Chapter 1

THE PENN TREEBANK: AN OVERVIEW

Ann Taylor
University of York
Heslington, York, UK
at9@york.ac.uk

Mitchell Marcus, Beatrice Santorini
University of Pennsylvania
Philadelphia PA, USA
{mitch,beatrice}@linc.cis.upenn.edu

Abstract The Penn Treebank, in its eight years of operation (1989-1996), produced approximately 7 million words of part-of-speech tagged text, 3 million words of skeletally parsed text, over 2 million words of text parsed for predicate-argument structure, and 1.6 million words of transcribed spoken text annotated for speech disfluencies. This paper describes the design of the three annotation schemes used by the Treebank: POS tagging, syntactic bracketing, and disfluency annotation and the methodology employed in production. All available Penn Treebank materials are distributed by the Linguistic Data Consortium `http://www.ldc.upenn.edu`.

Keywords: English, Annotated Corpus, Part-of-speech Tagging, Treebank, Syntactic Bracketing, Parsing, Disfluencies

INTRODUCTION

The Penn Treebank, in its eight years of operation (1989-1996), produced approximately 7 million words of part-of-speech tagged text, 3 million words of skeletally parsed text, over 2 million words of text parsed for predicate-argument structure, and 1.6 million words of transcribed spoken text annotated for speech disfluencies. The material annotated includes such wide-ranging genres as IBM computer manuals, nursing notes, Wall Street Journal articles, and transcribed telephone conversations, among others. This pa-

A. Abeillé (ed.), Treebanks: Building and Using Parsed Corpora, 5–22.

per describes the design of the three annotation schemes used by the Treebank: POS tagging, syntactic bracketing, and disfluency annotation (section 1) and the methodology employed in production (section 2).[1] All available Penn Treebank materials are distributed by the Linguistic Data Consortium `http://www.ldc.upenn.edu`.

1. THE ANNOTATION SCHEMES

The majority of the output of the Penn Treebank consists of POS tagged and syntactically bracketed versions of written texts such as the Wall Street Journal and the Brown Corpus. In the early years of the project bracketing was done using a quite simple skeletal parse, while later phases made use of a richer predicate-argument bracketing schema. In the final phase of operation, we produced a tagged and parsed version of part of the Switchboard corpus of telephone conversations, as well as a version annotated for disfluencies. In the remainder of this section we discuss the design of the three annotation schemes.

1.1 Part-of-speech tagging

The part-of-speech (POS) tagsets used to annotate large corpora prior to the Penn Treebank were generally fairly extensive. The rationale behind developing such large, richly articulated tagsets was to approach "the ideal of providing distinct codings for all classes of words having distinct grammatical behaviour" (Garside, Leech, and Sampson 1987). The Penn Treebank tagset, like many others, is based on that of the Brown Corpus, but it differs from it in a number of important ways.

First, the stochastic orientation of the Penn Treebank and the resulting concern with sparse data led us to modify the Brown Corpus tagset (Francis, 1964; Francis and Kučera, 1982) by paring it down considerably. The key strategy in this reduction was to eliminate lexical and syntactic redundancy. Thus, whereas many POS tags in the Brown Corpus tagset are unique to a particular lexical item, the Penn Treebank tagset strives to eliminate such instances of lexical redundancy. For instance, the Brown Corpus distinguishes the forms of the verbs *have, be*, and *do* from other main verbs by different tags. By contrast, since the distinctions between the forms of these verbs is lexically recoverable, they are eliminated in the Penn Treebank and all main verbs receive the same set of tags. Distinctions recoverable with reference to syntactic structure were also eliminated. For instance, the Penn Treebank tagset does not distinguish subject pronouns from object pronouns even in cases where the distinction is not recoverable from the pronoun's form, as with *you*, since the distinction is recoverable on the basis of the pronoun's position in the parse tree in the parsed version of the corpus.

