

Learning Domain Theories

STEPHEN G. PULMAN & MARIA LIAKATA

Computational Linguistics Group, Centre for Linguistics, Oxford University

Abstract

By a ‘domain theory’ we mean a collection of facts and generalisations or rules which capture what commonly happens (or does not happen) in some domain of interest. As language users, we implicitly draw on such theories in various disambiguation tasks, such as anaphora resolution and prepositional phrase attachment, and formal encodings of domain theories can be used for this purpose in natural language processing. They may also be objects of interest in their own right, that is, as the output of a knowledge discovery process. We describe a method of automatically learning domain theories from parsed corpora of sentences from the relevant domain.

1 Domain theories

The resolution of ambiguity has been a perennial topic of interest among linguists and computational linguists. In the early days of generative linguistics, the existence of various types of ambiguity was used to justify abstract levels of linguistic structure:

- (1) Flying planes can be dangerous.
- (2) He saw the man with the telescope.

These examples show, respectively, that the same sequence of words can have different assignments of parts of speech (lexical categories), and that the same sequence of parts of speech can have differing constituent structures associated with them. However, after some initial flirtations with ‘semantic features’, the problem of resolving such ambiguities in a given context was not seriously pursued, since it was regarded as what we would now call ‘AI-complete’, requiring the encoding of salient features of the context as well as vast amounts of non-linguistic knowledge:

For practically any item of information about the world, the reader will find it a relatively easy matter to construct an ambiguous sentence whose resolution requires the representation of that item (Katz & Fodor, 1964:489).

Early work in computational linguistics reinforced this conclusion, pointing out that it is not just lists of facts that are involved in disambiguation, but reasoning based on those facts. Furthermore, the kind of contextual reasoning required to disambiguate some sentences can draw on facts which require a subtle understanding of social or political structures, rather than facts about the physical world:

- (3) The police refused the students permission to demonstrate because they advocated/feared violence.

The decision as to whether ‘they’ refers to ‘the police’ or to ‘the students’ with one or another of the alternative verbs, seems to be given by our assessment of the relative plausibilities of the propositions ‘police/students advocate violence’ and ‘police/students fear violence’, and the likelihood that each of these propositions could form a reason for the police withholding permission. This decision involves judging the plausibility of a proposition inferred from things already known, but itself novel: it is very unlikely that a hearer has consciously or unconsciously worried on any previous occasion about whether police advocate or fear violence more than students do.

Current implemented approaches to the disambiguation problem rely on statistical methods, for the most part. Starting with a corpus of disambiguated sentences (a treebank), some kind of classifier is trained to distinguish ‘good’ from ‘bad’ parses of new sentences. The classifier will be trained on ‘features’ in the machine learning sense, typically some combination of head words and grammatical or structural relations. This classifier may be implicit as an integral part of the parsing algorithm (e.g., Collins 1997) or a separate component acting as a ranking mechanism on the output of an N-best parser (as in early work like Alshawi and Carter 1994, or more recent systems like Collins and Duffy 2002).

At the current state of the art, training a statistical classifier to perform disambiguation is the method of choice, and will deliver a level of performance that traditional symbolic methods, even where they are available, are unable to approach. Nevertheless, our position is that while these techniques are clearly the best current engineering solution to the problem of disambiguation, they are unsatisfactory in several scientific respects. Firstly, these solutions do not easily (or at all) generalise from one type of disambiguation problem to another. For example, in the following sentences it is intuitively clear that the same piece of general knowledge - that you can cut bread with a knife - is used to carry out a PP attachment decision, interpret a compound nominal, and assign an antecedent to a pronoun:

- (4) Tell him to cut the bread with the knife.
 (5) He picked up the bread knife.
 (6) Put the bread on the wooden board. Use the knife to slice it.

But it is not at all straightforward to reuse the type of classifier that one might build to disambiguate the PP attachment for the remaining two interpretation tasks.

Secondly, it is by no means clear that one can in practice extend statistical methods to the case of context-dependent disambiguation, or to the kind of reasoning found in cases of conversational implicature or ‘relevance’. Many disambiguation decisions require knowledge of what entities are salient, unique, etc.

