A Labelled Corpus for Prepositional Phrase Attachment

Brian Mitchell and Robert Gaizauskas

NLP Group, Department of Computer Science, University of Sheffield,
Regent Court, 211 Portobello Street, Sheffield, UK, S1 4DP
{brianm,robertg}@dcs.shef.ac.uk

Abstract

This paper describes a labelled corpus intended for training learning algorithms to attach prepositional phrases (PPs). Taken from the PTB2, we believe it is the largest available resource for this purpose, especially as it contains many patterns in which PPs occur ambiguously (nearly all previous research has focused on just one pattern) and we present some results for the five most common patterns. Moreover, the corpus contains some features that, to our knowledge, have not been used before for attaching PPs.

1. Introduction

This paper describes the format of a labelled corpus suitable for training machine learning algorithms (MLAs) to perform prepositional phrase (PP) attachment. We believe it is the largest available resource for this purpose. The paper also describes the conversion of the Penn Treebank II (Marcus et al., 1994) to this format. The PTB2 contains, implicitly and explicitly, the information needed for PP attachment (including some features that, to our knowledge, have not yet been tried for PP attachment) but the distilled form of the information is not yet widely available to the NLP community. This papers reports the results of some experiments carried out on the PP corpus.

To be able to create a genuinely useful resource for training MLAs to disambiguate prepositional attachments accurately, we must first understand PP attachment itself and why it is problematic for computers. Changing the attachment of a prepositional phrase can (dramatically) alter a sentence's meaning, so attaching PPs correctly is essential. Whilst people generally have no problem doing this — headlines such as "*miners refuse to work after death*" may cause us amusement but rarely any difficulty — the best scoring computer models, such as those by Collins and Brooks (1995) and Zavrel et al. (1997), only manage up to 85% accuracy, some way short of the 93·2% accuracy expected of people (Ratnaparkhi et al., 1994). Moreover, all these figures are only for a single pattern of ambiguity, known here as the *canonical form*: a binary choice between attachment to a verb or to a noun. Whilst it is the most common pattern of prepositional ambiguity, other patterns do exist in significant numbers (Table 1) and should be included in a training corpus. Since there is no single (or simple) piece of information that we know of which always identifies the correct prepositional attachment, training data must include a variety of features in the hope that these compensate for the computer's lack of world knowledge.

2. Potentially Useful Data Features

2.1. Data Features Used Previously

Much of the regularly cited work on corpus-based PP attachment has used the bare minimum feature set (and mostly only considered the canonical form of ambiguity). Hindle and Rooth (1993) used just the attachment candidate words (a verb and a noun) plus the preposition. Collins and Brooks (1995) also included the head noun from the prepositional complement, following the lead from Brill and Resnik (1994) and Ratnaparkhi et al. (1994), and both these sets of researchers had further systems that utilised semantic hierarchies: `WordNet` noun synset classes in the former case, an automatically derived hierarchy in the latter. Merlo et al. (1997) used the same triple as Hindle and Rooth for the canonical form but included the noun at the head of a preceding PP's complement when extending this pattern to two and three consecutive PPs. Not one of these systems was given part-of-speech (POS) tags and few use lemmas.

Although not a training feature as such, we also need the attachment point of a PP since supervised learning algorithms must know the "*answer*" for each training and test instance. A corpus annotation scheme might explicitly state prepositional attachment but if it does not then the attachment must be calculated, see §4.1 for a description of this derivation process for the PTB2.

2.2. Novel Data Features

Two (related) features that have not yet been used in published work on PP attachment are the lexical and phrasal distances from a preposition to its attachment point. Counting every word token (but not punctuation) between the two points yields the lexical distance, whilst counting the logical groups of word tokens gives the phrasal distance. It is reasonable to assume that proximity is a possible indicator of prepositional attachment. Both distance measures can be used simultaneously and both will probably have to be calculated since such information does not usually directly appear in a corpus. It is worth noting that the lexical and phrasal distances need not be represented by numeric values, since not all MLAs handle numeric data well: categories like *near*, *medium*, and *far* could be used to represent arbitrary ranges (1–2, 3–6, and 7+ for example).