A second difference between the Penn Treebank and the Brown Corpus concerns the significance accorded to syntactic context. In the Brown Corpus, words tend to be tagged independently of their syntactic function. For instance, in the phrase *the one*, *one* is always tagged as CD (cardinal number), whereas in the corresponding plural phrase *the ones*, *ones* is always tagged as NNS (plural common noun), despite the parallel function of *one* and *ones* as heads of their noun phrase. By contrast, since one of the main roles of the tagged version of the Penn Treebank corpus is to serve as the basis for a bracketed version of the corpus, we encode a word's syntactic function in its POS tag whenever possible. Thus, *one* is tagged as NN (singular common noun) rather than as CD (cardinal number) when it is the head of a noun phrase.

Thirdly, since a major concern of the Treebank is to avoid requiring annotators to make arbitrary decisions, we allow words to be associated with more than one POS tag. Such multiple tagging indicates either that the word's part of speech simply cannot be decided or that the annotator is unsure which of the alternative tags is the correct one.

The Penn Treebank tagset is given in Table 1.1. It contains 36 POS tags and 12 other tags (for punctuation and currency symbols). A detailed description of the guidelines governing the use of the tagset can be found in Santorini (1990) or on the the Penn Treebank webpage[2].

1.2 Syntactic bracketing

Skeletal parsing. During the operation of the Penn Treebank, two styles of syntactic bracketing were employed. In the first phase of the project the annotation used was a skeletal context-free bracketing with limited empty categories and no indication of non-contiguous structures and dependencies.

```
( (S
   (NP Martin Marietta Corp.)
   was
   (VP given
       (NP a
           $ 29.9
           million Air Force contract
           (PP for
               (NP low-altitude navigation
                   and
                   targeting equipment)))))
  .)
```

The set of syntactic tags and null elements used in the skeletal bracketing are given in Table 1.2. More detailed information on the syntactic tagset and guide-

Table 1.1. The Penn Treebank POS tagset

CC	Coordinating conj.	TO	infinitival *to*
CD	Cardinal number	UH	Interjection
DT	Determiner	VB	Verb, base form
EX	Existential there	VBD	Verb, past tense
FW	Foreign word	VBG	Verb, gerund/present pple
IN	Preposition	VBN	Verb, past participle
JJ	Adjective	VBP	Verb, non-3rd ps. sg. present
JJR	Adjective, comparative	VBZ	Verb, 3rd ps. sg. present
JJS	Adjective, superlative	WDT	Wh-determiner
LS	List item marker	WP	Wh-pronoun
MD	Modal	WP$	Possessive *wh*-pronoun
NN	Noun, singular or mass	WRB	Wh-adverb
NNS	Noun, plural	#	Pound sign
NNP	Proper noun, singular	$	Dollar sign
NNPS	Proper noun, plural	.	Sentence-final punctuation
PDT	Predeterminer	,	Comma
POS	Possessive ending	:	Colon, semi-colon
PRP	Personal pronoun	(	Left bracket character
PP$	Possessive pronoun	)	Right bracket character
RB	Adverb	"	Straight double quote
RBR	Adverb, comparative	`	Left open single quote
RBS	Adverb, superlative	``	Left open double quote
RP	Particle	'	Right close single quote
SYM	Symbol	''	Right close double quote

lines concerning its use are to be found in Santorini and Marcinkiewicz (1991) or on The Penn Treebank website[3].

Following the release of the first Penn Treebank CD-ROM, many users indicated that they wanted forms of annotation richer than those provided by the project's first phase, as well as an increase in the consistency of the preliminary corpus. Some also expressed an interest in a less skeletal form of annotation, expanding the essentially context-free analysis of the current treebank to indicate non-contiguous structures and dependencies. Most crucially, there was a strong sense that the Treebank could be of much more use if it explicitly provided some form of predicate-argument structure. The desired level of representation would make explicit at least the logical subject and logical object of the verb, and indicate, at least in clear cases, how subconstituents are semantically related to their predicates. Therefore in the second phase of the project a new style of annotation, Treebank II, was introduced.

