Statistical and logical reasoning in disambiguation

Stephen G. Pulman

Phil. Trans. R. Soc. Lond. A 2000 **358**, 1267-1280
doi: 10.1098/rsta.2000.0585

Statistical and logical reasoning in disambiguation

By Stephen G. Pulman

University of Cambridge Computer Laboratory, New Museums Site, Cambridge CB2 3QG, UK and
SRI International Cambridge Computer Science Research Centre, Suite 23, Millers Yard, Mill Lane, Cambridge CB2 1RQ, UK

While recent statistical methods for disambiguation in natural language processing have been very successful, earlier arguments still hold for the position that fully successful disambiguation requires reasoning to new conclusions from old facts. We explore ways of complementing statistical approaches with the use of 'domain theories': collections of facts and axioms that characterize the typical structure of some task domain. In particular, we hypothesize that disambiguation decisions can supply tacit information about such theories, and that the theories can, in part, be automatically induced from such data. We describe a pilot experiment in which a partial domain theory for the domain of air-travel information was induced from a corpus of disambiguated example sentences, the resulting theory then being used successfully in disambiguating other sentences from the same domain.

Keywords: natural language semantics; ambiguity; domain theories; inductive logic programming

1. Introduction

Ambiguity has played an important role in linguistics. The existence of some types of ambiguity is in fact the clearest evidence that sentences have abstract structure over and above a linear sequence of words:

1 a Flying planes can be dangerous
b He saw the man with the telescope

However, the process of disambiguation itself has received relatively little attention within linguistics, probably because of influential early work in linguistic semantics that stressed the variety of non-linguistic knowledge needed for disambiguation, and, by implication, the difficulty of eliciting and encoding such knowledge so as to be able to use it for this purpose:

> 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, p. 489).

In fact, it is not just having the relevant information available that is needed: a wealth of examples seems to indicate that we have to go on to draw arbitrarily complex inferences from that information to make disambiguation decisions.

2 Little John was looking for his toy box.
The box was in the pen.

Bar-Hillel's famous example (Bar-Hillel 1960) demonstrates that in order to select the more plausible interpretation of 'pen' as 'play pen' rather than 'fountain pen' we have to draw inferences about the context of the sentence itself. This reasoning presumably proceeds along roughly the following lines. We locate the referent for 'the box' as 'the toy box', and we find (let us assume) just the two candidate meanings for 'pen'. We know something about the relative sizes of these objects and find that we can construct a picture consistent with these relative sizes for the interpretation 'box in play pen' but not for 'box in fountain pen', and hence we choose the former.

Other examples that have become equally well known suggest that the reasoning involved can draw on facts that 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. However, both here and in the earlier example, the decision involves judging the plausibility of a proposition inferred from things already known, but which is itself novel: it is very unlikely that a hearer has consciously or unconsciously worried on any previous occasion about whether toy boxes can fit into fountain pens, or whether police fear violence more than students do.

Observations like these have led to a widespread perception within the linguistics community, and to some extent within the computational linguistics community, that successful disambiguation in automatic natural language processing systems is something that awaits the completion of the programme of traditional artificial intelligence: the ability to represent and reason with the knowledge of the world available to the average native speaker of a language. This is not a programme that is currently regarded as feasible.

More recently, people have turned to statistical methods for resolving some types of ambiguity; methods that appear to be relatively successful without involving the kind of complex knowledge representation and reasoning schemes implied by Katz & Fodor's (1964) claim. Some of these methods (e.g. focus-based schemes for pronoun resolution) rely on the frequency of particular configurations of structural properties, and, thus, do not even attempt to use anything like a domain theory in the sense that we are using it. Others, however, try to include some notion of content via the inclusion of particular words or word-senses; for example, the 'preference-based' disambiguation schemes in the Core Language Engine (Alshawi & Carter 1994), or the lexical head and dependency based schemes of Collins (1998).