Consider the task of deciding whether ‘by the factory’ should be construed as an agentive or a locative phrase in the following examples (taken from a Google search for ‘by the factory’):

- (7) Four (4) Whelen #64 strobe lights shall be located on the rear of the body, over the rear taillights. The lens colors shall be (2) red, (1) amber, and (1) blue. They shall be strategically located by the factory using good judgement.
- (8) In our offices, located by the factory, we have our export department where experienced personnel attends the requirements of our export costumers (*sic*) ...

The correct decision in the first example requires the correct pronoun resolution to have been made. Surely the task of getting enough context and disambiguated example training material pairs would be in practice impossible: the sparse data problem is bad enough for context-independent sentence disambiguation.

What is the alternative? We suggest that the traditional wisdom — that you need a lot of knowledge of the world to make disambiguation decisions — is largely correct. Furthermore, the observation that this knowledge of the world is not just a list of facts, but is organised in a deductive structure that allows you to make novel inferences is also largely correct. The question is, how do we get such theories? The early pessimism about the possibility of building large scale monolithic theories by hand is surely justified, although there have been some heroic (and expensive) attempts (Lenat 1995). Moreover, it is not clear that a single ‘theory of the world’ is what is wanted: experience in using general linguistic classification schemes, such as WordNet (Fellbaum 1998), suggests that in particular domains, word usage and relationships can be rather different from ‘general usage’, which is presumably based on an abstraction from many such domains, not necessarily chosen systematically. Rather, a more focused and less ambitious aim is to develop a theory for a particular domain: by ‘domain theory’ we mean a collection of facts and generalisations or rules which capture what commonly happens (or does not happen) in some domain of interest.

Domain specific hand-crafted theories have been developed from time to time: the earliest was perhaps embodied in the ‘preference semantics’ of Wilks (1975) (although this was in fact quite general in its intended coverage) and more recent attempts have been described by Hobbs et al. (1993). However, even constructing such smaller theories is still an extremely labour intensive business.

What we need is some way of automatically, or semi-automatically, inducing domain theories from data of some kind. But from what kind of data? Here, we can perhaps go back to the original observation that in order to make a disambiguation decision, we need a great deal of knowledge about the world. However, the observation that disambiguation decisions depend on knowledge of the world can be made to cut both ways: *just as we need a lot of knowledge of the world*

to make disambiguation decisions, so a given disambiguation decision can be interpreted as telling us a lot about the way we view the structure of the world. Since in the general case it is a much easier job to disambiguate sentences than to directly encode the theory that we are drawing on in so doing, a better strategy for trying to build a domain theory would be to try to capitalise on the information that is tacitly contained in those disambiguation decisions.

2 A partial ATIS domain theory

In Pulman (2000) it was shown how it was possible to learn a simple domain theory from a disambiguated corpus: a subset of the ATIS (air travel information service) corpus (Godfrey & Doddington 1990). Ambiguous sentences were annotated as shown to indicate the preferred reading:

```
[i,would,like,
  [the,cheapest,flight,from,washington,to,atlanta]]
```

```
[can,i,[have,a,steak_dinner],on,the,flight]
```

```
[do,they,[serve,a,meal],on,
  [the,flight,from,san_francisco,to,atlanta]]
```

```
[i,would,like,[a,flight,from,boston,to,san_francisco,
  [that,leaves,before,'8:00']] ]]
```

The ‘good’ and the ‘bad’ parses were used to produce simplified first order logical forms representing the semantic content of the various readings of the sentences. The ‘good’ readings were used as positive evidence, and the ‘bad’ readings (or more accurately, the bad parts of some of the readings) were used as negative evidence. For example, the good and bad readings of the first example above are:

Good:

```
∃A.flight(A) & from(A,washington)
  & to(A,atlanta) & cheapest(A)
  & would_like(e73,I,A)
```

Bad:

```
∃A.flight(A) & from(A,washington)
  & cheapest(A) & would_like(e75,I,A)
  & to(e75,atlanta)
```

We use an eventish semantics for verbs: ‘e73’ etc are skolem constants deriving from an event quantification, denoting events. The bad reading claims that it is the liking event that is going to Atlanta, not the flight. Note that some components of the logical form are common to both the good and bad readings: when

shared components are factored out, and we have skolemised and transformed to clausal form, the positive and negative evidence derived from this sentence would be:

Positive:

```
cheapest(sk80).
flight(sk80).
from(sk80,washington).
to(sk80,atlanta).
would_like(e73,I,sk80).
event(e73).
```

Negative:

```
not(to(e73,atlanta))
```

Note that we have added some extra background sortal information to distinguish events from other individuals. We also add some other background sortal information: that United, American etc. are airlines, that Atlanta and Washington are cities, and so on.