Table 1.2. The Penn Treebank syntactic tagset

ADJP	Adjective phrase
ADVP	Adverb phrase
NP	Noun phrase
PP	Prepositional phrase
S	Simple declarative clause
SBAR	Subordinate clause
SBARQ	Direct question introduced by *wh*-element
SINV	Declarative sentence with subject-aux inversion
SQ	Yes/no questions and subconstituent of SBARQ excluding *wh*-element
VP	Verb phrase
WHADVP	Wh-adverb phrase
WHNP	Wh-noun phrase
WHPP	Wh-prepositional phrase
X	Constituent of unknown or uncertain category
*	"Understood" subject of infinitive or imperative
0	Zero variant of *that* in subordinate clauses
T	Trace of wh-Constituent

Predicate-argument structure. The new style of annotation provided three types of information not included in the first phase.

1 A clear, concise distinction between verb arguments and adjuncts where such distinctions are clear, with an easy-to-use notational device to indicate where such a distinction is somewhat murky.

2 A non-context free annotational mechanism to allow the structure of discontinuous constituents to be easily recovered.

3 A set of null elements in what can be thought of as "underlying" position for phenomena such as wh-movement, passive, and the subjects of infinitival constructions, co-indexed with the appropriate lexical material.

The goal of a well-developed predicate-argument scheme is to label each argument of the predicate with an appropriate semantic label to identify its role with respect to that predicate (subject, object, etc.), as well as distinguishing the arguments of the predicate, and adjuncts of the predication. Unfortunately, while it is easy to distinguish arguments and adjuncts in simple cases, it turns out to be very difficult to consistently distinguish these two categories for many verbs in actual contexts. It also turns out to be very difficult to determine a set of underlying semantic roles that holds up in the face of a few

paragraphs of text. The Treebank II scheme is an attempt to come up with a middle ground which allows annotation of those distinctions that seem to hold up across a wide body of material. After many attempts to find a reliable test to distinguish between arguments and adjuncts, we abandoned structurally marking this difference. Instead, we decided to label a small set of clearly distinguishable roles, building upon syntactic distinctions only when the semantic intuitions were clear-cut. However, getting annotators to consistently apply even the small set of distinctions discussed here was fairly difficult.

In the skeletal parsing scheme discussed in section 1.2 we used only standard syntactic labels (e.g. NP, ADVP, PP, etc.) for our constituents (see Table 1.2) – in other words, every bracket had just one label. The limitations of this become apparent when a word belonging to one syntactic category is used for another function or when it plays a role which we want to be able to identify easily. In Treebank II style, each constituent has at least one label but as many as four tags, including numerical indices, taken from the set of functional tags given in Table 1.3. NPs and Ss which are clearly arguments of the verb are unmarked by any tag. An open class of other cases that individual annotators feel strongly should be part of the VP are tagged as *-CLR* (for CLosely Related); constituents marked *-CLR* typically correspond to the class of predication adjuncts proposed by (Quirk et al. 1985)[4]. In addition, a handful of semantic roles are distinguished: direction, location, manner, purpose, and time, as well as the syntactic roles of surface subject, logical subject, and (implicit in the syntactic structure) first and second verbal objects.

```
( (S (NP-SBJ-1 Jones)
     (VP followed
         (NP him)
         (PP-DIR into
                 (NP the front room))
         ,
         (S-ADV (NP-SBJ *-1)
                (VP closing
                    (NP the door)
                    (PP behind
                        (NP him)))))
     .))

( (S (ADVP-LOC Here)
     (NP-SBJ-1 he)
     (VP could
         n't
         (VP be
             (VP seen
                 (NP *-1)
                 (PP by
                     (NP-LGS (NP Blue Throat)
                             and
                             (NP his gang))))))
     .))
```

Treebank II style also adds null elements in a wide range of cases; these null elements are co-indexed with the lexical material for which the null element