Some of this latter family of statistical techniques can be seen as ways of trying to build a domain theory. The result of statistical training is a crude kind of theory of the structure of the domain that allows us to make assessments of the relative probabilities of competing propositions. For example, if we were able to train on a large relevant corpus of English, annotated in the appropriate way, we might hope that our preference measures would tell us that given sentences like

4 a I blew up the safe with the dynamite
b I saw the man with the bicycle

the verb phrase attachment is preferred for (a) on the grounds that the simple proposition 'blow up with dynamite' is more likely than 'safe with dynamite', whereas the noun phrase attachment is preferred for (b), because 'man with bicycle' is more likely than 'see with bicycle'.

The problem with such schemes is that they are very dependent on the testing data being sufficiently like the training data in all the ways relevant for the disambiguation decision. A scheme for lexical disambiguation trained on a large corpus of sentences of English, for example, would be very unlikely to make the right decision for Bar-Hillel's (1960) example, or for the demonstrators example, because the information needed to trigger the decision is simply not present in the form of the sentence; that was Bar-Hillel's (1960) point. It might be argued that this problem could be solved by including more conditioning factors in the statistical model; for example, taking into account some of the properties of surrounding sentences, which act as a context. The problem with this suggestion is that the corpus used for training now has to be very much larger than before if a sufficient number of relevant observations are to be found in it. It might also be argued that examples like Bar-Hillel's (1960) are rare enough that for practical purposes failure to resolve them does not matter; this is a defensible engineering principle, but it still leaves us without a scientific account of what this kind of disambiguation involves.

What is really needed is to find some way of combining some properties of statistical approaches with theories that have sufficient deductive structure to allow them to be used in reasoning to the appropriate conclusions, even when these conclusions are not directly reflected in any of the previously observed data. Some kinds of automatic clustering and back-off schemes can be seen as a way of trying to add some of this deductive structure to n-gram or other methods. But the types of inferences that these extra mechanisms sanction are typically rather simple: essentially 'isa' hierarchies and other types of taxonomic reasoning. It is not likely that they will extend to, say, reasoning involving logical connectives, complex predicates, or quantifiers.

2. Building domain 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. 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. Domain-specific hand-crafted theories have been developed from time to time: the earliest was perhaps embodied in the 'preference semantics' of Wilks (1975), 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 capitalize on the information that is tacitly contained in those disambiguation decisions.

In order to test this hypothesis and develop a method for learning such domain theories, a small pilot experiment was carried out aimed at constructing a logically structured domain theory automatically from a disambiguated corpus. In overview, the method was to use the disambiguated corpus to produce a set of logical expressions representing linguistically possible interpretations of the sentences. These interpretations can be assumed to describe both what is and is not possible in the domain. Roughly speaking, the logical form representing the correct reading of a sentence can be assumed to describe something possible in the domain ('positive data'), whereas logical forms corresponding to impossible readings describe something that cannot happen in the domain ('negative data'). This is an oversimplification in several ways, of course. Firstly, the correct and incorrect logical forms overlap in some of their components, and so the overlap must be factored out of the negative data. Secondly, it may not be the case that the incorrect reading is actually impossible rather than just unlikely in the context. The first simplification must be remedied, but we will have to ignore the second, at least for the time being.

Having got the negative and positive data, we can then apply a machine learning method to induce a theory that adequately predicts the positive and negative data, ideally from a small and compact set of axioms. The method used was the 'inductive logic programming' (ILP) paradigm (Muggleton & De Raedt 1994; Muggleton 1995*a*). Given a background theory B, and some evidence E, the task of ILP is to find a hypothesis H such that $B \,\&\, H \models E$. This latter expression can be rearranged as $B \,\&\, \overline{E} \models \overline{H}$. For certain types of first-order theory there will be some set of ground literals $\overline{\bot}$ true in every model of $B \,\&\, \overline{E}$. Since $\overline{H}$ must also be true in every such model, it follows that $B \,\&\, \overline{E} \models \overline{\bot} \models \overline{H}$. Then, for any such H, it must be the case that $H \models \bot$. The set of candidates for H can be enumerated by searching through the lattice of clauses that subsume $\bot$, or variants of $\bot$. The CProgol system (Muggleton 1995*a*) searches this lattice to produce the candidate for H which explains E in the most economical way, that is, resulting in the smallest number of clauses in the final theory.

