A Factored Functional Dependency Transformation of the English Penn Treebank for Probabilistic Surface Generation

Irene Langkilde-Geary*, Justin Betteridge†

*Brigham Young University
Provo UT
irenelg@cs.byu.edu
† Carnegie-Mellon University
Pittsburg PA
betteridge@gmail.com

Abstract

This paper describes a featurized functional dependency corpus automatically derived from the Penn Treebank. Each word in the corpus is associated with over three dozen features describing the functional syntactic structure of a sentence as well as some shallow morphology. The corpus was created for use in probabilistic surface generation, but could also be useful as a resource for the study of English and the development of other NLP applications.

1. Introduction

Since the release of the Penn Treebank (Marcus et al., 1994), work on large-scale, robust probabilistic parsing has flourished, greatly advancing the state-of-the-art. In contrast, the inverse problem of probabilistic general-purpose surface generation has received much less attention. Besides our own work on HALogen (Langkilde-Geary, 2002) there are the Fergus (Bangalore and Rambow, 2000a) and Amalgam (Ringger et al., 2004) systems that have attempted to achieve broad coverage of English and evaluated their degree of success. Yet probabilistic approaches to sentence plan realization hold promise for applications in machine translation, human-computer dialog and automatic summarization, among others.

One important reason for this neglect is likely due to the mismatch between the style of annotation on the Penn Treebank and the kind of information typically needed to perform generation. For example, our own work on the HALogen system included 3+ person-years of effort to programmatically transform the Penn Treebank into a factored functional dependency representation. The transformed representation is more suitable than the original annotation was both for probabilistic language modeling as well as for simulating test inputs to the generator system. The resulting corpus seems likely to be useful to others as well. It consists of over 1 million words, and is now publicly available upon request (to those who already have rights to the original Treebank).

The most significant distinguishing characteristics of this corpus compared to similar efforts are the following:

- a dependency-style representation that designates head words for each constituent phrase
- labeling of every child-parent dependency with one of 33 syntactic functional roles (see table 3) that subsume predicate-argument structure and compound verbal constructions,
- association of each word with a hybrid set of features including factored part-of-speech information, base word form, and constituency-related information, (see Table 2 for the most important ones),
- additional nesting structure to make conjunction and punctuation relationships clearer, and
- treatment of prepositions and complementizers as role markers that are dependents of the head word on their right, to help localize probabilistic dependencies.

This paper briefly describes our task-oriented evaluation (Section 2.), the most important elements of the representation (Section 3.), how we converted it from the Penn Treebank (Section 4.), and related work (Section 5.).

2. Evaluation

The corpus has undergone a task-oriented evaluation in the context of surface generation. One task was to reproduce the original Penn Treebank sentence from the transformed corpus given the unordered dependendency structure (where adjuncts at the same level of the structure and on the same side of the head were treated as a unit in determining constituent order). This evaluation indirectly measures the quality of the corpus through its impact on the quality of the regenerated output. The evaluation is also affected by the determinism of the mapping from constituent ordering to functional roles as well as the correctness of the generator mapping rules from functional roles back to constituent positions.

On a development set, we adjusted the rules for designating head words and functional roles to maximize as much as

Simple String Accuracy	98.8%
Exact Match	87.1%

Table 1: Accuracy of regeneration on Section 23 of the Penn Treebank

FEATURE	VALUE
Head	a node id
Role	syntactic function, see Table 3
LogicalRole	captures active/passive voice equivalence
Rolemarker	a string for preposition or complementizer dependent
Direction	+/- from head
Relative position (RP)	absolute distance from head
Absolute position (AP)	distance from start of sent
Junction	a junction id
Junction position	integer
Group type (GT)	clause, np, other
Subject position	default, postaux, final, after-nonaux-finite,
Word	a word as it appears in a leaf node of Treebank
Base	uninflected and uncontracted form of verbs and nouns
Cat Cat0	Original tag Conflated version of tag: 8 tags for closed class words 12 for punctuation 4 open class (v, n, adjs, advs)
Cat1	a subdivision of Cat— verbs: 5 atomic moods nouns: common, proper, or pron adjs: cardinal, possessive-pron, comp, superl, other advs: neg, wh, comp, superl, other
Tense	applies to verbs: past, present
Person/Number (PN)	s, p

Table 2: Main features associated with each word

possible this determinism. Two roles (adverbial and particle) were subdivided into 7 classes to encode their position relative to an optional sibling, and adjuncts and punctuation were marked as occuring either to the left or right of their head. The accuracies on Section 23 of the Penn Treebank, which we used as a blind test set, are shown in Table 1. Simple string accuracy, measuring the number of word insertions and deletions relative to the number of words in the original sentence, is 98.8%. (Displaced words are penalized twice.) Over 87.1% of the 2416 sentences in the regenerated test set were exactly the same as the original sentence. A forthcoming paper describes additional experiments we performed.