Table 1.3. Functional Tags

Text Categories	
-HLN	headlines and datelines
-LST	list markers
-TTL	titles
Grammatical Functions	
-CLF	true clefts
-NOM	non NPs that function as NPs
-ADV	clausal and NP adverbials
-LGS	logical subjects in passives
-PRD	non VP predicates
-SBJ	surface subject
-TPC	topicalized and fronted constituents
-CLR	closely related - see text
Semantic Roles	
-VOC	vocatives
-DIR	direction & trajectory
-LOC	location
-MNR	manner
-PRP	purpose and reason
-TMP	temporal phrases

stands. The current scheme uses two symbols for null elements: **T**, which marks WH-movement and topicalization, and * which is used for all other null elements. Co-indexing of null elements is done by suffixing an integer to non-terminal categories (e.g. `NP-10, VP-25`). This integer serves as an id number for the constituent. A null element itself is followed by the id number of the constituent with which it is co-indexed. Crucially, the predicate-argument structure can be recovered by simply replacing the null element with the lexical material that it is co-indexed with.

```
(SBARQ (WHNP-1 What)
       (SQ is
           (NP-SBJ Tim)
           (VP eating
               (NP *T*-1)))
       ?)

Predicate Argument Structure:
eat(Tim, what)
```

```
(S (NP-SBJ-1 The ball)
   (VP was
       (VP thrown
           (NP *-1)
           (PP by
               (NP-LGS Chris)))))

Predicate Argument Structure:
throw(Chris, ball)
```

A null element is also used to indicate which lexical NP is to be interpreted as the null subject of an infinitive complement clause; it is co-indexed with the controlling NP, based upon the lexical properties of the verb.

```
(S (NP-SBJ-1 Chris)
   (VP wants
       (S (NP-SBJ *-1)
          (VP to
              (VP throw
                  (NP the ball))))))


Predicate Argument Structure:
wants(Chris, throw(Chris, ball))
```

Finally, we also use null elements to allow the interpretation of other grammatical structures where constituents do not appear in their default positions. Null elements are used in most cases to mark the fronting (or "topicalization" of any element of an S before the subject (except in subj-aux inversion). If an adjunct is topicalized, the fronted element does not leave a trace since the level of attachment is the same, only the word order is different. Topicalized arguments, on the other hand, always are marked by a null element:

```
(S (NP-TPC-5 This)
   (NP-SBJ every man)
   (VP contains
       (NP *T*-5)
       (PP-LOC within
               (NP him))))
```

Again, this makes predicate argument interpretation straightforward, if the null element is simply replaced by the constituent to which it is co-indexed.

With only a skeletal parse as used in the first phase of the Treebank project, many otherwise clear argument/adjunct relations cannot be recovered due to its essentially context-free representation. For example, there is no good representation for sentences in which constituents which serve as complements to the verb occur after a sentence-level adverb. Either the adverb is trapped within the VP, so that the complement can occur within the VP, where it belongs, or else the adverb is attached to the S, closing off the VP and forcing the complement to attach to the S. This "trapping" problem serves as a limitation when using skeletally parsed material to semi-automatically derive lexicons for particular applications.

"Trapping" problems and the annotation of non-contiguous structure can be handled by simple notational devices that use co-indexing to indicate discontinuous structures. Again, an index number added to the label of the original constituent is incorporated into the null element which shows where that constituent should be interpreted within the predicate argument structure. We use a variety of null elements to show how non-adjacent constituents are related; such constituents are referred to as "pseudo-attached". There are four different types of pseudo-attach, as shown in Table 1.4.

Table 1.4. The four types of pseudo-attachment

ICH	Interpret Constituent Here
PPA	Permanent Predictable Ambiguity
RNR	Right Node Raising
EXP	Expletive

The **ICH** pseudo-attach is used for simple extraposition, solving the most common case of "trapping":

```
(S (NP-SBJ Chris)
   (VP knew
       (SBAR *ICH*-1)
       (NP-TMP yesterday)
       (SBAR-1 that
               (S (NP-SBJ Terry)
                  (VP would
                      (VP catch
                          (NP the ball)))))))
```

Here, the clause *that Terry would catch the ball* is to be interpreted as an argument of *knew*.