The ILP paradigm was chosen for several reasons. Firstly, it is known to scale-up effectively to realistically sized problems. Secondly, the particular ILP algorithm used is tolerant to noise or inconsistency in the data. Thirdly, the framework offers the promise of being able to integrate logical theories with statistical modelling by learning rules with probabilities associated with them (Muggleton 1995*b*).

In our case, the resulting set of clauses constitute the domain theory we are looking for, and can be used, in principle, to carry out the kind of disambiguation task we are concerned with. Disambiguation itself can be carried out in several ways: the most straightforward, but least efficient, is simply to find each interpretation in turn and discard those that are inconsistent with the axioms of the domain theory. The remaining interpretations are guaranteed to be possible ones, but some further reasoning may still be required to select the preferred one (e.g. interpretations that add no new information will be consistent but uninformative).

Step 1: data collection

To begin with, a set of 500 sentences from the Air Travel Information Service (ATIS) corpus were collected by searching for those containing the words ‘flight’, ‘fare’ and ‘meal/dinner/breakfast’. The ATIS corpus was chosen because although it is real data, the domain is very simple and constrained, and constrained even further here by limiting ourselves to just a few types of sentence. However, it is worth pointing out that even though the domain is simple, it is representative of many types of task domain that are suitable for practical commercial applications.

A simple unification grammar and parser using a formalism described in Pulman (1996, 2000) was used to parse the sentences into a kind of ‘quasi-logical form’ (QLF). After adding the 30 or 40 most-frequent missing vocabulary items to the system, and, in some cases, simplifying the sentences in ways that were irrelevant to the experiment (e.g. normalizing airline and city names, dates, times, etc.), about 200 sentences were obtained that parsed fully. One hundred and twenty of the sentences were unambiguous, and although such sentences contain useful information, they were ignored for the experiment. The remaining 80 sentences were then partly disambiguated by adding brackets to the sentence:

```
[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']]]
```

Step 2: creating a ‘QLF bank’

The unbracketed sentences were parsed and the bracketing used to divide the resulting QLFs automatically into good and bad. A good QLF is one deriving from a parse that is consistent with the constituent structure indicated by the bracketing. An example of a good QLF for the first sentence above, simplified to abstract away from the particular notation used, is

$$\lambda\text{A.would(like(A,i,the(}\lambda\text{B.flight(B)}\ \&\ \text{from(B,washington)}\ \&\ \text{to(B,atlanta)}\ \&\ \text{cheapest(B))))} \quad (5)$$

An example of a bad QLF is

6 λA.would(like(A,i,the(λB.flight(B) & from(B,washington)
& cheapest(B)) & to(A,atlanta)))

(Note that QLFs are treated as predicates of events and will be subject to the relevant contextual resolutions.) In the first QLF, it is the flight, B, that is to Atlanta, correctly corresponding to the reading on which the prepositional phrases are attached to the object noun phrase. In the bad QLF, it is the event argument of the verb, A, that is to Atlanta, corresponding to the verb phrase attachment of that prepositional phrase. Other QLFs for this sentence are essentially permutations of these solutions.

Many sentences have more than one QLF that is consistent with the bracketing assigned, and, thus, get multiple QLFs characterized as good. This sentence, for example, when the prepositional phrases modify the noun phrase only, has the parse given where 'cheapest' scopes over the nominal phrase 'flight' and its modifiers 'flight from...', and also one where 'cheapest flight' is the nominal modified. Given the oversimplified semantics we have assigned to 'cheapest', the result is to get logically equivalent interpretations differing only in the orderings of the conjuncts inside the meaning of the definite noun phrase.

