Using Graphical Models for PP Attachment

Anders Søgaard
Center for Language Technology
University of Copenhagen
Njalsgade 142
DK-2300 Copenhagen
soegaard@hum.ku.dk

Abstract

PP attachment has attracted considerable interest the last two decades. The standard set-up has been to extract quadruples of verbs, direct objects, prepositions and prepositional complements from the Wall Street Journal and classify them as either low or high attachments. State-of-the-art results almost equal human performance in the standard set-up. Recently, however, Atterer and Schütze (2007) has questioned this methodology. In this paper, we show that state-of-the-art results can be achieved by simpler means than what has previously been shown, using graphical models, but also that state-of-the-art parsers perform insignificantly worse than state-of-the-art PP attachment classifiers. This questions the usefulness of previous studies of PP attachment, even if the methodology in these studies is sound.

1 Introduction

One of the main challenges in parsing has for a long time been assumed to be the resolution of ambiguity. One frequently studied type of ambiguity is prepositional phrase (PP) attachment. Given a quadruple of a verb, a direct object, a preposition and a prepositional complement (the head of the NP2 embedded in the PP), PP attachment – or PP re-attachment – is the task of determining whether the PP should attach to the verb (V) or the direct object (N). PP attachment is thus construed as a binary classification problem, typically with labels N and V.

The standard features for PP-attachment used in Ratnaparkhi et al. (1994) and subsequent studies are listed in Figure 1. The seven features are the ones in rows 2–8. A distributional cluster is a set of words that have similar distributions according to a hierarchical clustering algorithm, typically based on probabilities in a bigram language model. The granularity of clusters varies, but we will use a 1000 clusters in our experiments below.

Example. To see the complexity of this learning problem, consider the following four examples:

(1) (Andy Warhol) painted paintings with 3D-glasses.

(2) (Andy Warhol) painted [portraits with 3D-glasses].

(3) (Andy Warhol) painted [paintings with ice-cream].

(4) (Andy Warhol) painted portraits with ice-cream.

The square brackets indicate likely low attachment, i.e. that the prepositional phrase most naturally modifies the noun. Note that the four data points have an XOR-like distribution in two-dimensional space:

3D-glasses	V	N
ice-cream	N	V
	paintings	portraits

The distributional cluster features are of no help here, since there is a function from words to clusters. In sum, the learning problem has few dimensions, but variables are highly interdependent.

The interdependence of the variables given the class, illustrated by the examples above, was initially ignored in studies such as Ratnaparkhi et al. (1994) and Collins and Brooks (1995). However, it was the interdependence of the variables that motivated Toutanova et al. (2004) (who report the best result

Bolette Sandford Pedersen, Gunta Nešpore and Inguna Skadiņa (Eds.)
NODALIDA 2011 Conference Proceedings, pp. 206–213

1	label (N or V)
2	verb
3	verb (distributional cluster)
4	direct object
5	direct object (distributional cluster)
6	preposition
7	prepositional complement
8	prepositional complement (distributional cluster)

Figure 1: Standard features in PP-attachment.

in the literature on the standard Wall Street Journal dataset for English PP attachment) to consider graphical models.

The most important of our seven features is undoubtedly the prepositions (feature 6). This has been noted before, for example by Collins and Brooks (1995) who base their backed-off estimate on this observation (Sect. 6): "A key observation in choosing between these tuples is that the preposition is particularly important to the attachment decision." Moreover, selecting PP attachment site only by the preposition typically provides a relatively strong baseline. This observation was also central in the models used by Toutanova et al. (2004).

Previous studies are reviewed in Sect. 2. Atterer and Schütze (2007) recently questioned the methodology used in these studies, however. We briefly summarize their discussion in Sect. 3.

Sect. 4 presents two PP attachment algorithms based on graphical models that are simpler than the one proposed by Toutanova et al. (2004), yet perform as well as theirs. This is an interesting empirical result independently of whether the methodology in re-attachment studies is sound or not.