The **RNR** tag is used for so-called "right-node raising" conjunctions, where the same constituent appears to have been shifted out of both conjuncts.

```
(S But
   (NP-SBJ-2 our outlook)
   (VP (VP has
           (VP been
               (ADJP *RNR*-1)))
       ,
       and
       (VP continues
           (S (NP-SBJ *-2)
              (VP to
                  (VP be
                      (ADJP *RNR*-1)))))
       ,
       (ADJP-1 defensive)))
```

In order that certain kinds of constructions can be found reliably within the corpus, we have adopted special marking of some special constructions. For example, extraposed sentences which leave behind a semantically null "it" are parsed as follows, using the **EXP** tag:

```
(S (NP-SBJ (NP It)
           (S *EXP*-1))
   (VP is
       (NP a pleasure))
   (S-1 (NP-SBJ *)
        (VP to
            (VP teach
                (NP her)))))

Predicate Argument Structure:
pleasure(teach(*someone*, her))
```

The **PPA** tag was introduced to indicate "permanent predictable ambiguity", those cases in which one cannot tell where a constituent should be attached, even given context. Here, annotators attach the constituent at the more likely site (or if that is impossible to determine, at the higher site) and pseudo-attach it at all other plausible sites using the **PPA** null element.[5]

```
(S (NP-SBJ I)
       (VP saw
           (NP (NP the man)
               (PP *PPA*-1))
           (PP-CLR-1 with
                     (NP the telescope))))
```

1.3 Disfluency annotation

The final project undertaken by the Treebank (1995-6) was to produce a tagged and parsed version of the Switchboard corpus of transcribed telephone conversations, along with a version which annotated the disfluencies common in speech (fragments of words, interruptions, incomplete sentences, fillers and discourse markers).

The disfluency annotation system (based on Shriberg (1994)) distinguishes complete utterances from incomplete ones, labels a range of non-sentence elements such as fillers, and annotates restarts.

Table 1.5. Disfluency Annotation

Utterances	
/	end of complete utterance
-/	end of incomplete utterance
Non-sentence elements	
F	fillers (*uh, um, huh, oh,* etc.)
E	explicit editing term (*I mean, sorry,* etc.)
D	discourse marker (*you know, well,* etc.)
C	coordinating conjunction (*and, and then, but,* etc.)
A	aside
Restarts	
$[RM+RR]$	restart with repair (see text)
$[RM+]$	restart without repair

Restarts have the following form:

```
Show me flights from Boston on  uh from Denver on Monday
               |-------RM-----|-IM------RR-------|
                              IP
  RM = reparandum
  IP = interruption point
  IM = interregnum (filled pause or editing terms)
  RR = repair
```

In the annotation, the entire restart with its repair is contained in square brackets. The IP is marked by a "+", and any IM material (filled pauses, etc.) follows the "+".

```
Show me flights
   [ from Boston on + {F uh } from Denver on ] Monday

   |-------RM-------|---IM---|-------RR------|
                    IP
```

A: he's pretty good. / He stays out of the street / {C and, } {F uh, } if I catch him I call him / {C and } he comes back. / {D So } [he, + he's] pretty good about taking to commands [and + –
B: {F Um. } /
A: – and] things. /
B: Did you bring him to a doggy obedience school or –
A: No – /
B: – just –
A: – we never did. /
B: – train him on your own / {C and, } -/
A: [I, + I] trained him on my own / {C and, } {F uh, } this is the first dog I've had all my own as an adult. /
B: Uh-huh. /

Figure 1.1. Sample disfluency annotation

A detailed account of the disfluency annotation can be found in Mateer and Taylor 1995 or on the Penn Treebank website `http://www.cis.upenn.edu/~treebank`.

2. METHODOLOGY

The three types of Treebank annotation, POS tagging, syntactic bracketing, and disfluency annotation, are all produced by the same two-step method, automatic annotation followed by manual correction. The correction of each type of annotation is done with the aid of a task-specific mouse-based package written in GNU Emacs Lisp, embedded in the GNU Emacs editor (Lewis and Laliberte 1990). POS tagging and disfluency annotation (when relevant) feed syntactic bracketing, but the first two are independent of each other and can be done in parallel, with the two output streams then being automatically merged, if desired.

