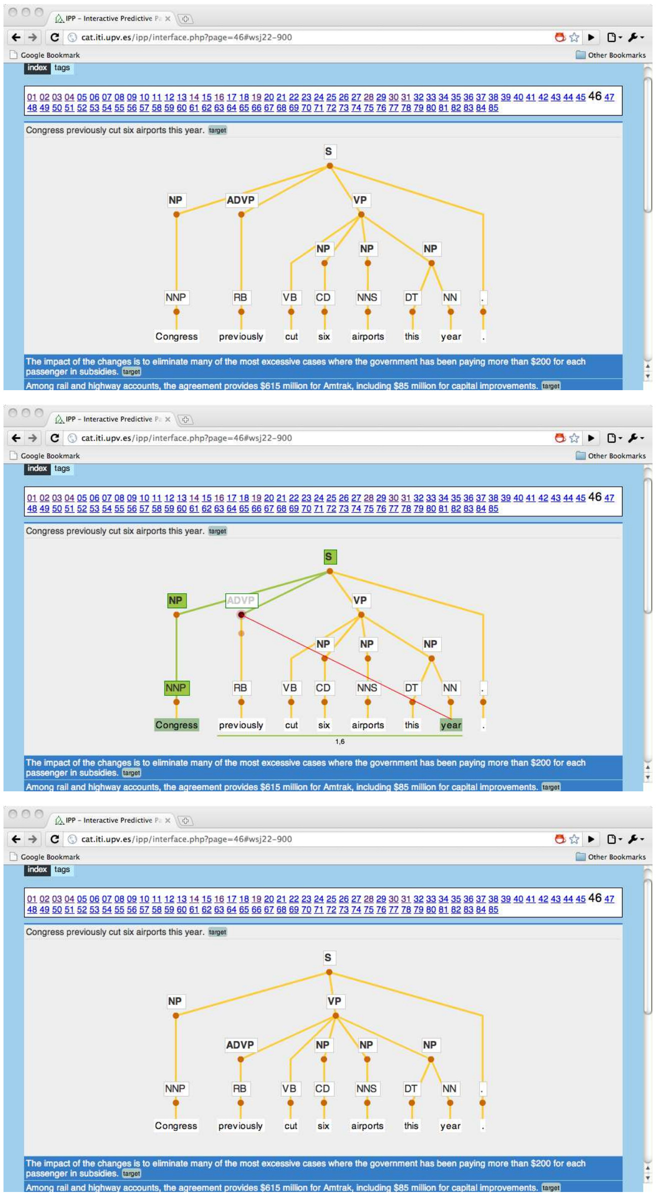
Given that the parse server internally works with a grammar in CNF, the server also performs transparent tree debinarization/binarization and unary expansion/callpsing, when sending and receiving trees to the client.

B. Web client

The client application runs on any modern Web browser, with the only requirement being the Flash plugin (99% of market penetration as of 2010, according to Adobe). The client's hardware requirements are quite low, as the parse server runs on a different computer. Additionally, the chosen client architecture provides cross-platform compatibility and requires neither computational power nor disk space on the client's machine.

The client interface has several additions that aid within the annotation process. Each validated user interaction is

This is a preprint for personal use only. The published paper may be subject to some form of copyright.



Congress previously cut six airports this year.

The impact of the changes is to eliminate many of the most excessive cases where the government has been paying more than $200 for each passenger in subsidies.

Among rail and highway accounts, the agreement provides $615 million for Amtrak, including $85 million for capital improvements.

Figure 3. Usage of IPP-Ann: the user modifies the span of a constituent and the system produces a new tree.

saved as a log file on the server side, so a tree's annotation session can be later resumed. Sentences with a started annotation session are shown in black text over a light green background. The annotation-in-progress log files can also be deleted, individually at the sentence level, or all the logs in a corpus at once.

When a tree is validated by the user, the final error-free tree is stored for later reuse. Validated sentences are shown in white text over a dark green background.

The user can also enter personalized text to parse through our client (see Figure 1). She either can write a single sentence (used mainly for testing purposes), or upload a file with several sentences for annotation. The text introduced by the user is automatically segmented by the client by an internal tokenizer (which separates punctuation marks from each other and from regular words). The system uses one of the currently provided grammars (either for English or for Spanish), and language detection is performed automatically.