Step 3: turning to normal form

The ILP system used, CProgol (Muggleton 1995*a*), requires the evidence to be in the form of first-order clauses. Since the QLFs produced by the parser are intrinsically higher-order logical expressions, they have to be transformed into first-order near-equivalents. This is achieved in the following way. Firstly, the QLFs are resolved, in a null context, using the system described in Pulman (2000). Resolution consists of locating the interpretation of pronouns and definites, filling in any ellipses, and so on. In the ATIS corpus there are actually very few contextually dependent constructs like this, since the corpus consists of single sentences rather than dialogues, and so resolution for our sentences is fairly simple. In our case it essentially amounts to predicating the event predicate of a new event discourse referent, translating determiners to quantifiers with as near as possible to their surface scope ordering, and resolving all definites as existentials. We then transform these to approximately equivalent first-order expressions by translating generalized quantifiers to first-order versions, where possible, and stripping out any higher-order operators (like 'would') that correspond to tense and aspect information:

7 ∃A.flight(A) & from(A,washington) & to(A,atlanta)
& cheapest(A) & like(e73,i,A)

8 ∃A.flight(A) & from(A,washington) & cheapest(A)
& like(e75,i,A) & to(e75,atlanta)

Next we need to transform these first-order expressions into the clausal forms required by CProgol. This can be done by the usual procedures of skolemization and normalizing to conjunctive normal form. Each of the resulting sets of disjunctions of literals represents a clause. We also flatten the clauses by translating any functions into extra predicates in a way illustrated by an artificial example:

$$a(b(c), d(e)) \Rightarrow a(1,2) \,\&\, b'(1,c) \,\&\, d'(2,e).$$

In clausal form, our good and bad QLFs look like:

9 [[flight(sk80)],[from(sk80,washington)],[to(sk80,atlanta)],
[cheapest(sk80)],[like(e73,i,sk80)]]

10 [[flight(sk82)],[from(sk82,washington)],[cheapest(sk82)],
[like(e75,i,sk82)],[to(e75,atlanta)]]

Step 4: separating negative and positive evidence

We now have to factor out the good and bad components of a bad QLF. As can be seen from the previous example, the good and bad QLFs agree in some of their components, reflecting the shared elements of the analyses of the original sentence, but differ in the component that corresponds to the differing prepositional phrase attachment. However, in going to a normal form, the process of skolemization has made it a non-trivial exercise to automatically detect this overlap. In other cases, the normal-form process will produce still more radical differences between similar QLFs: for example, some different prepositional phrase attachments will produce different scoping behaviour, leading to one literal being positive in one QLF but negative in another. In fact, in going to CProgol form we ignore negations, whether they arise from the original sentence or from the normal-form process. This makes the next step easier, but also has an intuitive motivation: a negative literal is denying that some proposition is true in the domain. Under normal circumstances, this will be a proposition that nevertheless makes sense in the domain in question. There are some circumstances in which negatives are about the structure of the domain—for example, 'My shoes don't sing'—but more usually they are saying that something that is possible happens not to be the case: 'My shoes aren't black'. Hence, ignoring the negation sign, although producing a clause that does not mean the same as the original, is still producing valid data as input to the process of theory induction.

To find the overlap we search for a set of substitutions of arguments of the predicates in the two clause sets which

(a) preserves sort information as far as possible (i.e. events map to events, flights to flights);

(b) preserves the connections between different clauses (i.e. literals with the same arguments in one clause set map to literals with the same arguments in the other clause set); and

(c) minimizes the difference between the two clause sets.

When we have the best such set of substitutions, which will be a set of pairs of arguments from the two clause sets, any individual pairs in the set violating one or more of these conditions flag non-overlapping components.

In the current example, the difference is located in the two literals 'to(sk80,atlanta)' and 'to(e75,atlanta)'. We can now apply the structure-preserving components of the substitution to make the two clause sets explicitly overlap, and represent the good QLF as positive data and the non-overlapping component of the bad QLF as negative data. The data input to CProgol is

```
% positive:
   cheapest(sk80).            flight(sk80).
   from(sk80,washington).     to(sk80,atlanta).
   like(e73,i,sk80).          event(e73).
% negative:
   not(to(e73,atlanta)).
```

Notice that we add an explicit assertion that `e73` is an event, since this necessary type information is not otherwise explicitly represented.

The final dataset derived from our 80-sentence disambiguated corpus contains about 700 such clauses once duplicates are eliminated.