Sect. 5 qualifies the discussion in Atterer and Schütze (2007), showing that state-of-the-art dependency parsers perform about as well as state-of-the-art re-attachment classifiers on the standard PP attachment dataset. This strengthens the claim in Atterer and Schütze (2007) that PP re-attachment studies are of little practical relevance.

2 Previous studies

Collins and Brooks (1995) use a simple backed-off estimate for modeling PP attachment. In a way similar to nearest-neighbor learning, they first look for identical quadruples in the training data, then for triples and then for pairs. They report a score of 84.5% on the Wall Street Journal dataset. This is identical to the score reported with cross-product features above. It is also similar to what was reported in Abney et al. (1999) (84.6%). It is also similar to what was achieved by Vanschoenwinkel and Manderick (2003) using SVMs and kernel methods (84.8%).

Toutanova et al. (2004) present an approach to PP attachment similar to ours. They manually construct a Markov chain Bayesian network with only two independence assumptions; namely, that given a verbal attachment, the second noun is independent of the first noun, and that given a nominal attachment, the second noun is independent of the verb. The parameters of the graphical model are learned discriminatively by random walks. In addition to the training data, Toutanova et al. (2004) use large amounts of automatically parsed quadruples from the BLLIP corpus (Charniak, 2000). They also use morphological analysis and WordNet features to achieve the best reported results in the literature. Without these additional resources they report an accuracy of 85.9%. Using morphological analysis they achieve 86.2%, and using all features, incl. WordNet features, they achieve 87.6% which is very close to human performance. One of the algorithms proposed below will be similar to this algorithm, but we will learn the graph structure automatically from the training data using no additional resources.

In this paper we will only consider results obtained without additional manually constructed resources. It should be mentioned that Zhao and Lin (2004) report a result of 86.5% on this data set, using more advanced distributional clusters and nearest neighbor classification. Their study is exploratory, and they report several results for the test

data, ranging from 83.1% to 86.5%.

Finally, there has been some work in what has been referred to as "unsupervised" PP attachment (Ratnaparkhi, 1998; Kawahara and Kurohashi, 2005). The idea in this work is to extract unambiguous examples of triples or quadruples from large amounts of raw data, typically based on automatically inferred part-of-speech. The methods subsequently used are standard supervised techniques, so in a way this can be seen as supervised learning with outlier detection on large amounts of data. It is therefore not that surprising that results are good. Ratnaparkhi (1998), who pioneered this approach, report an accuracy of 81.9% (compared to 81.6% in his earlier work), and Kawahara and Kurohashi (2005) report an impressive 87.3%, but in an explorative study where different results are reported.

3 Is re-attachment sound?

Previous studies of PP attachment have, as should be clear now, assumed an oracle that provides preliminary syntactic structure, i.e. extracts the relevant quadruples from gold-standard parse trees. The task is then to re-attach the involved PPs. PP attachment classifiers perform considerably worse in the absence of oracles, however.

Atterer and Schütze (2007) present empirical evidence for this and discuss the difficulties with the standard methodology of using extracted quadruples. One problem arises if the parser does not find the direct object and the prepositional phrase and then does not recognize the ambiguity. It is also harder to decide low or high attachment if the head words of the direct object and the prepositional complement are not correctly identified. Accuracy without oracles drops about 5%.

Atterer and Schütze (2007) also argue that state-of-the-art parsers do almost as good attachment decisions as re-attachers in realistic scenarios, i.e. in the absence of oracles. In Sect. 5 we show that state-of-the-art *also* do almost as good attachment decisions as attachment classifiers when oracle quadruples are provided.