2.1 Part-of-speech tagging

During the early stages of the Penn Treebank project, the initial automatic POS assignment was provided by PARTS (Church 1988), a stochastic algorithm developed at AT&T Bell Labs. PARTS uses a modified version of the Brown Corpus tagset close to our own and assigns POS tags with an error rate of 3–5%. The output of PARTS was automatically tokenized and the tags assigned by PARTS were automatically mapped onto the Penn Treebank tagset. This mapping introduced about 4% error, since the Penn Treebank tagset makes certain distinctions that the PARTS tagset does not. Later, the automatic POS assignment was provided by a cascade of stochastic and rule-driven taggers developed on the basis of our early experience. Since these taggers are based on the Penn Treebank tagset, the 4% error rate introduced as an artefact of mapping from the PARTS tagset to ours is eliminated, and we obtain error rates of 2–6%. Finally, during the Switchboard project we switched to the then recently released Brill tagger (Brill 1993).

The result of the first, automated stage of POS tagging is given to annotators to correct. The POS correction interface allows annotators to correct POS assignment errors by positioning the cursor on an incorrectly tagged word and then entering the desired correct tag (or sequence of multiple tags). The annotators' input is automatically checked against the list of legal tags and, if valid, appended to the original word-tag pair separated by an asterisk. Appending the new tag rather than replacing the old tag allows us to easily identify recurring errors at the automatic POS assignment stage. Finally, in the distribution version of the tagged corpus, any incorrect tags assigned at the first, automatic stage are removed.

2.2 Syntactic bracketing

The methodology for bracketing the corpus is completely parallel to that for tagging—hand correction of the output of an automatic process. Fidditch, a deterministic parser developed by Donald Hindle first at the University of Pennsylvania and subsequently at AT&T Bell Labs (Hindle 1988, Hindle 1989) is used to provide an initial parse of the material. Annotators then hand correct the parser's output using a task-specific mouse-based interface implemented in GNU Emacs Lisp. Fidditch has three properties that make it ideally suited to serve as a preprocessor to hand correction:

- It always provides exactly one analysis for any given sentence, so that annotators need not search through multiple analyses.
- It never attaches any constituent whose role in the larger structure it cannot determine with certainty. In cases of uncertainty, Fidditch chunks

Output of tagger

```
Battle-tested/NNP Japanese/NNP industrial/JJ managers/NNS
here/RB always/RB buck/VB up/IN nervous/JJ newcomers/NNS
with/IN the/DT tale/NN of/IN the/DT first/JJ of/IN
their/PP$ countrymen/NNS to/TO visit/VB Mexico/NNP ,/,
a/DT boatload/NN of/IN samurai/NNS warriors/NNS blown/VBN
ashore/RB 375/CD years/NNS ago/RB ./.
```

Hand-corrected by annotator

```
Battle-tested/NNP*/JJ Japanese/NNP*/JJ industrial/JJ
managers/NNS  here/RB always/RB buck/VB*/VBP up/IN*/RP
nervous/JJ newcomers/NNS  with/IN the/DT tale/NN of/IN
the/DT first/JJ of/IN their/PP$ countrymen/NNS to/TO
visit/VB Mexico/NNP ,/, a/DT boatload/NN of/IN
samurai/NNS*/FW  warriors/NNS blown/VBN ashore/RB 375/CD
years/NNS ago/RB ./.
```

Final version

```
Battle-tested/JJ Japanese/JJ industrial/JJ managers/NNS
here/RB always/RB buck/VBP up/RP nervous/JJ newcomers/NNS
with/IN the/DT tale/NN of/IN the/DT first/JJ of/IN
their/PP$ countrymen/NNS to/TO  visit/VB Mexico/NNP ,/,
a/DT boatload/NN  of/IN samurai/FW warriors/NNS  blown/VBN
ashore/RB 375/CD years/NNS ago/RB ./.
```

Figure 1.2. Part-of-speech tagging pipeline

the input into a string of trees, providing only a partial structure for each sentence.

- It has rather good grammatical coverage, so that the grammatical chunks that it does build are usually quite accurate.

