Efficiency in Ambiguity: Two Models of Probabilistic Semantics for Natural Language

Daoud Clarke
University of Sussex, Brighton
daoud.clarke@gmail.com

Bill Keller
University of Sussex, Brighton
billk@sussex.ac.uk

Abstract

This paper explores theoretical issues in constructing an adequate probabilistic semantics for natural language. Two approaches are contrasted. The first extends Montague Semantics with a probability distribution over models. It has nice theoretical properties, but does not account for the ubiquitous nature of ambiguity; moreover inference is NP-hard. An alternative approach is described in which a sequence of pairs of sentences and truth values is generated randomly. By sacrificing some of the nice theoretical properties of the first approach it is possible to model ambiguity naturally; moreover inference now has polynomial time complexity. Both approaches provide a compositional semantics and account for the gradience of semantic judgements of belief and inference.[1]

1 Introduction

This paper explores theoretical issues in developing an expressive and computationally tractable, probabilistic semantics for natural language. Our general approach is situated within the formal, compositional semantics developed by Richard Montague, which is augmented to allow for probabilistic judgements about truth and inference. The present work takes as a point of departure a number of key assumptions. First, an adequate semantics should provide an account of both lexical and phrasal (i.e. compositional) meaning. Second, it should provide for judgements about degrees of belief. That is, a semantics should account for beliefs that statements are more or less likely to be true, or that one statement may entail another to a certain degree. Third, an adequate computational semantics should support effective procedures for learning semantic representations and for inference.

Vector space models of meaning have become a popular approach to computational semantics. Distributional models represent word meanings as vectors of corpus-based distributional contexts and have been successfully applied to a wide variety of tasks, including, *inter alia*, word sense induction and disambiguation (Khapra et al., 2010; Baskaya et al., 2013), textual entailment (Marelli et al., 2014), coreference resolution (Lee et al., 2012) and taxonomy induction (Fountain and Lapata, 2012). The success of vector-space models is due to several factors. They support fine-grained judgements of similarity, allowing us to account for semantic gradience, for example, that the lexeme *pear* is more similar to *banana* than to, say, *cat*. Moreover, distributional vectors can be learnt in an unsupervised fashion from corpus data, either by counting occurrences of distributional contexts for a word or phrase, or by performing more sophisticated analysis on the data (Mikolov et al., 2013; Pennington et al., 2014).

Vector-based approaches differ in many regards from compositional, model-theoretic treatments of meaning such as Montague semantics or Discourse Representation Theory (Kamp and Reyle, 1993). It has proved challenging to extend vector space models to account for the way in which meanings may be composed and to support inference. The problem of developing a fully compositional, distributional semantics has recently become a very active area of research (Widdows, 2008; Mitchell and Lapata, 2008; Baroni and Zamparelli, 2010; Garrette et al., 2011; Grefenstette et al., 2011; Socher et al., 2012;

[1] The authors are grateful for the comments of a number of anonymous reviewers on earlier drafts. The work was funded as part of EPSRC EP/I037458/1, *A Unified Model of Compositional and Distributional Semantics: Theory and Applications.*

Proceedings of the 11th International Conference on Computational Semantics, pages 129–139,
London, UK, April 15-17 2015. ©2015 Association for Computational Linguistics

Lewis and Steedman, 2013) and efforts made to find a theoretical foundation (Clarke, 2012; Kartsaklis et al., 2014). Researchers have also begun to address the problem of combining the strengths of both the distributional and model-theoretic approaches (Clarke, 2007; Coecke et al., 2010; Garrette et al., 2011; Lewis and Steedman, 2013).

This paper considers an alternative strategy for developing a computational semantics. Starting with a compositional, model-theoretic semantics, this is augmented with ideas drawn from probabilistic semantics (Gaifman, 1964; Nilsson, 1986; Sato, 1995). Probabilistic semantics provides a rich seam of ideas that can be applied to the construction of a compositional semantics with desirable properties such as gradience and learnability. Below, we explore two different ways in which this might be achieved. Our objective is to map out some of the territory, to consider issues of representational adequacy and computational complexity and to provide useful guidance for others venturing into this landscape.