Another annotation type appearing in the PTB2 is the *phrase function tag* (PFT). 45% of the PPs in the PTB2 have such a tag and we chose to include them in the PP training corpus, though the experiments that we report here did not make use of them. This is because PFTs are not common in other corpora and we wanted to show the utility of a basic set of readily available features. Table 2 shows that these basic features offer good accuracy even for a basic decision tree.

lemmas
$\begin{bmatrix}\text{VG1} & \text{SNP1} & \text{PP1} & \text{COMP1}\\ \texttt{do}, & \texttt{this}, & \texttt{in}, & \texttt{public}\end{bmatrix}$

original words
$\begin{bmatrix}\text{VG1} & \text{SNP1} & \text{PP1} & \text{COMP1}\\ \texttt{done}, & \texttt{this}, & \texttt{in}, & \texttt{public}\end{bmatrix}$

POS tags
$\begin{bmatrix}\text{VG1} & \text{SNP1} & \text{PP1} & \text{COMP1}\\ \texttt{VBN}, & \texttt{DT}, & \texttt{IN}, & \texttt{NN}\end{bmatrix}$

VG head fields
$\begin{bmatrix}\text{PFT} & \text{Lemma} & \text{Orig} & \text{POS}\\ \texttt{null}, & \texttt{do}, & \texttt{done}, & \texttt{VBN}\end{bmatrix}$

SNP head fields
$\begin{bmatrix}\text{PFT} & \text{Lemma} & \text{Orig} & \text{POS}\\ \texttt{null}, & \texttt{this}, & \texttt{this}, & \texttt{DT}\end{bmatrix}$

PP head fields
$\begin{bmatrix}\text{PFT} & \text{Lemma} & \text{Orig} & \text{POS}\\ \texttt{LOC}, & \texttt{in}, & \texttt{in}, & \texttt{IN}\end{bmatrix}$

COMP head fields
$\begin{bmatrix}\text{PFT} & \text{Lemma} & \text{Orig} & \text{POS}\\ \texttt{null}, & \texttt{public}, & \texttt{public}, & \texttt{NN}\end{bmatrix}$

lexical distances
$\begin{bmatrix}\text{VG1} & \text{SNP1}\\ 2, & 1\end{bmatrix}$

phrasal distances
$\begin{bmatrix}\text{VG1} & \text{SNP1}\\ 2, & 1\end{bmatrix}$

attachment
$\begin{bmatrix}\texttt{VG\_1}\end{bmatrix}$

Figure 1: The logical structure of the final CSV format for the PTB data.

3. Corpus Format

The new corpus must be directly suitable for training MLAs. The simplest format to use is comma-separated variable (CSV). The representation we selected is described below. Some fields are repeated: starting each instance with the original words and their lemmas means instances are readily identifiable and problems with lemmatisation are easily spotted. There is no obligation for a MLA to use every field but most MLAs can, effectively, ignore certain features by not using them in the classification process. The ML environment `Weka` also permits the user to deselect certain data features.

In the following descriptions, an *attachment candidate* is an actual word to which the PP in question might attach. For verb groups (VGs), the last verb is the attachment candidate, for other phrases it is the head word. The fields in the PP corpus can be grouped logically as follows:

lemmas: only nouns and verbs were lemmatised with an in-house program, `morph`, used in our Information Extraction systems. There is one lemma for the attachment candidate in each phrase and the lemma is repeated in that phrase's group of fields. The lemmas appear in their own group purely to make each instance easier to identify manually.

original words: the attachment candidates, one per phrase, as they appear in the original document, before lemmatisation. Like the lemmas, these words are repeated elsewhere in the instance and are grouped together purely for human readability.