4 Using graphical models for PP attachment

Graphical models (Jordan, 1998) are a happy marriage between probability theory and graph theory, and graphical models are best understood as a framework for talking about and generalizing over various known models, including mixture models, factor analysis, hidden Markov models, Kalman filters and Ising models. Graphical models are also ways of compactly representing joint probability distributions. Their discriminative analogues, conditional random fields (Lafferty et al., 2001), which model conditional probabilities directly rather than joint probabilities, will also be included under the term graphical models here. We will refer to generative graphical models as Bayesian networks, and to discriminative graphical models as conditional random fields. In general, graphical models are graphs in which each node represents a variable whose distribution is to be inferred, and edges represent dependencies.

4.1 Bayesian networks with cross-product of features

Using Bayesian networks we need a method for learning directed graphs over our variables and a method for doing inference in them. This is in contrast to Toutanova et al. (2004) who designed the Bayesian network by hand, guided by linguistic intuition. In the experiments below, we focus on *hill climbing* for learning graphs (Jordan, 1998). Estimation or inference is simple. The choice of using hill climbing is primarily motivated by replicability and computational efficiency, but our initial experiments showed that more advanced methods such as K2 or conditional independence tests did not lead to better results on development data. We also restrict ourselves to Bayesian networks where all nodes have at most one parent. In other words, our graphical models are unordered trees. Experiments showed that allowing for two or three parents did not lead to better results either. This means that our graphical models are much simpler than the ones used in Toutanova et al. (2004).

The *estimation* method is simply to compute

$$\arg\max_y P(Y = y|\mathbf{parents}(Y))$$

1	x_i
2	x_{i+1}
3	x_{i+2}
4	$c_i(x_i)$
5	$c_{i+1}(x_{i+1})$
6	$c_{i+2}(x_{i+2})$
7	$c_j(x_{i-1})$

Figure 2: Feature template used in all experiments for classifier c_j.

Finally, we take the cross-product of the standard features (Figure 1). Since some of these feature pairs may be irrelevant, we use a particle swarm approach to assign weights to each feature. In particular, we use the RapidMiner 4.6 implementation of particle swarm feature weighting with a default parameter setting of 40 generations of size 6.[1]

4.2 Stacked conditional random fields

In conditional random fields, each node in our graphs has an exponential family distribution. Some variables are observed, whereas others are to be inferred (in our case using a quasi-Newton approach). In classification, there is really only one hidden variable, but we will pretend that there are as many as there are observable variables. The new hidden variables will just be copies of this variable.

In particular, we will assume chain graphs, i.e. a subclass of the graphs considered in Bayesian networks. In addition, we will condition each hidden variable on the corresponding observed variables and the two *succeeding* observable variables.[2] The full set of features is presented in Figure 2. A feature $c_i(x_i)$ means the prediction of the classifier trained on sequences of length i prediction for node i (its final prediction). Note that these features are orthogonal to the features of the underlying classification problem (Figure 1). The features of the classification problem only affect the length of the sequences that are given to our stacked sequential labeler.

Our learning algorithm C2C transforms an n-dimensional classification problem to a sequence labeling task for sequences of m length with $m \leq n$. In some cases we will group two or more variables together in nodes, which is why m may be smaller than n. This happens, for example, if we use both word forms and distributional clusters to represent words in PP attachment (in a way similar to Ratnaparkhi et al. (1994)). Each node is then represented by two features, so $m = \frac{n}{2}$. Note also that all sequences in our data will have the same length, i.e. the number of attributes in the original classification data set. Note also that the special case where we group n variables together reduces to standard classification.

Otherwise (when $m = n$) we will transform a classification data set with data points:

$$y^i \quad x_1^i \quad x_2^i \quad \dots \quad x_n^i$$

into a sequence labeling task for sequences $x_1^i \dots x_n^i$.

The idea is then to train n many models; the smallest model will be trained on sequences of length 1 (the first feature x_1^i paired with the class label), the next smallest model on sequences of two nodes (x'_1, x'_2), and so on. The largest model will be trained on the full length sequences. Each node will be augmented with a prediction feature initialized as 'NULL' for all nodes. Each model on sequences of length j will be used to set the value of the prediction feature of node i in each sequence, basically representing the class prediction this far in the sequence.