2 Background

2.1 Montague Semantics

In the early 1970s, Richard Montague detailed a formal treatment of natural language semantics (Montague, 1970a,b, 1973). Montague's conception of semantics was truth-conditional and model-theoretic. He considered that a fundamental aim of any adequate theory of semantics was "to characterise the notions of a true sentence (under a given interpretation) and of entailment" (Montague, 1970b). A central methodological component of Montague's approach was the *Principle of Compositionality*: the meaning of an expression is a function of the meanings of its parts and the way they are combined syntactically.

Following Montague, we assume that natural language expressions are parsed by a categorial grammar. Further, every word has an associated function with a type. Let $\mathcal{T}$ be the smallest set such that:

Basic types: $e, t \in \mathcal{T}$

Complex types: if $\alpha, \beta \in \mathcal{T}$, then $\alpha/\beta \in \mathcal{T}$.

Note that type α/β is the type of a function from type β to type α.

The set B of *basic expressions* comprises symbols denoting meanings of words. Each $b \in B$ has a type τ_b. The set Λ of *well-formed expressions*, and the extension of τ to Λ are defined as follows. Let Λ be the smallest set such that:

- $B \subseteq \Lambda$
- For every pair $\gamma, \delta \in \Lambda$ such that $\tau_\gamma = \alpha/\beta$ and $\tau_\delta = \beta$, then $\gamma(\delta) \in \Lambda$ and $\tau_{\gamma(\delta)} = \alpha$

Let Λ_τ denote the set of well-formed expressions of type τ. A *sentence* is a well-formed expression of type t.

The set D_τ of *possible denotations* of type τ is defined by:

$$\begin{aligned} D_e &= E \\ D_t &= \{\bot, \top\} \\ D_{\alpha/\beta} &= {D_\alpha}^{D_\beta} \end{aligned}$$

where E is a set of *entities*. Thus the denotation of a complex type is a function between the denotations for the types from which it is composed. An *interpretation* is a pair $\langle E, F\rangle$ such that E is a non-empty set and F is a function with domain B such that $F(b) \in D_{\tau_b}$ for all $b \in B$. A well-formed expression γ has the value $[\![\gamma]\!]$ in the interpretation $\langle E, F\rangle$, where:

- $[\![b]\!] = F(b)$ for $b \in B$
- $[\![\gamma(\delta)]\!] = [\![\gamma]\!]([\![\delta]\!])$ for $\gamma \in \Lambda_{\alpha/\beta}$ and $\delta \in \Lambda_\beta$.

subject	*verb*	*object*	m_1	m_2	m_3	m_4
john	likes	john	1	1	1	1
john	likes	mary	0	1	1	1
mary	likes	john	1	0	0	1
mary	likes	mary	0	0	1	1
john	loves	john	1	1	1	1
john	loves	mary	1	0	1	1
mary	loves	john	1	1	1	1
mary	loves	mary	1	0	1	1

Table 1: Four possible models describing relationships between John and Mary.

John likes Mary $\mu(\{m_2, m_3, m_4\}) == 0.9$
Mary likes John or Mary $\mu(\{m_1, m_3, m_4\}) == 0.8$
John likes Mary given that he loves her $\mu(\{m_3, m_4\})/\mu(\{m_1, m_3, m_4\}) = 0.7/0.8$

Table 2: Statements and their probabilities given the models in Table 1.

A sentence s is *true* in interpretation $\langle E, F\rangle$ if $[\![s]\!] = \top$, otherwise it is *false*. A *theory* T is a set of pairs $(s, \hat{s})$, where s is a sentence and $\hat{s} \in \{\top, \bot\}$ is a truth value. A *model* for a theory T is an interpretation $\langle E, F\rangle$ such that $[\![s]\!] = \hat{s}$ for every sentence $s \in T$. In this case we say that the model *satisfies* T, and write $\langle E, F\rangle \models T$.