POS tags: of the original words. If combining several corpora that use different tagsets to create PP attachment instances, POS tags must be translated to a common tagset.

fields for the Verb Group head: the first of these fields is the function tag for the VG or `null` if there is no tag. The second, third, and fourth fields are the lemma, original word, and POS tag of the *grammatical head* (see §4.1) of the VG. Note that the head word may be different from the attachment candidate — for example in "*had thought*" *had* is the head but *thought* is the attachment candidate — though not in the example in Figure 1. For VGs, the attachment candidate can be considered the *semantic head*.

fields for the Simple Noun Phrase head: as for the VG fields.

fields for the Prepositional Phrase head: each PP (there may be multiple PPs in a pattern of ambiguity) is always represented by four fields. These fields are the same as for VGs and SNPs.

fields for the Complement head: for each PP there is a corresponding complement with the same fields, though the values will of course differ.

lexical distances: there is one numeric value for the attachment candidate attachment in each phrase in the pattern.

phrasal distance: as for lexical distances but with the figures for phrasal distances.

attachment: this is the target function value for the MLA. The value is suffixed with a number to identify which phrase is the actual attachment point for the preposition, so in a pattern containing two adjectival phrases, the first is `ADJ_1` and the second `ADJ_2`.

4. Corpus Conversion Process

Having identified some features that we want to use for training MLAs, we now describe the process used to convert the parsed Wall St Journal texts of the PTB2 into input vectors suitable for training a PP attachment algorithm.

The first five stages of conversion maintain the original PTB2 nested bracket structure format, with the new information supplementing existing phrase-level annotations. The remaining stages manipulate existing annotations whilst retaining the PTB2 format, except the last two stages which break away from this format. The conversion process, therefore, produces numerous versions of the original data. Some of the new annotations, such as a PP's attachment point, refer to another bracket in the same sentence. We numbered each bracket sequentially, resetting the count to 1 at every new sentence (or equivalent top level structure), so a reference to `29` indicates the twenty-ninth bracket in the current sentence. These references do not appear in the final training data.

4.1. Identifying Head Words

No current version of the Penn Treebank has explicit annotations indicating prepositional attachment. But if we assume that the attachment point is the head word of a PP's parent phrase, then we can derive an attachment for nearly all 117,822 PPs in the PTB2 — instances were the PP is first in a sentence are ignored since, bracketing-wise, the parent is generally the sentence-level bracket itself. The heads were found using the rules adapted by Collins and Brooks (1995) from Magerman (1994). Although neither of those publications offers an evaluation of the accuracy of the rules (and, of course, there is no consensus on exactly which word is the "*head*" in certain cases, for example co-ordination), a manual checking of several hundred examples showed acceptable behaviour. The rules for finding heads do not always lead directly to lexical items: since the head of a phrase must be one of its direct children, many phrases have another phrase as their head, though they can be traced down to the lexical level. Therefore, we prefer to refer to these annotations as *head containers*.

4.2. Identifying Prepositional Complements

This stage indicates which phrase is a PP's complement and simplifies the later stage of conversion to machine learning input vectors. A `-PC=`*nn* annotation (where *nn* is the relevant bracket number) is appended to any annotations already on the current PP annotation.

4.3. Identifying PP Attachment Points

Having identified head containers, PP attachment points (taken to be the head word of a PP's parent) can be found. This process involves finding a PP's parent and then tracing the head containers down to the lexical level. The bracket number of the attachment point is noted explicitly in a `-AP=`*nn* annotation appended to the PP annotation.

4.4. Grouping Verbs

The fourth stage involves collecting verb phrases (VPs) into verb groups (VGs). Each verb in the PTB annotation scheme commands its own bracket, so consecutive verbs look like this:

```
(VP (MD would)
  (VP (VB have)
    (VP (VBN convicted) ... )))
```

To make the output resemble that from a chunking parser consecutive verbs must be collected into a single group:

```
(VP (VG (MD would) (VB have) (VBN convicted)) ···)
```

which involves removing the brackets belonging to all but the first VP whilst keeping the encompassing VP bracket so that the scope of the verb phrase is retained. So after grouping, the data contains a mixture of VPs and VGs but every verb directly belongs to a VP. A verb which is not adjacent to any other (for example if an adverb intersects them) appears in a VG on its own. Attributes such as `-AP` are updated during verb grouping because the number of brackets changes.

4.5. Identifying Simple Noun Phrases

This straightforward conversion phases makes the text look more like the output of a chunking parser. All noun phrases containing no sub-phrases are relabelled `SNP` for simple noun phrase (sometimes referred to as *base NPs* in other literature).