Next we used a particular Inductive Logic Programming algorithm, Progol (Muggleton 1994), to learn a theory of propositional relations in this domain: i.e., what kinds of entities can be in these relations, and which cannot:

```
on(+any,+any)
from(+any,+any)
to(+any,+any)
at(+any,+any)
```

The +any declaration says that we have no prior assumptions about sortal restrictions on these predicates. Among others we learn generalisations like these (all variables are implicitly universally quantified):

```
fare(A) & airline(B)    → on(A,B)
event(A) & flight(B)    → on(A,B)
meal(A) & flight(B)     → on(A,B)
flight(A) & day(B)      → on(A,B)
flight(A) & airline(B)  → on(A,B)
```

Fares and flights are on airlines; events (like serving a meal) can be on flights, meals can be on flights, and flights can be on (particular) days. However:

```
event(A) & airline(B)  → not(on(A,B))
event(A) & city(B)     → not(from(A,B))
event(A) & city(B)     → not(to(A,B))
```

— events cannot be on airlines, and events don't go to or from cities. Using a different ILP algorithm, WARMR (Dehaspe & De Raedt 1997), for discovering frequent patterns (in data mining terms, association rules) and the system

Aleph (Srinivasan 2003) for deriving constraints from these rules, we obtained the following facts:

Flights are direct or return, but not both:

```

direct(A)                → flight(A)
return(A)                → flight(A)
direct(A) & return(A)   → false

```

These generalisations are true with respect to the given corpus, but may not be true more generally. However, these constraints between prepositions are presumably of general validity:

```

from(A,B) & to(A,B)      → false
at(A,B) & after(A,B)    → false
at(A,B) & before(A,B)   → false
after(A,B) & before(A,B) → false
etc.

```

Having learned this domain theory we were able to show that it could be used successfully in disambiguating a small held-out section of the corpus, by checking for consistency between logical forms and domain theories. There are many variations to be explored here, but in this particularly simple setting the negative domain theory, i.e., the generalisations or constraints about what does NOT happen are the easiest to employ. A ‘good’ reading of an ambiguous sentence will be consistent with a negative constraint, whereas a ‘bad’ reading will lead to a contradiction. This can be implemented using a model builder for first order logic, in our case, a version of Satchmo (Manthey & Bry 1988). We assume as axioms the negative domain theory. We parse a sentence, and turn each parse tree into a first order logical form. We then add each logical form in turn to the axioms, and see whether we can build a model, i.e., whether the conjunction of the negative domain theory and the logical form is satisfiable or consistent. If we can build a model, the logical form is a plausible one; if we cannot, it is an implausible one. To illustrate from the example of a bad logical form given earlier:

```

∃A.flight(A) & from(A,washington)
  & cheapest(A) & would_like(e75,I,A)
  & to(e75,atlanta)
  ... & event(e75) & city(atlanta)

```

will contradict:

```

∀A,B.event(A) & city(B) → not(to(A,B))

```

and so no model can be constructed.

While the numbers of sentences involved in this experiment are too small for the results to be statistically meaningful, the experiment proved that the method works in principle, although of course in reality the notion of logical consistency

is too strong a test in many cases. Note also that the results of the theory induction process are perfectly comprehensible - we get a theory with some logical structure, rather than a black box probability distribution. The theory we have got can be edited or augmented by hand, and could be used for other types of disambiguation task, such as word disambiguation and pronoun resolution. It could also be used for other types of NLP task altogether: logical domain theories have recently been used to improve performance in natural language question answering tasks (Moldovan et al 2003) and it has been argued that automatic creation of templates for Information Extraction could benefit from such domain theories (Collier 1998). The output of our theory induction process thus meets the criteria of transparency and reusability discussed earlier.