2.2 Probabilistic Semantics

The idea of attaching probabilities to propositional truth is an old one and related to the foundation of probability itself (Keynes, 1921; Łoś, 1955). Gaifman (1964) discusses probability measures for first order calculus; the work of Sato (1995) concerns probability measures for logic programs. The idea we adopt is to associate probability measures with the space of models. Our approach is closely related in motivation to work by Cooper et al. (2014) and Goodman and Lassiter (2014).

In model-theoretic semantics, a way to view the meaning of a statement s is as the set of all interpretations $\mathcal{M}_s$ for which s is true. Probabilistic semantics extends this idea by assuming that models occur randomly. Formally, probabilities are defined in terms of a probability space $\langle \Omega, \sigma, \mu\rangle$, where Ω is the set of all models, σ is a sigma algebra associated with theories and μ is a probability measure on σ. We can then estimate the probability of a sentence s as the sum of the probabilities of all models $\mathcal{M}_s$, for s.

In general, the set of models will be infinite, but for purposes of exposition, consider a simple example with a small number of models, as in Table 1. Each column defines a different model of the relationship between John and Mary. We assume a probability distribution over models, with $P(m_1) = 0.1$, $P(m_2) = 0.2$, $P(m_3) = 0.3$, $P(m_4) = 0.4$ (all other possible models have probability zero). We can then deduce the probability of statements about John and Mary, as shown in Table 2.

3 A Probabilistic Montague Semantics

Let Ω be the set of all Montague-style interpretations, $M(T)$ the set of all models for the theory T and σ_0 the set of all sets of models that satisfy some theory: $\sigma_0 = \{M(T) : T \text{ is a theory}\}$. In general, σ_0 is not a sigma algebra, as it is not guaranteed for any two theories T_1, T_2 that $M(T_1) \cup M(T_2) \in \sigma_0$. For any Montague grammar containing a propositional fragment this would hold, but even if this is not the case we can still define a probability space by considering the sigma algebra σ generated by σ_0: the smallest sigma algebra containing σ_0. For μ a probability measure on σ, $\langle \Omega, \sigma, \mu\rangle$ is a probability space describing the probability of theories. The probability of T is defined as $\mu(M(T))$. For sentences s_1 and s_2 such that $\mu(M(\{(s_2, \top)\}) > 0$, the conditional probability $P(s_1|s_2)$ is interpreted as the *degree to which s_2 entails s_1* and defined as

$$P(s_1|s_2) = \frac{\mu(M(\{(s_1, \top), (s_2, \top)\}))}{\mu(M(\{(s_2, \top)\}))}$$

Note too that for $\mu(M(\{(s_2, \top)\})) > 0$, if s_2 *logically entails* s_1, then $P(s_1|s_2) = 1$.

3.1 Restricting the Space of Models

A key objective is to be able to learn semantics from corpus data. We describe one way this may be achieved within our framework. The central idea is to limit the number of denotations under consideration and define a probabilistic generative model for interpretations. Assume E is fixed. Let $\phi_\tau = \{[\![\lambda]\!] : \lambda \in \Lambda_\tau\}$ be the set of denotations occurring with type τ. Assume F is constrained s.t. $|\phi_\tau| = n_\tau$, where n_τ is a constant for each type satisfying $n_\tau \leq |D_\tau|$. Note that $\phi_\tau \subseteq D_\tau$ and ϕ_τ can be a lot smaller than D_τ if the range of F restricted to Λ_τ does not cover all of D_τ. We also assume that the occurring denotations are ordered, so we can write $\phi_\tau = \{d_{\tau,1}, d_{\tau,2}, \ldots d_{\tau,n_\tau}\}$. The restriction in the number of occurring denotations makes learning a distribution over models practicable, since the space of exploration can be made small enough to handle in a reasonable amount of time.