Step 5: inducing a domain theory

To use the ILP system CProgol to induce a relevant theory, we need to supply some background information. In our case this consists of type information about the various constants to be found in our evidence:

```
city(atlanta).          airline(united).
date('16/8').           day(tuesday).
time('8:00').           breakfast(X) --> meal(X).
etc.
```

We also need to provide some information about what predicates we are trying to learn the theory of. In this simple experiment, we try to learn the theory describing the possible uses of the four most-frequently occurring prepositions in the training corpus, since prepositional phrase attachment ambiguities are our focus:

```
:- modeh(1,on(+any,+any))?    :- modeh(1,from(+any,+any))?
:- modeh(1,to(+any,+any))?    :- modeh(1,at(+any,+any))?
```

These declarations say that predicates corresponding to the prepositions can occur as the head of a clause that can succeed once, and that there is no restriction on the types of the arguments to the predicates, although in other richer domains we might well want to supply such information to narrow down the search.

We also need to supply analogous information about the predicates that can be included in the body of a clause. In our case we let that be the remainder of the logical-form constants representing content word senses in the vocabulary of the system, since we do not want to impose any *a priori* guesses about the theories to be learned. Again, in a richer domain it might be sensible to incorporate some more structure to restrict the hypothesis search space. In our domain the most important predicates that actually figure in the theories learned are

```
:- modeb(1,city(+city))?        :- modeb(1,flight(+flight))?
:- modeb(1,airline(+airline))? :- modeb(1,time(+time))?
:- modeb(1,day(+day))?          :- modeb(1,meal(+meal))?
:- modeb(1,fare(+fare))?        :- modeb(1,event(+event))?
```

3. Positive and negative theories

Having read all this information into the CProgol system, we then ask it to 'generalize' the various predicates that we have specified: namely, `on/2`, `from/2`, `to/2`, and `at/2`. The resulting final theory for 'on' is

```
on(e45,sk48).            % I'd like a meal on flight...
on(e56,sk66).            % I'll have breakfast on flight...
on(e58,sk70).            % ditto
on(v72,sk71).            % what meal is on the flight?
on(sk295,sk296).         % I want information on flight...
fare(A)                    -->  on(A,B)
flight(A) & day(B)         -->  on(A,B)
flight(A) & airline(B)     -->  on(A,B)
```

These eight clauses completely predict all of the relevant positive evidence and none of the negative evidence. However, we note that the first five clauses are from the original evidence, suggesting that the algorithm was unable to find a clause that generalized from them sufficiently. Inspection of the hypotheses considered showed that candidates for specific clauses generated from the evidence included

```
event(A) & flight(B) --> on(A,B)
```

which characterizes the first three clauses adequately but also characterizes a lot of the negative data. The fourth piece of evidence gave rise to the specific clause

```
meal(A) & flight(B) --> on(A,B)
```

which suffers from the same problem, and, thus, is not considered further.

Something analogous happens with 'at', where the final theory is

```
at(sk102,'14:00').    at(sk120,'15:24').
at(sk143,'18:00').    at(sk147,'9:00').
at(sk273,'17:00').
fare(A)   --> at(A,B)
```

and where the rule covering the first five cases is considered, but does not achieve sufficient compression of the data:

```
flight(A) & time(B) --> at(A,B)
```

The remaining two theories are all but identical:

```
flight(A) --> from(A,B)    fare(A) --> from(A,B)
flight(A) --> to(A,B)      fare(A) --> to(A,B)
```

Together, these clauses seem like a reasonable characterization of the kinds of things that are in the `on/2`, `at/2`, `to/2` and `from/2` relations in the simple domain represented by the corpus. Of course, the ground unit clauses are not going to apply to any new sentences, so we must ignore them. Unfortunately, we are unable to proceed to use some of the remaining axioms directly to rule out impossible interpretations of other sentences in the way we had envisaged. The original plan was to use a theorem prover to check for consistency between the axioms and candidate logical forms for sentences. However, the theorem prover we had implemented (the Satchmo system, a simple and efficient Prolog-based system for full first-order logic, described in Manthey & Bry (1988)) has a requirement that all the clauses used in it must be 'range restricted', i.e. any variables occurring in the consequent of an implication must also appear in the antecedent. The maximally compact set of clauses discovered by CProgol as the theory of these four prepositions does not have this property: in the clauses for 'from', for example, the variable B in the head does not appear in the body.