When the user has finished annotating, she can download the syntax trees for each sentence that was validated (and also partially annotated) as an XML file; e.g., for later custom post-processing.

1) Communication protocol: Several annotators can work simultaneously from different locations on the same corpus (although not on the same sentence). When somebody is annotating a sentence, it is locked and shown with a red background.

By using the CAT-API library, the client communicates with the IPP server through binary TCP sockets, which provide low latency times. Moreover, client and server communicate via asynchronous HTTP connections, so there are no page refreshes when annotating a new sentence.

C. Technical requeriments

As previously commented, the only requirement for the client is a Web browser with the Flash plugin installed.

The parse server can be run in an average computer (Pentium 4 or higher recommended) with enough amount of RAM (for the grammars we use, 1 GB of RAM is enough). The default servers accessed when using our demonstration online are both running on a Intel® Core^{TM} 2 Quad CPU running @ 2.40GHz with 4 GB of RAM installed, since there are more CAT-API prototypes [8] on the same machine.

V. Conclusions and Future Work

We have introduced IPP-Ann, a Web-based interactive-predictive tool for syntactic tree annotation that can greatly speed the work of linguists creating new corpora. According to experiments on user simulation, the effort savings achieved by using our tool has been estimated to be around 46% when compared to using a non-interactive automatic system.

Future work includes several improvements on the client side (e.g., adding several XML export formats, and parsing directly text from an external web page), as well as some additions to the server side (e.g., adding grammars for new languages, or using better performing parsing algorithms).

Acknowledgments

Work partially supported by the EC (FEDER/FSE) and the Spanish MEC/MICINN under the MIPRCV "Consolider Ingenio 2010" program (CSD2007-00018), MITTRAL (TIN2009-14633-C03-01), and ALMPR (Prometeo/2009/014) research projects, and the FPU fellowship AP2006-01363.

This is a preprint for personal use only. The published paper may be subject to some form of copyright.

References

[1] R. Sánchez-Sáez, J. A. Sánchez, and J. M. Benedí, "Interactive predictive parsing," in *Proceedings of the 11th International Conference on Parsing Technologies (IWPT'09)*. Paris, France: Association for Computational Linguistics, October 2009, pp. 222–225. [Online]. Available: http://www.aclweb.org/anthology/W09-3835

[2] D. Carter, "The TreeBanker. A tool for supervised training of parsed corpora," in *Proceedings of the Workshop on Computational Environments for Grammar Development and Linguistic Engineering*, 1997, pp. 9–15.

[3] S. Oepen, D. Flickinger, K. Toutanova, and C. Manning, "LinGO Redwoods," *Research on Language & Computation*, vol. 2, no. 4, pp. 575–596, 2004.

[4] I. Hiroshi, N. Masaki, H. Taiichi, T. Takenobu, and T. Hozumi, "ebonsai: An integrated environment for annotating treebanks," in *Second International Joint Conference on Natural Language Processing*, 2005, pp. 108–113.

[5] R. Sánchez-Sáez, L. A. Leiva, J. A. Sánchez, and J. M. Benedí, "Interactive predictive parsing using a web-based architecture," in *Proceedings of the Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT'10)*, Los Angeles, United States of America, June 2010.

[6] ——, "Interactive predictive parsing framework for the spanish language," in *XXVI Congreso de la Sociedad Española para el Procesamiento del Lenguaje Natural*, Valencia, Spain, September 2010.

[7] V. Alabau, D. Ortiz, V. Romero, and J. Ocampo, "A multimodal predictive-interactive application for computer assisted transcription and translation," in *ICMI-MLMI '09: Proceedings of the 2009 international conference on Multimodal interfaces*. New York, NY, USA: ACM, 2009, pp. 227–228.

[8] V. Alabau, J. M. Benedí, F. Casacuberta, L. A. Leiva, D. Ortiz-Martínez, . V. Romero, J. A. Sánchez, R. Sánchez-Saez, A. H. Toselli, and E. Vidal, "Cat-api framework prototypes," in *Proceedings of the 1st International Workshop on Interactive Multimodal Pattern Recognition in Embedded Systems (IMPRESS) in the DEXA conference*, 2010.