4.6. Chunking

The parse trees in the PTB were adapted to resemble the output from a chunker. The emulation works by printing any phrase containing at least one lexical item (not counting certain categories such as `-NONE-`, commas, brackets and other punctuation, as well as co-ordinating conjunctions), but excluding the first phrase embedded within a prepositional phrase. This strategy chunks examples such as "*the users* *of the produce* *from last year*" into three pieces (underlined separately).

Number	Pattern	Instances
1	`VG SNP PP`	22,816
2	`VG PP PP`	6,345
3	`VG SNP PP PP`	6,266
4	`VG ADJP PP`	2,360
5	`VG SNP SNP PP`	2,172

Table 1: The five most common patterns of potential PP ambiguity found in the chunked PTB2.

4.7. Identifying Patterns of Potential Ambiguity

This step does not manipulate the corpus as such, but analyses the output of the chunking phase. The previous stage produces a list of phrase names which not only have their annotations (such as `-AP=`*nn*) intact but are now embellished with indices so that the correct text is recoverable from the PTB2 files. Ignoring these indices and annotations for brevity, we have a series of patterns, one for each top-level sentence. For example

`S` $\longrightarrow$ `SNP ADJP SNP VG SNP PP SNP`

which describes the first sentence in the PTB. These grammar patterns must be parsed to identify any prepositional phrase with a potentially ambiguous attachment. The full sentence pattern is split into parts separated at the VGs because we assume that PPs do not attach across groups of verb — in the PTB2 only 0·98% of PPs attach across the first VG in front of a PP.

Given the assumption about attaching across VGs and having segmented the sentence at the VGs, a PP is deemed to be potentially ambiguous if it is not first or second in its segment. The end of the pattern of potential ambiguity is taken to be the last PP in that segment. This allows for multiple PPs, for example `VG SNP PP PP`, though they need not be consecutive, for example `VG PP SNP PP`. However, such patterns can themselves contain more cases of potential ambiguity. Once the second PP from the pattern `VG SNP PP PP` has been attached, the pattern collapses to become `VG SNP PP` which itself contains a potentially ambiguous PP. This method of collapsing patterns finds over 124,500 ambiguous PP instances. A benefit of the method is an increased number of examples for the more common patterns, especially the canonical form, for which nearly 23,000 examples were attained. Table 1 lists the five most common patterns of PP potential ambiguity discovered in the chunked version of the PTB2, though there are many more.

4.8. Conversion to Input Vectors

The last stage of actual corpus conversion takes the instances of the most numerous patterns discovered during the analysis phase (Table 1), recovers the relevant data from the modified PTB2 (stage 5, SNPs), and writes the data to the CSV format described in §3. Most of the required features have been explicitly marked by now, though the lexical and phrasal distances must be calculated here. The field names for each pattern are written to separate header files, so they are not mistakenly treated as instances.

	Ungeneralised Data	Generalised Data
Pattern 1	82·9041%	83·3370%
Pattern 2	96·2480%	95·9217%
Pattern 3	80·6632%	80·5155%
Pattern 4	78·7804%	78·9123%
Pattern 5	85·7796%	82·5863%

Table 2: Some results on the new corpus for the ID3-gain ratio algorithm.

5. Experiments Using the New Corpus

The utility of the PP attachment corpus can be demonstrated by some experimental results. Thousands of (stratified 10-fold cross-validation) experiments have been carried out on the converted PTB2 (Mitchell, 2004) though for brevity only a few are reported here, see Table 2. The decision tree was a custom implementation based on Mitchell (1997). It uses ID3 (Quinlan, 1986), one of the simplest algorithms for decision trees since it does not use pruning. Our implementation uses gain ratio in its calculations, a measure which penalises features that have a large range of values. In terms of PP attachment, this forces the algorithm to prefer the preposition as the primary discriminant for partitioning the data because it is a *closed class* of words whereas nouns, verbs, adjectives, and adverbs are *open class* words and, as such, are potentially limitless sets. We have previously shown (Mitchell and Gaizauskas, 2002) that an ID3 gain ratio decision tree has comparable PP attachment accuracy (for Pattern 1, the canonical form) to more sophisticated algorithms such as EDTBL (Brill and Resnik, 1994) which scored 80·8%, though on a smaller data set derived from the PTB1. Using a larger set also from the PTB1 (Ratnaparkhi et al., 1994), Collins and Brooks (1995) attained 84·6% (using backed-off estimation) and Zavrel et al. (1997) 84·4% (using *k*-Nearest Neighbours). This data set has nearly the same number of training instances for Pattern 1 as those in our novel PP corpus. Merlo et al. (1997) (also using backed-off estimation) attained 69·9% accuracy for Pattern 3, a verb, a noun, and two consecutive PPs.