3. Representation

The representation used in the corpus is a functional dependency structure where each word token is associated with a uniform set of features, each having a relatively small set of possible values. The most important of these features is summarized in Table 2.

3.1. Head and Role

The *head* and *role* features in Table 2 are the primary elements that represent these functional relationships. The head represents the dependency link to another word token, and the role describes the kind of dependency. The set of surface roles we distinguish are listed in Table 3. They are

Adverbial	Leftpunc	Rightpunc
Aspect	Locative	Rolemarker
Aux	LGS-adjunct	Subject
Beneficiary	Manner	Taxis
Closely-related	Modal	Temporal
Dative	Object	To-infinitive
Determiner	Particle	Top
Dependent	Polarity	Topic
Direction	Pre-determiner	Voice
Extent	Predicate	Withinmod
Junction-marker	Purpose	

Table 3: Roles

a combination of the functional tags annotated on clausal constituents in Treebank and the functional roles implied by part-of-speech tags and nonterminal labels. When no other more specific tag was available from the original annotation, we assign the generic role of "dependent". Note that auxiliary verbs in clauses are treated as dependents of the main verb, if there is one, as described in Section 4.2.. The treatment of prepositions and conjunctions is also nonstandard, and is described shortly.

3.2. LogicalRole

We also derive a *logicalrole* feature that captures the equivalence between active and passive voice sentences by labeling constituents with the role they would have had in an active voice version of the sentence. (In the future, we plan to extend the use of this feature to indicate the deep role of extraposed and ergative subjects as well as topicalized constituents.)

3.3. GroupType

The *grouptype* feature we use is all that remains in our representation from the base nonterminal tags annotated on the Treebank corpus. This feature generalizes all possible nonterminal labels to three gross constituency types: clause, np, and other. We define a clause grouptype to be a node meeting at least one of the following conditions:

- it acts as the head of a subject, object, or dative node that does not have a rolemarker dependent
- it has an auxiliary dependent (ie., a node with a role of aux, modal, taxis, aspect, or voice)
- it is not itself functioning as an auxiliary dependent, and is a verb in indicative mood

Note that all clauses have a verb as their head.
NP grouptypes can only have a noun, adjective, or determiner category as their head. (We automatically correct any nodes that differ from this by reassigning the category of the head token in the NP to be either a noun or an adjective.) NP grouptypes are nodes that either have a determiner, or syntactically can permit one (whether or not it would be semantically appropriate).
All other nodes are assigned a grouptype of 'other'. The above definitions may seem overly narrow by not clearly designating as 'clause' or 'NP' some nodes that might usually be classified as such, instead leaving their classification

ambiguous. However, from our perspective, the classification of these ambiguous cases doesn't matter. We consider this ambiguity to be real, in the sense that it is the sort of ambiguity that lies behind historical shifts in language use over time.

The reason for these definitions is that the grouptype feature has a very specific use in our generator. Since the generator allows an input to leave any (or all) function words unspecified, the grouptype feature is used to signal when the generator needs to consider inserting auxiliaries or determiners. It is also used to provide a context that affects the selection of rolemarkers for clausal dependents. (The grouptype itself can also be left unspecified, and the generator will try all options and compare their probabilistic likelihood.) Even though the grouptype feature is tied to the task of generation, we believe that it will also help sharpen any statistical model of language that makes use of it.

Note that clause and np grouptypes are not necessarily mutually exclusive. Relevant examples include "the has-beens" and "the movie 'Look Who's Coming to Dinner"'.