3 Scaling up

Of course, the ATIS corpus is relatively simple and homogeneous. The question is whether this technique will scale up to larger and noisier corpora. However, this is rather difficult to do in practice since the method requires the corpus in question to be analysable to the extent that complete logical forms can be recovered for both good and bad readings of sentences. We know of no such large corpora: however, the Penn Tree Bank (Marcus et al. 1994) is a good starting point, since the syntactic annotations for sentences given there are intended to be complete enough for semantic interpretation, in principle, at least.

In practice, as we reported in Liakata and Pulman (2002), it is by no means easy to do this. We were able to recover partial logical forms from a large proportion of the treebank, but these are not complete or accurate enough to simply replicate the ATIS experiment. In order to approach this we selected about 40 texts containing the verb ‘resign’, all reporting, among other things, ‘company succession’ events, a scenario familiar from the Message Understanding Conference (MUC) task (Grishman & Sundheim 1996). The texts amounted to almost 4000 words in all. Then we corrected and completed the automatically produced logical forms by hand to get a fairly full representation of the meanings of these texts (as far as is possible in first order logic). We also resolved by hand some of the simpler forms of anaphoric reference to individuals to simulate a fuller discourse processing of the texts.

To give an example, a sequence of sentences like:

J.P. Bolduc, vice chairman of W.R. Grace & Co. ... was elected a director. He succeeds Terrence D. Daniels,... who resigned.

was represented by the following sequence of literals:

```
verb(e1,elect).
funct_of('J.P._Bolduc',x1).
...
```

```

subj(e1, unspecified) .
obj(e1, x1) .
description(e1, x1, director, del) .

```

```

verb(e5, succeed) .
subj(e5, x1) .
funct_of('Terrence_D._Daniels', x6) .
obj(e5, x6) .
verb(e4, resign) .
subj(e4, x6) .

```

The representation is a little opaque, for various implementation reasons. It can be paraphrased as follows: there is an event, *e1*, of electing, the subject of which is unspecified, and the object of which is *x1*. *x1* is characterised as 'J P Bolduc', and *e1* assigns the description *del* of 'director' to *x1*. There is an event *e5* of succeeding, and *x1* is the subject of that event. The object of *e5* is *x6*, which is characterised as Terrence D Daniels. There is an event *e4* of resigning and the subject of that event is *x6*.

The reason for all this logical circumlocution is that we are trying to learn a theory of the 'verb' predicate, in particular we are interested in relations between the arguments of different verbs, since these may well be indicative of causal or other regularities that should be captured in the theory of the company succession domain. If the individual verbs were represented as predicates rather than arguments of a 'verb' predicate we would not be able to generalise over them: we are restricted to first order logic, and this would require higher order variables.

We also need to add some background knowledge. We assume a fairly simple flat ontology so as to be able to reuse existing resources. Some entities were assigned to classes automatically (see below for details of the clustering techniques used), others had to be done by hand. The set of categories used were:

company, financial instrument, financial transaction, location, money, number, person, company position, product, time, and unit (of organisation).

As before, the representation has these categories as an argument of a 'class' predicate to enable generalisation:

```

class(person, x1) .
class(company, x3) .
etc.

```

Ideally, to narrow down the hypothesis space for ILP, we need some negative evidence. In the ATIS case, we were able to do this because our parser was able to find all parses for the sentences in question, good and bad. In the case of the Penn Tree Bank, only the good parse is represented. There are several possible

ways of obtaining negative data, of course: one could use a parser trained on the Tree Bank to reparse sentences and recover all the parses. However, there still remains the problem of recovering logical forms from ‘bad’ parses. An alternative would be to use a kind of ‘closed world’ assumption: take the set of predicates and arguments in the good logical forms, and assume that any combination not observed is actually impossible. One could generate artificial negative evidence this way.

Alternatively, one can try learning from positive only data. Prolog and Aleph are able to learn from positive only data, with the appropriate settings. Also, so-called ‘descriptive’ ILP systems like WARMR do not always need negative data: they are in effect data mining engines for first order logic, learning generalisations and correlations in some set of data.