So our model is in a way similar to ensemble-based methods or corrective modeling, i.e. a form of stacking (Wolpert, 1992). The smallest model in PP-attachment may try to guess low or high attachment based on the verb, for example. The next model sees the next word, say the object noun, but also the class predicted by the smaller model and tries to guess whether the PP attaches to the verb or to the object noun. The next model then sees the preposition also and possibly corrects the guess, and finally the larger model produces the final class prediction. Note that the bigger models when predicting the label of node x_j^i *both* have access to the predictions of the smaller models and its own last prediction for x_{j-1}^i. Herein lies the strength of our model.

The overall algorithm is sketched in Figure 3. In the first line, we simply rewrite our classification training and test sets, respectively T_0 and T_1, as sequential labeling data sets. If $m = n$, this amounts to pairing the class label with all attributes and see-

[1] http://rapid-i.com/

[2] The intuition here is that the preceding observable variables are reflected in the previous classifier's predictions; see below.

1: $T_t = \tau(T_0)$, $T_s = \tau(T_1)$
2: **for** $1 \leq i < m$ **do**
3: $c_i = \mathbf{train}(\{\mathbf{x}_1 \dots \mathbf{x}_i | \mathbf{x}_1 \dots \mathbf{x}_m \in T_t\})$
4: $y_1 \dots y_i = c_i(\mathbf{x}_1 \dots \mathbf{x}_i)$
5: $\mathbf{x}_i[i] = y_i$ # update prediction feature
6: **end for**$c_m = \mathbf{train}(T_t)$
7: **for** $\mathbf{x}_1 \dots \mathbf{x}_m \in T_s$ **do**
8: **for** $1 \leq i < m$ **do**
9: $y_1 \dots y_i = c_i(\mathbf{x}_1 \dots \mathbf{x}_i)$
10: $\mathbf{x}_i[i] = y_i$ # update prediction feature
11: **end for**
12: $y_1 \dots y_m = c_m(\mathbf{x}_1 \dots \mathbf{x}_m)$
13: **return** y_m
14: **end for**

Figure 3: C2C.

ing the pairs of attributes and class labels as sequences. We then do $m-1$ iterations. In each iteration, we train a sequential labeler c_i on sequences of i length in T_0, $1 \leq i < m$ (line 3). The sequential labeler is then applied to partial sequences, and the final prediction (for node i) is used to update the prediction feature of this node. This information is then used for training the classifier in the next iteration. We return the prediction for the last node in the sequences of length m in the test data.

4.3 Data

In our experiments we use the PP attachment dataset presented in Ratnaparkhi et al. (1994).[3] The dataset contains 20,801 quadruples from the syntactically annotated Wall Street Journal (Penn Treebank 0.5) with attachment decisions for training, 4,039 for development, and 3,097 for testing. Each quadruple consists of a verb, a direct object, a preposition and a prepositional complement, e.g.:

prepare	dinner	for	family	(V)
shipped	crabs	from	province	(V)
ran	broadcast	on	way	(N)
is	apartment	with	floors	(N)

Remember the label N means low attachment, while V means that the preposition is a complement of the verb. The quadruples are extracted from the Wall Street Journal relying on manual annotation. Consider some suggested lower and upper bounds on this dataset:

	Acc (%)
Majority baseline	59.0
Most likely for each preposition	72.2
Human (quadruples)	88.2
Human (sentences)	93.2

Majority baseline is the accuracy of a system that always predict low attachment. Here, 'Most likely for each preposition' means use the attachment seen most often in training data for the preposition seen in the test quadruple. The human performance results are taken from Ratnaparkhi et al. (1994), and are the average performance of three treebanking experts on a set of 300 randomly selected test events from the Wall Street Journal corpus, first looking at the four head words alone, then using the whole sentence. The results are thus not directly comparable to those obtained using the test section.