3.4. Rolemarker

We treat prepositions, complementizers, relativizers, and subordinating conjunctions as role markers, rather than as heads of PP or SBAR phrases as is commonly done. For example, in the sentence "ABC came in first with 8.8%" we make the preposition "in" depend on "first" rather than "came". The node for "in" is given the role of 'rolemarker'. The motivation behind this is to capture more directly the three-way relationship that exists between a preposition, the head noun to its right, and the governing word on the left to which they both relate, without having to overgeneralize and postulate a statistical dependence between every node and its grandparent as has been done recently in statistical parsing (Charniak, 2000; Klein and Manning, 2003).

The existence of this three-way relationship is nicely illustrated by the following example. Suppose in the context of machine translation the computer is given a choice between the prepositions "in" and "to" in the example sentence above. Using a bigram model built from 250 million words of the North American News Text corpus, the computer will make the unfelicitous choice of "ABC came to first with 8.8%". The raw word pair counts from this corpus indicate why this happens, and are shown here:

```
came_to 7559   to_first 931
came_in 3632   in_first 812

      came_in_first 21
      came_to_first  1
```

The bigram counts involving "to" are both higher than the corresponding counts with "in". (Unigram counts are roughly the same for the two prepositions—about 3.6 million.) However, the trigram counts capture the interdependence of the three words, and support our intuition that "in" is the correct choice.

3.5. Junction and Junction Position

Conjunction phrases are treated somewhat uniquely to enhance probabilistic modeling. Besides having a syntactic head as described earlier, heads of conjoined phrases have two additional kinds of relationships: a conjunction anchor and a conjunction sibling. The conjuction anchor is the conjunction itself, and the conjunction sibling points to the immediately preceding conjoined phrase, if any. Conjoined phrases also have a position feature indicating their left-to-right order.

This representation is intended to be redundant to order to accomodate both parsing and generation concerns. It also allows conjunction phrases to connect directly to their syntactic head without the conjunction as an intermediary. The conjunction anchor has the same syntactic head as the conjoined phrases. For example, in the sentence 'The Perch and Dolphin fields will start producing next year," all three of "Perch", "Dolphin", and "and" have the same syntactic head, namely "start". (The word "will" is treated as an auxiliary dependent of "start", as described in Section 4.2..) "Perch" is junction sibling of "Dolphin", and the word "and" is their conjunction anchor.

3.6. Subject Position

The subject position feature only applies to nodes with grouptype clause. Examples for each of the four possible values are shown here. The token to which the feature value applies is bracketed.

- default: Pierre Vinken will [join] the board.
- postaux: How did program trading [evolve] into this?
- after-nonaux-finite: "We have no useful information," [said] James with a sigh.
- final: Corresponding to the fall in profit rates [was]–in the early 1980s–the drop in the Q ratio.

4. Conversion Process

This section briefly describes the procedure that automatically converts sentences from the Penn Treebank into the representation accepted by the realizer. The process of deriving this representation is not as straightforward as one might expect.

At a high level of abstraction, the conversion from a Treebank parse to a functional dependency involves:

• Finding the base forms of words,
• Factoring Treebank categories of open class words into more basic features,
• Heuristically designating constituent heads,
• Inferring syntactic and logical roles for each node,
• Making coordination bracketing more explicit,
• Reorganizing compound prepositions into a single constituent,
• Associating punctuation with a content-bearing constituent,
• Removing null elements, and
• Trickling nonterminal information down to head leaf nodes and then reducing the structure from a constituency to a dependency form,
• Computing relationship IDs and position features.

We describe a few of the more complicated subtasks in the next few subsections.

Sentences in the Penn Treebank are annotated using phrase structure categorial grammar. In contrast, inputs to HALogen use a dependency-style notation. (We find a dependency-style notation to be more suitable as input to generation since the one of the main tasks to be performed is determining linear constituent order given the functional relationship between a pair of words.) Thus, two of the main tasks in constructing inputs automatically from the Treebank annotation are determining the head constituent and labeling the relationship between the head and each other child.

4.1. Base form derivation

One of the most fundamental pieces of information for natural language processing is the base form of any inflected word. The base form is a starting point for generation tasks, and is needed to generalize about the ways that different inflected forms of a word are used. In statistical modeling, use of the base form can help avoid fracturing similar phenomena into different classes and exacerbating sparse data problems.