To circumvent this problem, we take instead the more specific versions of the offending clauses produced by CProgol during the search. Even though these specific clauses will cover less data, they are all range restricted, and we can thus use them directly in the manner originally envisaged:

```
fare(A) & airline(B)   --> on(A,B).       flight(A) & day(B)  --> on(A,B).
flight(A) & airline(B) --> on(A,B).       meal(A) & flight(B) --> on(A,B).
event(A) & flight(B)   --> on(A,B).
flight(A) & city(B)    --> from(A,B).     fare(A) & city(B)   --> from(A,B).
flight(A) & city(B)    --> to(A,B).       fare(A) & city(B)   --> to(A,B).
fare(A)   & time(B)    --> at(A,B).       flight(A) & time(B) --> at(A,B).
```

This is a 'positive' theory: it describes what kind of thing happens in the ATIS domain: meals are on flights; flights are on airlines; flights are to and from cities; etc. Notice that we can also try to induce a 'negative' theory: a theory of what is *not* possible in the domain. (As before, 'not possible' needs attenuating for the general case to 'not plausible'.) To do this, we abstract out the negative data, reverse its polarity to make it all positive, and use CProgol in 'positive data only' mode to try to learn a theory of this data.

It turns out that the maximally compact theory is actually of very little use, because the negative data is very homogeneous, and these four clauses perfectly account for it all:

```
    on(A,B).      from(A,B).     to(A,B).    at(A,B).
```

Among other things, this reminds us of the difficulty of learning from positive only data, as the resulting theory is clearly over general. Moreover, we cannot use these clauses in our theorem prover, for they are not range restricted. Again, though, if we take the more specific clauses considered during the search, we have axioms that are both range restricted and at the right level of generality. Of course, we have to reintroduce the negation:

```
    event(A) & airline(B)    -->  not(on(A,B)).
    event(A) & city(B)       -->  not(from(A,B)).
    event(A) & city(B)       -->  not(to(A,B)).
    event(A) & time(B)       -->  not(at(A,B)).
```

4. Consistency checking for disambiguation

For consistency checking with Satchmo, expressions are represented in implicational form, with atomic sentences occurring as the consequents of an implication whose antecedent is 'true'. Negatives are represented as implications whose consequent is 'false'. Proving a theorem consists of adding the negation of it to the axioms and attempting to build a model of the combined set of clauses, essentially by forward chaining from atomic sentences. If every attempt to do so ends in 'false', there is no model and so the clause set is not satisfiable, and, thus, the theorem follows from the axioms. If a model can be built, then this shows that the negation of the theorem is consistent with the axioms, and so the theorem does not follow.

In order to disambiguate sentences at run time using a domain theory, there are several options we might pursue. A possible reading should be consistent with the positive ATIS theory. However, the positive ATIS theory contains no negations, and so the QLFs for bad examples, unless they happen to contain a relevant negation, will also be consistent with this theory. The positive theory says what is possible: it does not explicitly say what is impossible.

One way to use the positive theory to disambiguate would be to add a QLF to it and to see to what extent each component contributed to building a model for the sentence. We would expect that in a good QLF all the components would contribute, whereas in a bad QLF there would be at least one component that would not be integrated. Alternatively, we could negate the positive theory by making all the consequents negative, and then test QLFs for inconsistency. Good QLFs should be inconsistent, whereas bad QLFs will be consistent (since the quasi-negative theory only says that the things that really do happen don't happen, whereas a bad QLF says that the things that really don't happen do happen).

However, an alternative that is more faithful to the intended interpretation of these theories is to use the (real) negative ATIS theory to rule out bad QLFs. The simplest way to do this is to add a QLF to the axioms and test for satisfiability. If the QLF is a good one, the clause set will be satisfiable (since good QLFs are not inconsistent with axioms that say what doesn't happen). If the QLF is a bad one, the clause set will not be satisfiable, because the QLF will be asserting something that the axioms say does not happen.