4 Domain theory for company succession events

We found that the most successful method, given the absence of negative data, was to use WARMR to learn association rules from the positive data. As with all types of association rule learning, WARMR produces a huge number of rules, of varying degrees of coverage. We spent some time writing filters to narrow down the output to something useful. Such filters consist of constraints ruling out patterns that are definitely not useful, for example patterns containing a verb but no arguments or attributes. An example of such a restriction is provided below:

```
pattern_constraint(Patt):-
    member(verb(_,E,_A,_,_),Patt),
    (member(attr(_,E,Attr),Patt)
     ->
     \+constraint_on_attr(Patt,Attr)).
```

This says that a rule isn’t useful unless it contains a verb and one of its attributes that satisfies a certain constraint. A constraint is of the following form:

```
constraint_on_attr(Patt, Attr) :-
    member(class(_,Attr), Patt).
```

The above states that there should be a classification of the attribute Attr present in the rule. A useful pattern Patt will satisfy such constraints and thus make the ‘pattern_constraint’ predicate fail.

Some of the filtered output, represented in a more readable form compatible with the examples above are:

Companies report financial transactions:

```
subj(B,C) & obj(B,D) & class(fin_tran,D) & class(company,C)
→ verb(B,report)
```

Companies acquire companies:

subj(B,C) & obj(B,D) & class(company,D) & class(company,C)
 → verb(B,acquire)

Companies are based in locations:

obj(A,C) & class(company,C) & in(A,D) & class(location,D)
 → verb(A,base)

If a person is elected, another person resigns:

verb(H,elect) & obj(H,I) & class(person,I) & subj(C,L)
 & class(person,L)
 → verb(C,resign)

If person C succeeds person E, then someone has elected person C:

obj(A,C) & class(person,C) & verb(D,succeed) &
 subj(D,C) & obj(D,E) & class(person,E)
 → verb(A,elect)

If someone elects person C, and person D resigns, then C succeeds D:

subj(G,C) & verb(A,elect) & obj(A,C) & class(person,C)
 & verb(E,resign) & subj(E,D) & class(person,D)
 → verb(G,succeed)

While there are many other rules learned that are less informative than this, the samples given here are true generalisations about the type of events described in these texts: unremarkable, perhaps, but characteristic of the domain. It is noteworthy that some of them at least are very reminiscent of the kind of templates constructed for Information Extraction in this domain, suggesting a possible further use for the methods of theory induction described here.

In the ATIS domain, we were able to evaluate the usefulness and accuracy of the rules induced by applying them in a syntactic disambiguation task. However, we have not been able to replicate that evaluation here, for a variety of reasons of which the most important is the relative sparseness of the rules derived: much more information would be needed in most cases for successful disambiguation.

5 Next steps

In our current work we are trying to scale up again. However, to do this successfully we need better ways of obtaining negative evidence, and of formalising background knowledge. For the former, we intend to experiment with a wide coverage parser which is capable of delivering all analyses for sentences rather than simply the best one. For the latter we have used clustering techniques to try to group logical predicates into classes which will allow the various ILP algorithms to generalise over samples of evidence involving these predicates. In the

ATIS experiment we carried this step out by hand, whereas for the company succession events we used a mixture of automatically derived and hand made classes. Obviously, a larger data set would require full automation of the process.

In our current experiments we have tried clustering the entire set of logical form verb predicates derived from the PTB. More specifically, we clustered the verb argument slots of each predicate (e.g., *resign_arg1* is the subject of the verb *resign*). Note that the logical form predicates are derived automatically from the stem form of the word involved, and so no word sense disambiguation is being carried out, leading to a major source of noise in the data. We constructed a large three-dimensional matrix:

predicates X arguments X frequency

An artificial example of such a matrix is:

	cat	dog	father	computer
snore_arg1	3	2	10	0
hit_arg1	2	4	1	2
hit_arg2	6	7	2	20

This says that *cat* occurs as first argument of ‘snore’ three times, *dog* twice, *father* ten times and *computer* not at all. We used Autoclass (Cheeseman 1995) to form clusters of verb argument slots such that the ones grouped together are those verb slots filled by the same words. Classification of verbs and words that feature as their arguments was then easily derivable.