We assume that denotations are generated with probabilities conditionally independent given a random variable taking values from some set H. This gives us the following process to generate F:

- Generate a hidden value $h \in H$
- Generate $F(b) \in \phi_{\tau_b}$ for each $b \in B$ where $P(d_{\tau_b,i}|b, h) = \theta_{b,i,h}$
- Generate $d_{\alpha/\beta,i}(d_{\alpha,j}) \in \phi_\beta$ for each $d_{\alpha/\beta,i}$, $d_{\alpha,j}$, where $P(d_{\beta,k}|d_{\alpha/\beta,i}, d_{\alpha,j}, h) = \theta_{\beta,i,j,k,h}$

The parameters to be learnt are the probability distributions $\theta_{b,i,h}$ over possible values $d_{\tau_b,i}$, for each basic expression b and hidden value h, and $\theta_{\beta,i,j,k,h}$ over values $d_{\beta,k}$ for each function $d_{\alpha/\beta,i}$, argument $d_{\alpha,j}$ and hidden value h.

3.2 Learning and Inference

For a theory T and parameters θ, we compute $P(T|\theta)$ as follows. For each hidden variable h:

- Iterate over models for T. This can be done bottom-up by first choosing the denotation for each basic expression, then recursively for complex expressions. Choices must be remembered and if a contradiction arises, the model is abandoned.
- The probability of each model given h can be found by multiplying the parameters associated with each choice made in the previous step.

We can use Maximum Likelihood to estimate the parameters given a set of observed theories $\mathcal{D} = \{T_1, T_2, \ldots T_N\}$. We look for the parameters θ that maximize the likelihood

$$P(\mathcal{D}|\theta) = \prod_{n=1}^{N} P(T_i|\theta)$$

This can be maximised using gradient ascent or expectation maximisation. We have verified that this can be done for small examples with a simple Python implementation.

4 Stochastic Semantics

Our formulation of Probabilistic Montague Semantics does not adequately account for ambiguity in natural language. Although we have a probability distribution over interpretations, in any given interpretation each occurrence of a word must have the same denotation. This is unsatisfactory, as a word may exhibit different senses across a theory. For example, consider two sentences, each containing an occurrence of the word *bank*, but with different senses. In our current formulation both occurrences of *bank* are forced to have the same denotation. To allow a word's meaning to vary across a theory, one possibility is to represent different occurrences of a word by different predicates. For example, we might perform word sense disambiguation on each sentence and decide that the two occurrences of *bank* are unrelated. In this case, the first occurrence might be represented by the predicate *bank*$_1$ and the second by *bank*$_2$.

```
% Hidden variable
values(hidden, [h0, h1]).

% Types
values(word(noun, Word, Hidden), [n0, n1]).
values(word(verb, Word, Hidden), [v0, v1]).
values(word(det, Word, Hidden), [d0, d1]).
values(function(s, Value1, Value2, Hidden), [t0, t1]).
values(function(np, Value1, Value2, Hidden), [np0, np1]).
values(function(vp, Value1, Value2, Hidden), [vp0, vp1]).

evaluate(w(Word, Type), Hidden, Result) :-
  msw(word(Type, Word, Hidden), Result).
evaluate(f(Type, X, Y), Hidden, Result) :-
  evaluate(X, Hidden, XResult),
  evaluate(Y, Hidden, YResult),
  msw(function(Type, XResult, YResult, Hidden), Result).

theory([], _).
theory([truth(Sentence, Result)|Tail], Hidden) :-
  evaluate(Sentence, Hidden, Result),
  theory(Tail, Hidden).

theory(T) :-
  msw(hidden, Hidden),
  theory(T, Hidden).
```

Figure 1: A PRISM program describing probability distributions over natural language models used for our examples.

We consider an alternative which incorporates *degrees of ambiguity* into the representation of a word's meaning. This is consistent with many distributional frameworks, where a word is often represented as a vector of all of its contexts of occurrence, regardless of sense. We drop the requirement that in any given interpretation, each occurrence of a word must have the same denotation. Generalising this idea to denotations for all types results in an entirely different semantic model. It turns out that this model can be formulated in a straightforward way within the framework due to Sato and Kameya (1997). We call our new framework *Stochastic Semantics*. We no longer associate a set of models with a sentence, but instead assume that pairs of sentences and truth values are randomly generated in sequence. The probability of each pair being generated is conditionally independent of previous pairs given the hidden variable.