To illustrate the disambiguation process, consider the set of axioms that Satchmo will have if we load in the negative ATIS theory, and the background type information. When transformed to the implicational form required by Satchmo, these will look like:

1	airline(A) & event(B) & on(B,A)	$\rightarrow$	false
2	city(A) & event(B) & from(B,A)	$\rightarrow$	false
3	city(A) & event(B) & to(B,A)	$\rightarrow$	false
4	time(A) & event(B) & at(B,A)	$\rightarrow$	false
5	true	$\rightarrow$	city(washington)
6	true	$\rightarrow$	city(atlanta)

The implicational form of the good QLF for our example sentence 'I would like the cheapest flight from Washington to Atlanta' is

7	true	$\rightarrow$	cheapest(sk0)
8	true	$\rightarrow$	flight(sk0)
9	true	$\rightarrow$	from(sk0,washington)
10	true	$\rightarrow$	to(sk0,atlanta)
11	true	$\rightarrow$	like(e0,i,sk0)
12	true	$\rightarrow$	event(e0)

Table 1.

QLFs	satisfiable	unsatisfiable
good	12	0
bad	3	16

It will be clear that these clauses can be added to the axioms without allowing 'false' to be derived. However, if we instead try to add the clause set representing the bad QLF, which will be identical to 7–12 except for

10 true $\rightarrow$ to(e0,atlanta)

we will immediately have a contradiction, since clauses 3, 6, 10 and 12, taken jointly, will allow us to conclude 'false', with A=atlanta, and B=e0.

The effectiveness of the induced theory for disambiguation was tested using the standard regime of training and then testing on held-out data. There were 79 ambiguous sentences in the original corpus. Seven sentences were held out as test data (every tenth one, the corpus was in the order derived from ATIS); the positive and negative theories were derived from the data corresponding to the remaining 72 sentences, and the held-out subset was then disambiguated against the negative theory in the manner described. The derived theories were those above, i.e. no different than for the whole corpus, and the results for disambiguation of the testing set were as in table 1. The errors are the three bad QLFs incorrectly found to be consistent with the negative theory. These were due to the sentence 'Can I have a steak dinner on that flight?' The bad QLFs have the 'steak dinner on flight' interpretation, whereas the correct one is 'On that flight, can I have a steak dinner?' However, the negative theory (unlike the positive one) does not mention meals at all, and so no contradiction is found.

5. Conclusion

We have argued that successful disambiguation may require reasoning using a 'domain theory', reasoning that cannot always be approximated by statistical methods, successful though these have been. Clearly, the ideal would be to integrate these two complementary approaches to disambiguation, to achieve a theory with sufficient deductive structure to be able to account for cases perhaps quite distant from anything seen in the training corpus, but with the robustness characteristic of statistical methods.

It is unlikely that it will prove possible to construct the required domain theories by hand. But disambiguation decisions contain a lot of implicit information about the domain theory governing those decisions. In this paper we have outlined a method by which domain theories might in principle be deduced from such disambiguation decisions, and shown that at least for small and simple domains it is indeed possible in practice to automatically induce a logically structured domain theory from a disambiguated corpus. Furthermore, this automatically learned theory can be used to successfully disambiguate other sentences from the corpus.

While the work described here is no more than a pilot experiment—the corpus is small and unrealistically homogeneous, and the theory that is learned has a rather simple deductive structure—it nevertheless suggests that it is worth trying to scale-up to more complex domains and wider coverage grammars.

I am grateful to Stephen Muggleton for introducing me to the ILP paradigm and for advice about the use of CProgol. James Cussens, Gerald Gazdar, Karen Spärck Jones and Yorick Wilks also gave me very helpful comments on the first draft of this paper.

References

Alshawi, H. & Carter, D. M. 1994 Training and scaling preference functions for disambiguation. *Comp. Ling.* **20**, 635–648.

Bar-Hillel, Y. 1960 The present status of automatic translation of languages. In *Advances in computers* (ed. F. L. Alt), vol. 1, pp. 91–163. Academic.

Collins, M. 1998 Three generative, lexicalised models for statistical parsing. In *Proc. 35th Ass. for Computational Linguistics Conf., Madrid, Spain*, pp. 16–23.

Fellbaum, C. (ed.) 1998 *WordNet: an electronic lexical database*. Cambridge, MA: MIT Press.