The question of what to use as the base form is not a simple one, however, since the distinction between a base form and a morphological stem tends to become blurred. For our corpus we decided to lemmatize only nouns, verbs, and contractions ("n't" becomes "not"). The base word is represented in standard dictionary form.

We used the Perl packages Lingua::EN::Infinitive and Lingua::Wordnet to help automatically infer the base form of desired tokens. By combining them we were able to achieve higher quality and broader coverage processing.

We implemented the following procedure:

1. If a word has one or more hypens, split off the part after the last hyphen and use that part for the next few steps.

2. If the word contains non-alphanumberic characters, return the word unchanged as its own base.

3. Map the word to lower case.

4. Use the morph function in the Perl Wordnet package to look up the base form, given the Treebank part-of-speech (p-o-s) tag. If a base is returned, skip the next step.

5. Use the stem function in the Perl Infinitive package to obtain up to two possible base forms. Using the Treebank p-o-s tag, look up each stem in Wordnet. Return the first found there, if any. If neither is found, return the original word.

6. Restore original capitalization.

7. Re-concatenate hypen prefix, if any, to new base.

The morphological features associated with the original inflected word are inferred from the Treebank POS tag.

4.2. Phrase Head Determination

In determining heads of phrases, we use a set of rules we developed that specify for each EPTB non-terminal label how to determine the head of that constituent. The rules specify a set of child labels, such that the first child with one of those labels when searching from the specified end (right or left) of the constituent is to be considered as the head.

Our realizer uses a flat representation of verb phrases, in contrast to the Treebank, so the bracketing must be adjusted accordingly during the conversion. For example, in the clause fragment "it would sell its aging fleet", HALogen would represent "sell" as the head, with "would", "it", and "fleet" as dependents. A sample simplified flat representation of this is:

```
( / sell
  :subject it
  :modal would
  :object fleet)
```

The flattening process is made more complicated in the presence of coordinated VPs however. Since the Treebank's constituent-style annotation does not make explicit whether constituents preceding the verb modify the first or all verbs in a coordinated verb phrase, it is very difficult to accurately flatten coordinated clauses. We compromise by flattening the bracketing of clauses when possible, and leaving a minimal amount of nesting in place for coordinated phrases.

Noun phrase constituents are handled specially because of their more complex (and flatter) structure. The head of an *NP* or *NX* is chosen according to the following rules:

1. The first *CD* from the left – if the phrase is a date and has pattern *NN CD , CD ...* (November 1, 1989)

2. The last constituent – if it is a *JJ*, *CD*, *VB* (not 's), *-NONE-*, or *DT*

3. The last *NN*, *NX-TTL*, or *NX*

4. The last *NP*, *-TTL*, *-HLN*, or *-NOM* without a functional tag

5. The last constituent that is not a punctuation or *POS*, *-RRB-*, *PP*, *S*, *RRC*, *PRN*, *QP*, or that does not have a functional tag.

Theses rules are able to correctly determine that "high" is the head in phrases like *(NP (PRP$ its) (NN session) (JJ high))*. However, the flatness of base noun phrases still leads sometimes to ambiguity errors. For example, the flat annotation of the phrase "Boeing Co. 707s" leads to a dependency representation in which "Boeing" is improperly considered a dependent of "707s", rather than a modifier of "Co."

4.3. Functional Role Assignment

The functional tags annotated in the Treebank allow direct assignment of functional roles for some constituents. Such tags include *-ADV, -BNF, -CLR, -DIR, -DTV, -EXT, -LGS, -LOC, -MNR, -PRD, -PRP, -SBJ, -TMP, and -TPC*. There