An illustrative implementation of Stochastic Semantics is shown in Figure 1. The code is written in PRISM (Sato and Kameya, 1997), a probabilistic extension of the Prolog logic-progamming language that incorporates probabilistic predicates and declarations. In particular, PRISM allows *random switches*. In the code, the predicate `values` is used to declare a named switch and associate with it a number of possible outcomes. The probabilistic predicate `msw` allows a random choice to made amongst these outcomes at execution time. To simplify the code, complex types of the grammar are referred to by their corresponding natural language categories: type t is represented by `s`, type (t/e) by `vp` and type $t/(e/t)$ by `np`.

The program defines how sentences are randomly assigned truth values. For example, the switch `values(word(noun,Word,Hidden),[n0,n1])` introduces a switch for nouns having two possible outcomes, `n0` and `n1`. The outcomes are conditioned on the particular choice of noun (`Word`) and on the choice of hidden variable (`Hidden`). Similarly, switches are associated with complex types. For example the values associated with a sentence (`s`) switch are conditioned on the choices of its component parts and a hidden variable, and so on.

The probability of a theory is derived by evaluating the truth conditions of its component sentences. For example, a query returning the probability of the sentence *the cat likes the dog* would be expressed as a theory $\{(\mathit{likes}(\mathit{the}(\mathit{dog}))(\mathit{the}(\mathit{cat})), \top)\}$, which can be translated into a Prolog expression in a straightforward manner.

5 Complexity

We focus on determining the probability of a sentence, as this is needed for both learning and inference.

5.1 Complexity of Probabilistic Montague Semantics

We first consider the problem of *Probabilistic Montague Semantics satisfiability* (PM-SAT) and then show that the problem of computing the probability of a sentence must be at least as hard.

Definition (PM-SAT)**.** Given a restricted set of denotations ϕ_τ for each type τ and probability distributions defined by θ, determine whether the probability of a given sentence taking the value $\top$ is non-zero.

Theorem. PM-SAT *is NP-complete with respect to the length of the input sentence.*

Proof. We first show NP-hardness by reduction from SAT. Construct a special language for propositional logic with symbols P_i of type t, $\wedge$ and $\vee$ of type $(t/t)/t$ and $\neg$ of type t/t. For example $\vee(P_1)(\neg(P_2))$ is a sentence of this language. Using a more familiar and suggestive notation it might be written as $P_1 \vee \neg P_2$. We fix a generative model for interpretations of the language as follows. Hidden values are not needed, so assume H has a single element. Further assume distributions defined by θ such that the P_i take values in $\{\top, \bot\}$ with equal probability, while the symbols $\wedge$, $\vee$ and $\neg$ simply reproduce the familiar logical truth functions for conjunction, disjunction and negation respectively, with probability 1. Note for example that there is an interpretation for which $\vee(P_1)(\neg(P_2))$ has value $\top$. On the other hand, the sentence $\wedge(P_1)(\neg(P_1))$ will have value $\bot$ for all interpretations.

It follows from the construction above that a sentence has truth value $\top$ with non-zero probability if and only if it is satisfiable. Hence we can solve SAT if we can solve PM-SAT, and so PM-SAT is NP-hard. Finally, to show NP-completeness we note that PM-SAT can be solved by a non-deterministic machine in time linear in sentence length. This is achieved by choosing a value from H, assigning every possible value to all basic expressions and then recursively to complex expressions. Any assignment that gives a non-zero probability for the sentence taking the value $\top$ will return true. So PM-SAT is in NP and is thus NP-complete. □

Let us call the problem of computing the probability of a sentence in Probabilistic Montague Semantics PM-PROB. Clearly PM-PROB is at least as hard as PM-SAT, since if we knew the probability of a sentence we would know whether it was non-zero. It follows that PM-PROB is NP-hard.