We ran Autoclass assuming that word frequencies follow a normal distribution, with dependencies between members of the same class. Autoclass found between 32 to 55 clusters, depending on whether the table consisted of absolute frequencies, logarithms of frequencies or whether there had been a pre-processing stage for grouping together person names, locations, numerical expressions and company names. This pre-processing stage involved checking against various gazetteer lists. We decided to stick with 32 clusters, since the classification into more groups did not make the intuitive basis for the clusters any more obvious. These were the clusters obtained for the absolute frequencies of the pre-processed data.

Many of the resulting classes are difficult to interpret, although about half have some clear intuitive basis, once some outliers are ignored:

Class 9: *plunge, soar, climb, decline, jump, slide, plummet, pick up, slip*

These verbs seem to have a common theme of a sudden movement, and when we look at the most frequently occurring arguments of some of them it becomes clear that they describe sudden movements of what might be called the class of ‘financial indicators’:

PLUNGE_arg1: subject argument:

dow_jones_industrial_average, income, stock,
 person, profit, market, earnings,
 average, net, share, price,

SOAR_arg1: subject argument:

rate, price, profit, location, cost, dollar,
 volume, income, asset, interest, circulation,
 stock, revenue, jaguar_shares, earnings, person

PLUMMET_arg1:

price, market, stock, ual_stock, earnings, profit,
 company, indicator, yield

JUMP_arg1:

income, profit, price, person, company,
 revenue, location, earnings, index,
 dow_jones_industrial_average,
 contract, yield, cost, gold, people

This class of verbs seems to involve a causal mechanism for a similar type of financial change:

Class 10: bolster, increase, reduce, boost, occur,
 trigger

We find a class of aspectual verbs:

Class 16: close, begin, start, complete

and we also find the class of the ‘company succession verbs’

Class 18: appoint, resign, succeed

Some other classes also have a clear semantic relationship to each other:

Class 23: know, believe, think

Class 24: buy, pay, purchase, own

As well as Autoclass, which is essentially classification by Expectation Maximisation, we also tried another method of clustering based on Independent Component Analysis (Roberts & Everson 2001). This latter approach involved obtaining the matrix of frequencies and minimizing its dimensions. Four classes resulted from this method but while easily interpretable, they were too general to be of use for our purposes. Nevertheless, the theoretical foundation of this clustering technique may well make it more suitable for future work, since it makes more realistic assumptions about the distributions our samples are from.

In principle, the way we would proceed having satisfactorily grouped predicates into clusters (perhaps hierarchical, although so far we have not attempted this) would be then to assign informative labels to clusters, and represent the relations between clusters and their members as background knowledge:

plunge(X)	→	move_suddenly(X)
soar(X)	→	move_suddenly(X)
plummet(X)	→	move_suddenly(X)
...		
financial_indicator(X)	→	plummet(X)
financial_indicator(X)	→	soar(X)
...		
dow_jones_industrial_avge(X)	→	financial_indicator(X)
earnings(X)	→	financial_indicator(X)
market(X)	→	financial_indicator(X)
...		

This would enable the learning algorithm to generalise correctly over related examples.

As for evaluation, our original hope was to be able to use the derived theory both for syntactic disambiguation and for other tasks like reference resolution. In particular, we wanted to be able to deal with the cases that currently successful pronoun resolution algorithms (e.g., Hobbs 1978, Boguraev and Kennedy 1996) get wrong. These are cases where the structural configurations preferred by these algorithms yield interpretations that to us are clearly implausible. Lappin (2004) has argued that however successful a structural or statistically based algorithm is, there will always be a residue of cases like these where detailed world knowledge and reasoning is required to get the right result.

The following examples from the Penn Tree Bank illustrate the phenomenon in question:

(9) The government_i counts money as it_i is spent.

Parallelism and grammatical relation salience will combine to strongly prefer the reading indicated, which is of course wrong. In order to override this reading and select the correct one you need to know that while it is possible for money to be spent, it is not possible for a government to be spent (at least, not in the same sense).

An example close to the kind of clusters we have been able to induce is this one, where again the structural preference would be as indicated:

(10) Investors_i usually notice markets because they_i rise or fall rapidly.