The output of Fidditch, however, which is fairly complex, with word, X-bar, and phrase levels represented, was found to be too complicated for the annotators to handle at speed. They were therefore presented with a simplified parse containing only the phrase labels for correction. The simplified output of Fidditch is illustrated in Figure 2.2. In general, the annotators do not need to re-bracket much of the parser's output—a relatively time-consuming task. Rather,

the annotators' main task is to "glue" together the syntactic chunks produced by the parser. Using a mouse-based interface, annotators move each unattached chunk of structure under the node to which it should be attached. Notational devices allow annotators to indicate uncertainty concerning constituent labels, and to indicate multiple attachment sites for ambiguous modifiers. Insertion and co-indexing of null elements is accomplished simply by dragging from the associated lexical material to the site of the null element.

```
( (S
        (NP Her eyes)
        (AUX were)
        (VP glazed))
    (?
        (PP as
            (NP (SBAR if
                    (S (NP she)
                        (AUX did)
                        (?
                            (NEG n't))
                        (VP hear))))))
    (? or)
    (? even)
    (?
        (S
            (VP see
                (NP him))))
    (? .))
```

Figure 1.3. Simplified output of Fidditch before correction

The bracketed text after correction is shown in Figure 1.3. The fragments are now connected together into one rooted tree structure, functional tags are added and null elements inserted and co-indexed.
Finally the POS tags can be automatically combined with the skeletal parse to produce a tree with both POS and syntactic information.

2.3 Disfluency annotation

As with POS tagging and syntactic bracketing, annotating disfluencies is done in two steps, although in this case the automated step is just a simple Perl script which attempts to identify and bracket the more common non-sentence elements, such as fillers. The correction interface for disfluencies allows easy

```
( (S (NP-SBJ-2 Her eyes)
     (VP were
         (VP glazed
             (NP *-2)
             (SBAR-ADV as if
                       (S (NP-SBJ she)
                          (VP did n't
                              (VP (VP hear
                                      (NP *RNR*-1))
                                  or
                                  (VP (ADVP even)
                                      see
                                      (NP *RNR*-1))
                                  (NP-1 him)))))))
     .))
```

Figure 1.4. Bracketed text after correction

input and manipulation of the annotations which mark restarts and repairs with the same sort of mouse-driven package used for correcting the syntactic parse.

2.4 Productivity

The learning curve for the POS tagging task takes under a month (at 15 hours a week), and annotation speeds after a month exceed 3,000 words per hour. The rate for disfluency annotation is similar. Not surprisingly, annotators take substantially longer to learn the more complicated bracketing task, with substantial increases in speed occurring even after two months of training. Even after extended training, performance varies markedly by annotator, however, with speeds on the task ranging from approx. 750 words per hour to well over 1,000 words per hour after three or four months experience.

3. CONCLUSIONS

Although the Penn Treebank is no longer in operation the large amount of data produced by the project continues to provide a valuable resource for computational linguists, natural language programmers, corpus linguists and others interested in empirical language studies. In addition, the tools and methodology developed by the Penn Treebank have been adopted with some revision by an ongoing project to create parsed corpora of all the historical

stages of English, which is centred at the University of Pennsylvania and the University of York, with support from the University of Helsinki. The first corpus produced by the project, now in its second edition, the Penn-Helsinki Parsed Corpus of Middle English, Second Edition, (Kroch and Taylor 2000) (`http://www.ling.upenn.edu/mideng`), has been released and comparable corpora of Old and Early Modern English are in production .

Acknowledgments

The Penn Treebank has been supported by the Linguistic Data Consortium, by DARPA (grant No. N0014-85-K0018), by DARPA and AFOSR jointly (grant No. AFOSR-90-0066) and by ARO (grant No. DAAL 03-89-C0031 PRI). Seed money was provided by the General Electric Corporation (grant No. J01746000). The contribution of the staff of the Treebank, Grace Kim, Mary Ann Marcinkiewicz, Dagmar Sieglova (project administrators) Robert MacIntyre (data manager/programmer), and the many annotators who worked on the project over the years, Ann Bies, Constance Cooper, Leslie Dossey, Mark Ferguson, Robert Foye, Karen Katz, Shannon Keenan, Alison Littman James Reilly, Britta Schasberger, James Siegal, Kevin Stephens, and Victoria Tredinnick, is also gratefully acknowledged.