Hobbs, J. R., Stickel, M. E., Appelt, D. E. & Martin, P. 1993 Interpretation as abduction. *Artificial Intell.* **63**, 69–142.

Katz, J. J. & Fodor, J. A. 1964 The structure of a semantic theory. In *The structure of language: readings in the philosophy of language* (ed. J. J. Katz & J. A. Fodor), pp. 479–518. Englewood Cliffs, NJ: Prentice Hall. (Reprinted from *Language* **39**, 170–210.)

Manthey, R. & Bry, F. 1988 Satchmo: a theorem prover implemented in Prolog. In *CADE 88: 9th Conf. Automated Deduction*, pp. 415–434. Lecture Notes in Computer Science. Springer.

Muggleton, S. 1995*a* Inverse entailment and Progol. *New Generation Comput.* **13**, 245–286.

Muggleton, S. 1995*b* Stochastic logic progams. In *Proc. 5th Int. Workshop on Inductive Logic Programming, Department of Computer Science, Katholicke Universiteit, Leuven* (ed. L. De Raedt).

Muggleton, S. & De Raedt, L. 1994 Inductive logic programming: theory and methods. *J. Logic Programming* **19**, 629–679,.

Pulman, S. G. 1996 Unification encodings of grammatical notations. *Comp. Ling.* **22**, 295–328.

Pulman, S. G. 2000 Bidirectional contextual resolution. Available at www.cl.cam.ac.uk/users/sgp/bcr.ps. *Comp. Ling.* (In the press.)

Wilks, Y. A. 1975 Preference semantics. In *The formal semantics of natural language* (ed. E. Keenan), pp. 329–348. Cambridge University Press.

Discussion

Y. A. Wilks (*University of Sheffield, UK*). It is possible to separate your *goal* (learning portions of a domain theory from disambiguated text, so as to assist in disambiguating ambiguous text) from the *tool* you have used to achieve it (ILP used to acquire propositional representations of constraints on prepositional relations). Might not other tools (e.g. syntactic slot filling tools) allow you to achieve the same goal?

S. G. Pulman. Yes, indeed. However, I have chosen to pursue ILP because it has the attractive feature of delivering a domain theory with a clear (examinable, comprehensible) deductive structure.

F. Pereira (*AT & T Laboratories, Florham Park, NJ, USA*). The theory derived is propositional, not first order, as it includes only unary predicates applied to constants. Therefore, could you not have used techniques other than ILP to induce a Bayesian belief network? This would have allowed you to introduce probabilities into the derived theory, hence permitting a theory at a midway point between purely logical and purely statistical theories.

S. G. Pulman. The induced theory does not involve only unary predicates: `on`, for example, is a binary predicate, and the number of unary predicates reflects the nature of the linguistic analysis rather than anything specific to the technique. I would like to explore Bayesian networks, because they combine deductive structure with probabilities nicely. But, as you say, they are essentially propositional, and eventually I would like to see a quantificational aspect in the induced theory as well.

K. I. B. Spärck Jones (*University of Cambridge, UK*). You argue that the theories derived by your technique have the advantage of being transparent and editable. But in a more complex domain might you not get a non-transparent theory, the application of which would be much more complex?

S. G. Pulman. If such a theory were derived, it would suggest that the domain does not have much structure, i.e. that there was not much generality to be discovered in the domain.

P. Jourlin (*University of Cambridge, UK*). When you derive a contradiction in a candidate hypothesis you reject the hypothesis. But perhaps the contradiction is because the background knowledge of the domain is out of date. How do you overcome this problem?

S. G. Pulman. The background model needs to be rederived from up-to-date materials.

J. Cussens (*University of York, UK*). This is really just a comment on the previous question. Work has been done on belief revision within the ILP framework, though this is not as well developed as other parts of the framework and is hard. The key point, however, is that an updated theory can be derived without having to start from scratch.

M. Sabin (*Numerical Geometry Ltd, Cambridge, UK*). Where is the frontier between a sufficiently fine-grained grammar and a domain model?

S. G. Pulman. There probably is no frontier: one simply has to decide where to draw the line.