Ratnaparkhi et al. (1994) use the standard features listed in Figure 1, but also suggest to use n-gram features ($1 \leq n \leq 4$) over words and Brown clusters. Ratnaparkhi et al. (1994) used logistic regression to learn from these presentations.

In our experiments, we only use unigrams. Our feature vectors are therefore very short; we use eight features in the standard representation, namely words and clusters of verbs, direct objects, prepositions and prepositional complements, and four and ten features in the two alternative representations. Actually, since all prepositions belong to the same cluster in our hierarchical clustering, we only need to consider seven variables in the standard representation. For reproduceability, we use the Brown clusters available on the website that accompanies Turian et al. (2010) with $C = 1000$, to build our feature representations.

4.4 Results

We compare Bayesian networks and C2C with previous studies of PP attachment. Ratnaparkhi et al. (1994) used logistic regression on standard features. Since we do not use exactly the same hierarchical clusters as they did, we include both his reported results and results obtained with our feature representations using generalized iterative scaling (GIS).[4] Our graphical models are learned using

[3] ftp://ftp.cis.upenn.edu/pub/adwait/PPattachData/

[4] http://homepages.inf.ed.ac.uk/lzhang10/LogReg_toolkit.html

hill climbing and with the restriction that each node has at most one parent (P=1).

Our results are presented in Figure 4. Note that both our approaches perform as good as the approach in Toutanova et al. (2004).

5 How well do parsers perform?

PP attachment became an interesting topic with Ratnaparkhi et al. (1994) at a time where most parsers were grammar-based, and statistical PP attachment was necessary for ambiguity management. Statistical parsers of course do PP attachment themselves finding the most probable parse, but re-attachment may still improve the overall quality of parsers if attachment classifiers are trained specifically to deal with this problem.

Atterer and Schütze (2007) showed that attachment classifiers are only a little better than modern statistical parsers in realistic scenarios where quadruples are not known. Here we show that modern parsers are almost as good as re-attachment classifiers, even when quadruples *are* known.

To cast the re-attachment task as a dependency parsing problem, we convert the labeled quadruples into dependency structures the following way:

In a low attachment the preposition, which is the head of the prepositional complement and thereby head of the PP, is a dependent of the noun, and the dependency between them is labeled MOD for modifier. If the label is V, the preposition is a dependent of the verb, and the dependency is labeled OBL for oblique.

Consider, for example, the dependency structure in Figure 5. The black dependencies are correct and predicted dependencies. The blue dependency is correct, and the red is predicted. In other words, this quadruple, which is the first in the test section of our dataset, is annotated as V, but a low attachment is predicted.

The information available to the dependency parser in Figure 5 is words and hybrid POS tags and clusters. POS tags are obviously redundant on their own. In general, we tried three different feature representations: using only words, using words and clusters, and using words and hybrid POS tags and clusters. Note that the hybrid setting is similar to what was used for semi-supervised dependency parsing in Koo (2008). This setting led to the best results on the development data.

Parser	**Acc (%)**	**Toutanova et al.** (significance)
MaltParser	83.1	< 0.01
MSTParser	84.1	> 0.05
Opt. MaltParser	**85.1**	> 0.05
Bikel	83.7	< 0.05

Figure 6: PP re-attachment accuracy of state-of-the-art parsers.

	FORM	CPOSTAG
Input	0,1	1
Stack	0,1	0,1

Figure 7: Features used in Opt. MaltParser. 0 is the first word on the buffer (Input) or stack; 1 the second, and so on.

We trained two different parsers on the converted PP attachment dataset, namely MaltParser (Nivre et al., 2007) and MSTParser (McDonald et al., 2005). We report results for the two parsers with default parameters and a result for MaltParser with a feature model partially optimized on development data; all results are listed in Figure 6. The optimized feature model is presented in Figure 7.[5] The third column reports significance compared to the results in Toutanova et al. (2004), using χ^2 test. The result of the Bikel parser (Bikel, 2004) is taken directly from Atterer and Schütze (2007).