This interpretation requires it to be more plausible that investors rise or fall than that markets do. We would hope that the correct preference would be captured by generalisations like those sketched above.

It is arguable that a statistical system for pronoun resolution, which infers sortal restrictions on arguments from collocations, would probably be able to deal with the above example. However, this is certainly not the case for examples like the following:

- (11) Leaders_{*i*} of several Middle Eastern terrorist organisations met in Teheran to plan terrorist acts. Among them_{*i*} was the PFL of Palestine...

Again, if we ignore the content of the sentence, the most likely antecedent for ‘them’ is the one indicated. In order to see that this is incorrect, we need to know an axiom associated with ‘among’: roughly, that if X is among Ys, and all Ys are P, then X is P. However, this axiom is somewhat more abstract than those we are learning (although domain specific versions are attainable) and it remains to be seen whether it is possible to learn a theory with enough width and depth of coverage to be properly evaluated on something like a pronoun disambiguation task.

A further possibility is to use the domain theory learning method described here to learn *new* knowledge. The domain theories we have described so far are trying to encapsulate the ‘common-sense’ understanding that is required to interpret and disambiguate texts - this is routine, everyday knowledge that members of the relevant community can be expected to share. But if we instead had parsed texts from some technical domain - for example, the protein interaction research abstracts used in the information extraction experiment described in Thomas et al. 2000 - it is conceivable that an induced theory might well lead one to be able to deduce some completely new and useful information about that domain. By putting together information from different texts in the form of facts and inference rules, it may be possible to derive new conclusions that would not otherwise have been apparent.

Acknowledgements. We have had a great deal of help from members of the ILP community. We would particularly like to thank Ashwin Srinivasan (IBM, New Delhi), Steve Moyle (Oxford) and James Cussens (York) for their help with Aleph and Jan Struyf, Hendrik Blockeel and Jan Ramon (K.U. Leuven), for their generous help with WARMR. We also thank Steve Roberts (Oxford) for advice on clustering and for carrying out the ICA analysis for us.

REFERENCES

- Alshawi, H. & D. Carter. 1994. “Training and Scaling Preference Functions for Disambiguation”. *Computational Linguistics* 20:4.635-648.
- Boguraev, Bran & C. Kennedy. 1996. “Anaphora for Everyone: Pronominal Anaphora Resolution without a Parser”. *Proceedings of the 17th International Conference on Computational Linguistics (COLING 1996)*, 113-118. Copenhagen, Denmark.
- Cheeseman, P., J. Kelly, M. Self, J. Stutz, W. Taylor, & D. Freeman. 1998. “Auto-Class: A Bayesian Classification System”. *Proceedings of the 5th International Conference on Machine Learning, Ann Arbor, Michigan, June 12-14*, 54-64. San Francisco, Calif.: Morgan Kaufmann.
- Collier, Robin. 1996. “Automatic Template Creation for Information Extraction, An