A second version of the PP training data was created where numbers and proper names were converted to generic keywords, following the procedure outlined in Brill and Resnik (1994). Analysis of the results using paired t tests (recommended by Dietterich (1998)) shows that generalising the data (converting names and numbers to keywords) offers no statistically significant advantage and the half percent increase in accuracy reported by Collins and Brooks (1995) was probably due to the data order in their single run of the experiment. Intuitively, however, we feel that for live language engineering systems, generalising names and numbers is a sensible strategy.

6. Discussion

This paper introduced a resource suitable for training MLAs for PP attachment. The corpus' format, comma-separated variable, makes it readily usable by algorithms and ML environments such as Weka. The method for constructing the PP corpus means it contains more instances than other widely available resources. This is important because MLAs generally increase in accuracy with more training data. The corpus also contains features that we believe have not been used before for PP attachment: lexical and phrasal distances and phrase function tags. Moreover, it contains patterns of prepositional ambiguity that have not been widely covered in previous research, if at all. We also briefly presented some experiments on the corpus, though many more were carried out using more algorithms on more patterns of ambiguity than there is space to include. The results of all these experiments were analysed statistically and, surprisingly, generalising the data (converting names and number to keywords) offers no significant gain, though we still recommend the practice.

The code to convert the PTB2 to the PP corpus is available via the Sheffield NLP website: nlp.shef.ac.uk.

7. References

Eric Brill and Philip Resnik. 1994. A Rule-Based Approach to Prepostional Phrase Attachment Disambiguation. In *Procs. of COLING-94*.

Michael Collins and James Brooks. 1995. Prepositional Phrase Attachment through a Backed-Off Model. In *Procs. of the Third Workshop on Very Large Corpora*.

Thomas G. Dietterich. 1998. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Computation*, 10(7):1895–1923.

Donald Hindle and Mats Rooth. 1993. Structural Ambiguity and Lexical Relations. *Computational Linguistics*, 19(1):103–120.

David M. Magerman. 1994. *Natural Language Parsing as Statistical Pattern Recognition*. Ph.D. thesis, Stanford University.

Mitchell P. Marcus, Grace Kim, Mary A. Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The Penn Treebank: Annotating Predicate Argument Structure. In *Procs. of the Human Language Technology Workshop*.

Paola Merlo, Matthew W. Crocker, and Cathy Berthouzoz. 1997. Attaching Multiple Prepositional Phrases: Generalized Backed-off Estimation. In *Procs. of Second Conference on Empirical Methods in Natural Language Processing*.

Brian Mitchell and Robert J. Gaizauskas. 2002. A Comparison of Machine Learning Algorithms for Prepositional Phrase Attachment. In *Procs. of LREC-2002*, Las Palmas de Gran Canaria.

Tom M. Mitchell. 1997. *Machine Learning*. McGraw-Hill.

Brian Mitchell. 2004. *Prepositional Phrase Attachment using Machine Learning Algorithms*. Ph.D. thesis, University of Sheffield.

J. Ross Quinlan. 1986. Induction of Decision Trees. *Machine Learning*, 1(1):81–106.

Adwait Ratnaparkhi, Jeff Reynar, and Salim Roukos. 1994. A Maximum Entropy Model for Prepositional Phrase Attachment. In *Procs. of the ARPA Workshop on Human Language Technology*.

Jakub Zavrel, Walter M. Daelemans, and Jorn Veenstra. 1997. Resolving PP Attachment Ambiguities with Memory-Based Learning. In *Procs. of CoNLL 1997*.