Treebank	Surface Role	Conditions
NP	Dative	- parent VP, VB head, followed by an NP, no function tags
NP	Object	- parent VP, VB head, no function tags, preceded by rightpunc, VP,NP,PRT,RB,PP, or comma-delimited PRN
PP	Dative	- has -DTV child
PP	LGS-adjunct	- has -LGS child
"not", "n't"	Polarity	
"to "	To-infinitive	- parent is VP or S, next verb is infinitive
TO	Rolemarker	- precedes head
"have"	Taxis	- parent VP, next verb past-part
"be"	Voice	- parent VP, next verb past-part
"be"	Aspect	- parent VP, next verb pres-part
"do"	Aux	- parent VP, next verb infinitive
IN	Rolemarker	- precedes head
DT	Rolemarker	- parent is SBAR
DT	Determiner	- precedes head; parent not S, VP
PDT	Pre-det.	- precedes head; parent not S, VP
CC	Dependant	- first non-punc const.
CC	Leftpunc	- not first const., not only CC
CC	Junc-marker	- otherwise
CONJP	(same as CC)	
WH*	Topic	- parent is SBARQ
punc	Coordpunc	- separates coord. const.
punc	Leftpunc	- not SYM, is a valid leftpunc
punc	Rightpunc	- not SYM, is a valid rightpunc
	Withinmod	- precedes head, follows subject (if exists)
	Withinmod	- succeeds head, precedes subj.
	Withinmod	- is between "be" verb and -PRD
	Rolemarker	- parent is SBAR, precedes head
	Pre-det.	- precedes head, parent is not S or VP, next const. is DT
	Top	- head of whole sentence; does not depend on anything else
	Dependent	- otherwise

Table 4: Summary of rules to infer surface roles of Treebank constituents. The order of rules is important, since a later rule applies only when an earlier one doesn't. A sample reading is "A constituent with a CC tag has a surface role of dependant if it is the first constituent in the phrase that is not a punctuation."

is some inconsistency in the Treebank annotation, however. The tags *-LGS* and *-DTV* are sometimes attached to the PP, and sometimes to the head NP inside a PP. Only the PP actually gets the role in our conversion. The constituents with base tags *MD* and *PRT* are also directly assigned surface roles corresponding to their tag. In the remaining cases, however, some amount of processing is required to correctly infer the surface syntactic role. Table 4 summarizes the mappings from Treebank information to functional roles.

4.4. Coordinated Phrase Reorganization

Coordinated phrases in the Treebank are represented in a relatively flat way in many instances. Consequently, it is sometimes difficult to determine which constituents are being conjoined and which are modifiers or conjuncts, especially when there are more than two conjuncts separated by commas. Structures such as the following are not uncommon.

```
S-TPC-1 --> ADVP-TMP, PP, S, S CC S
NP --> S-NOM CC NP
```

To handle these sentences, we search for head-like constituents and separate them from any premodifiers, postmodifiers, or complements by adding an additional level of nesting. Defining head-like is problematic, as illustrated in these two examples. In the first, only the base tag matches with the child conjuncts. In the second, NP and S-NOM are conjoined without a UCP (unlike coordinated phrase) parent tag. Over the entire treebank of 50,000 sentences, we add a level of nesting to 5115 conjoined phrases.

Coordinated noun phrases are even more difficult because they are intentionally left very flat to avoid inconsistent semantic interpretation between annotators. Without extra bracketing, the heads of the conjuncts can often be confounded with the head of the NP. Such a sentence also will not generate properly in our system. The following phrase exemplifies this:

Original: [a sales and marketing executive]
Revised: [a [sales and marketing] executive].

Over the entire Treebank, we add extra nesting to an additional 6030 noun phrases.

4.5. Compound Preposition Grouping

In many sentences, prepositions have modifiers, as in *only to* or *more like*. In order to ultimately derive a correct dependency structure for these phrases, we group them together before assigning the role of *Rolemarker*. Additionally, extra nesting is added to phrasal prepositions like "because of" to group them together because they are viewed as functioning as a unit rather than independently in sentences "The company will sell its aging fleet of Boeing Co. 707s because of increasing maintenance costs." In this sentence, "because of" describes the relationship between "sell" and "increasing maintenance costs". We add an extra such level of nesting to compound prepositions 3985 times.

4.6. Punctuation Dependencies

In converting to a dependency representation, punctuation constituents must also be assigned to a head. For punctuation at the periphery of a constituent, this might at first seem fairly straightforward. The punctuation could simply given the role of leftpunc or rightpunc, as appropriate, and attached to the head of the constituent in which it appears.

This approach is somewhat problematic. It would assign a rightpunc comma to every subject with a comma-delimited appositive modifier. Clearly, if the appositive phrase were not there, the punctuation should not be either. Therefore, it seems more appropriate to associate the punctuation with the appositive phrase.

For every punctuation constituent, then, there are up to three options for association: to the right, to the left, or with head of the surrounding bracket. The inherent nature of some punctuation marks, such as an open parenthesis or

closing quote mark, helps simplify this decision sometimes. For instance, an open parenthesis is always a leftpunc that associates with the constituent to its immediate right.