Notes

1. This paper is a revised and expanded version of two earlier papers: Marcus et al. (1993) "Building a large annotated corpus of English: the Penn Treebank," in *Computational Linguistics* 19(2):313-220, and Marcus et al. (1994) "The Penn Treebank: annotating predicate argument structure," in *ARPA Human Language Technology Workshop.*

2. `http://www.cis.upenn.edu/~treebank`

3. `http://www.cis.upenn.edu/~treebank`

4. This use of this tag was an experiment which was largely unsuccessful. Although some very experienced annotators were internally fairly consistent in their use of this tag, less experienced annotators had a hard time with it and consistency across annotators was not high. It was not used in the parsing of the Switchboard corpus.

5. The use of *PPA* was discontinued in the Switchboard phase, since annotators did not reliably detect these ambiguities.

References

Bies, Ann, Mark Ferguson, Karen Katz, and Robert MacIntyre. (1995). *Bracketing Guidelines for Treebank II Style.* Ms., Department of Computer and Information Science, University of Pennsylvania.

Brill, Eric. (1993). *A Corpus-based Approach to Language Learning*. PhD Dissertation, University of Pennsylvania.

Church, Kenneth W. (1980). *Memory Limitations in Natural Language Processing*, MIT LCS Technical Report 245. Master's thesis, Massachusetts Institute of Technology.

Church, Kenneth W. (1988). A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the Second Conference on Applied Natural Language Processing. 26th Annual Meeting of the Association for Computational Linguistics*, pages 136–143.

Francis, W. Nelson (1964). *A Standard Sample of Present-day English for Use with Digital Computers*. Report to the U.S Office of Education on Cooperative Research Project No. E–007. Brown University, Providence RI.

Francis, W. Nelson and Henry Kučera. (1982). *Frequency Analysis of English Usage. Lexicon and Grammar*. Houghton Mifflin, Boston.

Garside, Roger, Geoffrey Leech, and Geoffrey Sampson. (1987). *The Computational Analysis of English. A Corpus-based Approach*. Longman, London.

Hindle, Donald. (1983). *User Manual for Fidditch*. Technical memorandum 7590–142, Naval Research Laboratory.

Hindle, Donald. (1989). Acquiring disambiguation rules from text. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*.

Kroch, Anthony S. and Ann Taylor. (2000). *The Penn-Helsinki Parsed Corpus of Middle English, Second Edition*. Department of Linguistics, University of Pennsylvania.

Lewis, Bil, Dan Laliberte, and the GNU Manual Group. (1990). *The GNU Emacs Lisp Reference Manual*. Free Software Foundation, Cambridge MA.

Marcus, Mitchell P., Beatrice Santorini, and Mary Ann Marcinkiewicz. (1993) Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics* 19(2):313-330.

Marcus, Mitchell P., Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. (1994). The Penn Treebank: Annotating predicate-argument structure. In *ARPA Human Language Technology Workshop*.

Mateer, Marie, and Ann Taylor. (1995). *Disfluency Annotation Stylebook for the Switchboard Corpus*. Ms., Department of Computer and Information Science, University of Pennsylvania.

Quirk, R., S. Greenbaum, G. Leech, and J. Svartvik. (1985). *A Comprehensive Grammar of the English Language*, Longman, London.

Santorini, Beatrice. (1990). *Part-of-speech Tagging Guidelines for the Penn Treebank Project*. Technical report MS-CIS-90-47, Department of Computer and Information Science, University of Pennsylvania.

Santorini, Beatrice and Mary Ann Marcinkiewicz. (1991). *Bracketing Guidelines for the Penn Treebank Project*. Ms., Department of Computer and Information Science, University of Pennsylvania.

Shriberg, E.E. (1994). *Preliminaries to a Theory of Speech Disfluencies*. PhD Dissertation, University of California at Berkeley.