It follows that state-of-the-art dependency parsers are *not* significantly worse than state-of-the-art PP attachment classifiers. This questions the usefulness of PP attachment classifiers in statistical parsing.

6 Conclusions

We have contributed to the PP attachment literature in two ways. First we have presented two new algorithms that equal state-of-the-art in their performance on the standard Wall Street Journal dataset. One is based on Bayesian networks, the other is based on conditional random fields. Second we have strengthened the claim in Atterer and Schütze (2007)

[5] Optimization was performed by greedily removing features from the default feature model for the arc-eager parsing algorithm. All other features were left unoptimized.

Learning algorithm	Method	Params/Ref	Acc (%)
LogReg	GIS	Standard	80.4
LogReg	RRRR94	Standard	81.6
Backed-off estimate	CB95	Words	84.5
Boosting	ASS99	Standard	84.6
Bayesian networks	TMN04	Words	**85.9**
Bayesian networks	Hill climbing, P=1	Cross-product	85.8
C2C	-	Standard	**85.9**
Human (quadruples)			88.2

Figure 4: Results.

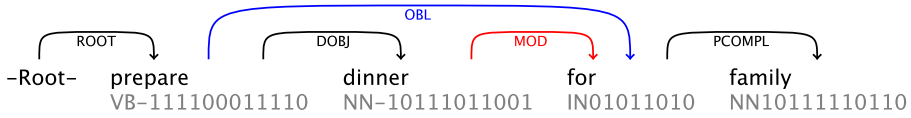

Figure 5: PP re-attachment as parsing problem.

that statistical parsers can do PP attachment almost as good as specialized classifiers in the absence of oracles. In fact, we have shown that state-of-the-art dependency parsers are insignificantly worse than such classifiers even in the presence of oracles. In particular, we showed that the difference between a statistical parser and the best re-attachment classifiers was less than 0.8%.

References

Steven Abney, Robert Schapire, and Yoram Singer. 1999. Boosting applied to tagging and PP-attachment. In *EMNLP*.

Michaela Atterer and Hinrich Schütze. 2007. Prepositional phrase attachment without oracles. *Computational Linguistics*, 33(4):469–476.

Daniel Bikel. 2004. Intricacies of collins parsing model. *Computational Linguistics*, 30(4):479–512.

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

Michael Collins and James Brook. 1995. Prepositional phrase attachment through a backed-off model. In *Workshop on Very Large Corpora*.

Michael Jordan, editor. 1998. *Learning in graphical models*. MIT Press.

Daisuke Kawahara and Sadao Kurohashi. 2005. Pp-attachment disambiguation boosted by a gigantic volume of unambiguous examples. In *IJCNLP*.

Terry Koo, Xavier Carreras, and Michael Collins. 2008. Simple semi-supervised dependency parsing. In *ACL*.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *ICML*.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005. Non-projective dependency parsing using spanning tree algorithms. In *HLT-EMNLP*.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. MaltParser: a language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Adwait Ratnaparkhi, J Reynar, and S Roukos. 1994. A maximum entropy model for prepositional phrase attachment. In *ARPA Workshop on Human Language Technology*.

Adwait Ratnaparkhi. 1998. Statistical models for unsupervised prepositional phrase attachment. In *COLING*.

Kristina Toutanova, Christopher Manning, and Andrew Ng. 2004. Learning random walk models for inducing word dependency distributions. In *ICML*.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *ACL*.

B Vanschoenwinkel and B Manderick. 2003. A weighted polynomial information gain kernel for resolving pp attachment ambiguities with support vector machines. In *IJCAI*.

David Wolpert. 1992. Stacked generalization. *Neural Networks*, 5:241–259.

S Zhao and Dekang Lin. 2004. A nearest-neighbor

method for resolving PP-attachment ambiguity. In *IJCNLP*.

ISSN 1736-6305 Vol. 11
http://hdl.handle.net/10062/16955