5.2 Complexity of Stochastic Semantics

Let us call the problem of computing the probability of a sentence for Stochastic Semantics, SS-PROB. We show that SS-PROB is computationally easier than PM-PROB. Note that for Stochastic Semantics there is no dependency between different parts of a sentence. Dynamic programming can then be used to store the probability distribution over possible denotations associated with each expression, so that they are computed once for each hidden value. Let L be the number of expressions in a sentence and n the maximum number of denotations for all types, i.e. the greatest value of n_τ for all types τ. The algorithm to compute the probability of a sentence is as follows. For each $h \in H$:

- For each basic expression of type τ, compute the probability distribution over ϕ_τ; this can be computed in maximum $O(n)$ time.
- Recursively for each complex expression of type τ, compute the probability distribution over ϕ_τ, this requires maximum $O(n^2)$ time since we need to iterate over possible values of the expression and the type it acts on.

For a given hidden value in H, computing the probability of a sentence requires L computations each of complexity at most n^2. The total worst-case time complexity for SS-PROB is thus $O(|H|Ln^2)$. This is linear in the number of expressions L and so linear in the length of the sentence.

Text	Hypothesis	Ent.
some cats like all dogs	some animals like all dogs	Yes
no animals like all dogs	no cats like all dogs	Yes
some dogs like all dogs	some animals like all dogs	Yes
no animals like all dogs	no dogs like all dogs	Yes
some men like all dogs	some people like all dogs	Yes
no people like all dogs	no men like all dogs	Yes
no men like all dogs	no people like all dogs	No

Table 3: Example Text and Hypothesis sentences, and whether entailment holds. Both our systems are able to learn from the data above the line that the determiner "no" reverses the direction of entailment.

Noun	Hidden	n0	n1
animals	h0	0.00	1.00
animals	h1	0.67	0.33
cats	h0	1.00	0.00
cats	h1	0.47	0.53
dogs	h0	0.70	0.30
dogs	h1	0.55	0.45
men	h0	1.00	0.00
men	h1	0.60	0.40
people	h0	0.00	1.00
people	h1	0.53	0.47

Table 4: Learnt probabilities obtained using the Stochastic Semantics implementation.

6 Discussion

A learning system such as those we have described can be adapted to the task of recognising textual entailment (Dagan et al., 2005). This task is to determine whether one natural language sentence (the "text") entails another (the "hypothesis") and thus generalizes some important natural language problems, including question answering, summarisation and information extraction. For example, a pair for which entailment holds could be translated to the following set of theories:

$$\{(s_T, \top), (s_H, \top)\}, \{(s_T, \bot), (s_H, \top)\}, \{(s_T, \bot), (s_H, \bot)\}$$

where s_T is the sentence associated with the text and s_H the sentence associated with the hypothesis.

We verified that our two systems were able to learn some simple logical features of natural language semantics on toy textual entailment examples. In particular, determiners such as "no" reverse the direction of entailment, so that while "some cats" entails "some animals", "no animals" entails "no cats" (see Table 3). As an aside, we note that while our formalism does not employ explicit representations of lexical semantics, such representations can be recovered from the learnt models. The representation of a word is a tensor rather than a vector, because there is a distinct probability distribution over possible values for each hidden value. Table 4 shows the learnt values for the nouns obtained using the Stochastic Semantics implementation. If we want to compare words we can consider the matrix entries as a flat vector and use any of the standard similarity measures (e.g. cosine).

The flexibility granted by Stochastic Semantics results in the loss of certain nice properties. A necessary consequence of incorporating stochastic ambiguity into the formalism is the failure of logical entailment. If we do not disambiguate, then a sentence may not entail itself to degree 1 since it may mean different things in different contexts. We argue that it makes sense to sacrifice some properties which are expected when handling logical expressions in order to account for ambiguity as as an inherent property of language.

It is notable that the approach that accounts for ambiguity has lower complexity. Intuitively, this is because we do not have to keep track of the interpretation previously assigned to an expression: we are free to assign a new one. This also means that the expressive power of Probabilistic Montague Semantics is greater than that of Stochastic Semantics. In the latter, the meaning of a sentence can be viewed as simply a distribution over hidden variables, whereas in the former, the meaning also includes, for example, a record of all the words contained in the sentence. It is possible that Probabilistic Montague Semantics can be made more computationally efficient by placing further restrictions on the nature of distributions over models. For example, if we have no hidden variables, then the distribution can be described efficiently using Markov Logic Networks. Again, this restriction comes at the expense of expressive power.