- Overview". Technical Report CS-96-07. Sheffield, U.K.: University of Sheffield, Department of Computer Science.
- Collier, Robin. 1998. "Automatic Template Creation for Information Extraction". Ph.D. dissertation, University of Sheffield. Sheffield, U.K.
- Collins, Michael & Nigel Duffy. 2002. "New Ranking Algorithms for Parsing and Tagging: Kernels over Discrete Structures, and the Voted Perceptron". *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL'02)*, 263-270. Philadelphia, Pennsylvania, U.S.A.
- Collins, Michael. 1997. "Three Generative, Lexicalised Models for Statistical Parsing". *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (jointly with the 8th Conference of the European Association for Computational Linguistics)*, 16-23. Madrid, Spain.
- Dehaspe, L. & L. De Raedt. 1997. "Mining Association Rules in Multiple Relations". *Proceedings of the 7th International Workshop on Inductive Logic Programming (= Lecture Notes in Artificial Intelligence, 1297)*, 125-132. Berlin: Springer-Verlag.
- Dehaspe, L. 1998. "Frequent Pattern Discovery in First-Order Logic". Ph.D. dissertation, Katholieke Universiteit Leuven. Leuven, Belgium.
- Fellbaum, Christiane, ed. 1998. *WordNet: An Electronic Lexical Database*. Cambridge, Massachusetts: MIT Press.
- Hemphill, C., John Godfrey & George Doddington. 1990. "The ATIS Spoken Language Systems Pilot Corpus". *Speech and Natural Language Workshop, Hidden Valley, Pennsylvania*, 96-10. San Francisco, Calif.: Morgan Kaufmann.
- Grishman, Ralph & Beth Sundheim. 1996. "Message Understanding Conference-6: A Brief History". *Proceedings of the 16th International Conference on Computational Linguistics (COLING'96)*, 466-471. Copenhagen, Denmark. — www.cs.nyu.edu/cs/projects/proteus/muc/muc6-history-coling.ps [Source checked in May 2004]
- Hobbs, Jerry R. 1978. "Resolving Pronoun Preferences". *Lingua* 44.311-338.
- Hobbs, Jerry R., Mark E. Stickel, Douglas Appelt & Paul Martin. 1993. "Interpretation as Abduction". *Artificial Intelligence* 63:1/2.69-142.
- Katz, Jerrold J. & Jerry A. Fodor. 1964. "The Structure of a Semantic Theory." *The Structure of Language: Readings in the Philosophy of Language* ed. by J.J. Katz & J.A. Fodor, 479-518. Englewood Cliffs, N.J.: Prentice Hall.
- Lappin, Shalom & Herbert J. Leass. 1993. "An algorithm for Pronominal Anaphora Resolution". *Computational Linguistics* 20:4.535-561.
- Lappin, Shalom. 2004. "A Sequenced Model of Anaphora and Ellipsis Resolution". *Anaphora Processing: Linguistic, Cognitive, and Computational Modelling* ed. by A. Branco, A. McEnery & R. Mitkov. Amsterdam: John Benjamins.
- Lenat, Douglas B. 1995. "CYC: A Large-scale Investment in Knowledge Infrastructure". *Communications of the Association for Computing Machinery (CACM)* 38:11.33-38.

- Liakata, Maria & Stephen G. Pulman. 2002. "From Trees To Predicate-Argument Structures". *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2002)*, 563-569. Taipei, Taiwan.
- Manthey, Rainer & Francois Bry. 1988. "SATCHMO: A Theorem Prover Implemented in Prolog". *Proceedings of the 9th International Conference on Automated Deduction, Argonne, Illinois, U.S.A.* ed. by Ewin L. Lusk & Ross A. Overbeek (= *Lecture Notes In Computer Science* LNCS, 310), 415-434. Berlin: Springer.
- Marcus, Mitchell, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz & Britta Schasberger. 1994. "The Penn Treebank: Annotating Predicate Argument Structure". *ARPA Human Language Technology Workshop*. Princeton, N.J.
- Moldovan, Dan, C. Clark, S. Harabagiu & S. Maiorano. 2003. "COGEX: A Logic Prover for Question Answering". *Proceedings of Human Language Technology-North American Association for Computational Linguistics (HLT-NAACL'03)*, 87-93. Edmonton, Canada.
- Muggleton, S. 1995. "Inverse Entailment and Progol". *New Generation Computing* vol.13, 245-286.
- Pulman, Stephen G. 2000. "Statistical and Logical Reasoning in Disambiguation". *Philosophical Transactions of the Royal Society of London, Series A*, vol.358, 1267-1280.
- Roberts, S. & R. Everson. 2001. *Independent Component Analysis: Principles and Practice*. Cambridge: Cambridge University Press.
- Srinivasan, Ashwin. 2003. "The Aleph Manual". Oxford: Oxford University, Machine Learning Laboratory. — www.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/ [Source checked in May 2004]
- Thomas, J., D. Milward, C. Ouzounis, S.G. Pulman, & M. Carroll. 2000. "Automatic Extraction of Protein Interactions from Scientific Abstracts". *Proceedings of Pacific Symposium on Biocomputing 2000*, 541-552. Hawaii.
- Wilks, Yorick. 1975. "Preference Semantics". *Formal Semantics of Natural Language*, ed. by E.L. Keenan, 329-348. Cambridge: Cambridge University Press.