Through corpus analysis we devised a set of association rules for punctuation. We tried to maintain pairings of punctuation if they existed, and avoided crossing associations when several punctuation marks occured adjacent to each other. Otherwise, in general, punctuation appeared to associate with the more adjunct-like of the neighboring constituents rather than the head of the bracket. The more adjunct-like constituents are usually those farther from the head, so our association rules tend to assign outward associations.

4.7. POS Corrections

The semi-automatic process used to assign POS tags to words in Treebank understandably did not tag all of the difficult cases accurately. Some POS tag errors were particularly harmful, however, because they caused other important errors in deducing a head or inferring roles. Some errors conflicted with their parent's grouptype.

Armed with additional syntactic and functional context from the non-terminal labels in the EPTB and from earlier stages in our conversion process, the converter program corrects some of the more critical errors. Twenty kinds of changes are made, for a total of about 3255 corrections over the whole Treebank.

Table 5 lists the POS corrections we perform, and how often they are done. Note that the corrections appear in the conflated and factored category features, while the *cat* feature representing the original Penn Treebank tag was not changed.

Treebank	Correction	Frequency
DT	RB	50
IN	JJ	88
JJ	VB	193
JJ	VBG	29
JJ	VBN	99
NNS	VBZ	163
NN	VB	300
NN	VBG	99
RB	JJ	475
RBR	JJ	35
RBS	JJ	16
UH	RB	10
VBD	NN	9
VBG	NN	288
VB	NN	63
VBN	NN	61
VBN	VBD	639
VBP	NN	8
VBZ	NN	149
WDT	DT	481

Table 5: Corrections in Treebank POS tags and their freqencies

5. Related Work

The most closely related work (in English) to our knowledge is the following: (Rambow et al., 2002) describes the annotation of a functional dependency representation by hand on a smallish corpus of English dialogs; (Oepen et al., 2002) describes work to semiautomatically develop a large English corpus based on HPSG theory; (Cahill et al., 2002) semiautomatically adds LFG annotation on top of the Penn Treebank. The approach for augmenting the Treebank taken in this last reference is very similar to our own in some respects, as is the resulting annotation. However, our conversion process is rather ad-hoc, while they take a more controlled and principled appraoch. On the other hand, our corpus integrates both constituent-related and functional-dependency information into a single hybrid dependency-style representation, and programmatically corrects a number of structural and labeling problems with the original annotation.

6. Summary

We have described a significant effort to convert the Penn Treebank corpus into a functional dependency representation. We expect that this corpus and the software tools built for it will be useful as a resource for the study of English and the development of NLP applications.

7. References

S. Bangalore and O. Rambow. 2000a. Using tag, a tree model, and a language model for generation. In *Proc. of 1st INLG*.

A. Cahill, M. McCarthy, J. van Genabith, and A. Way. 2002. Automatic annotation of the penn-treebank with lfg f-structure information. In *Proc. LREC Workshop on Linguistic Knowledge Acquisition and Representation - Bootstrapping Annotated Language Data*.

E. Charniak. 2000. A maximum-entropy-inspired parser. In *Proc. NAACL*.

D. Klein and C. Manning. 2003. Accurate unlexicalized parsing. In *Proc. ACL*.

I. Langkilde-Geary. 2002. An empirical verification of coverage and correctness for a general-purpose sentence generator. In *Proc. INLG*. http://www.isi.edu/licensed-sw/halogen/verifcc.ps.

M. Marcus, G. Kim, M. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, K. Katz, and B. Schasberger. 1994. The Penn treebank: Annotating predicate argument structure. In *ARPA Human Language Technology Workshop*.

S. Oepen, K. Toutanova, S. Shieber, C. Manning, D. Flickinger, and T. Brants. 2002. The lingo redwoods treebank: Motivation and preliminary applications. In *Proc. COLING*.

0. Rambow, C. Creswell, R. Szekely, H. Taber, and M. Walker. 2002. A dependency treebank for english. In *LREC*.

E. Ringger, M. Gamon, R. Moore, D. Rojas, M. Smets, and S. Corston-Oliver. 2004. Linguistically informed statistical models of constituent structure for ordering in sentence realization. In *Proc. Coling*.