7 Related Work

van Eijck and Lappin (2012) presents a framework for probabilistic semantics that is motivated by the need to account for gradience effects as an intrinsic part of a model of linguistic competence. They outline a compositional semantics for a propositional language in which the probability of the truth of a sentence is defined in terms of a probability distribution over possible states of affairs (worlds). Whilst the importance of providing a computationally adequate explanation of semantic learning is emphasised, issues of the tractability of inference are not addressed.

In a computational setting, an important objection to the appeal to possible worlds is that they are not tractably representable. Cooper et al. (2014) proposes a rich type system with records in which the judgment about whether a given situation is of a given type is probabilistic. Unlike worlds, situation types (Barwise and Perry, 1983) are not maximal consistent sets of propositions, but may be as small or as large as necessary. A schematic theory of semantic learning is outlined, based on an individual's observations of situations and their types, modelled probabilistically. This account of meaning provides the basis for a compositional semantics employing a probabilistic interpretation function.

In contrast to (Cooper et al., 2014), the present work assumes a generative process over interpreting structures. In particular, this means that interpretation with respect to a given model is categorical, while meaning is defined in terms of the probability distribution over all models. By restricting the space of possible models we show that semantic learning is possible, where the primary data for learning are pairs of sentences and truth values (rather than probabilistic type judgments). A result of our learning is the ability to determine degrees of entailment between pairs of sentences.

The present work shares motivation with Goodman and Lassiter (2014), who argue for the role of uncertainty in cognition and language understanding whilst preserving a compositional, truth conditional semantics. They show how probability may be used to formalise uncertainty and the gradience of judgements about belief and inference. The approach introduces a stochastic λ-calculus to provide compositional tools for probabilistic modelling, but has not yet addressed the problem of learning.

Coecke et al. (Coecke et al., 2010) propose a framework based on category-theoretic similarities between vector spaces and pregroup grammars. Their approach is closely related to ours since it is also founded in Montague semantics: words are treated as linear functions between vector spaces. It was recently demonstrated that this approach can be extended to allow the simulation of predicate calculus using tensors (Grefenstette, 2013). Garrette et al. (2011) describe an approach to combining logical semantics with distributional semantics using Markov Logic Networks (Richardson and Domingos, 2006). Sentences are parsed into logical form using Boxer (Bos et al., 2004) and probabilistic rules are added using the distributional model of Erk and Padó (2010). Lewis and Steedman (2013) take a standard logical approach to semantics except that the relational constants used are derived from distributional clustering.

8 Conclusion

This paper has explored theoretical properties of models of meaning. The reader may question the value of a paper whose contributions are mainly theoretical in nature. Whilst we fully intend to further explore our ideas in an experimental setting, we believe that readers interested in probabilistic approaches to natural language semantics will benefit from the theoretical ideas presented here. In particular:

- We take a standard approach to natural language semantics (Montague semantics) and augment it using a standard approach (probabilistic semantics). We are thus in an area that seems natural to explore.

- We are able to demonstrate deficiencies of this approach, both in representational adequacy and computational complexity, that may provide useful guidance for others considering venturing into this landscape.

- We identify an alternative area for exploration that alleviates the difficulties associated with the first approach.

We have shown that:

1. It is possible to learn probability distributions over models directly from data by restricting the set of models, whilst retaining many of the desirable properties of full Montague semantics.
2. The problem of computing the probability that a sentence is true in this framework is NP-hard.
3. Taking account of lexical ambiguity suggests a new approach in which pairs of sentences and truth values are generated randomly. The probability that a sentence is true can then be computed in polynomial time.
4. Both models are able to learn from a few examples that quantifiers such as "no" reverse the direction of entailment.

In future work, we plan to apply our ideas to the general task of recognising textual entailment. This would involve learning from much larger datasets and provide a more stringent test of the practical application of the approach. We also plan to further investigate the relationship between our models and vector space representations of meaning. This, together with a development of the theory, may lead to interesting new ways to describe probabilistic semantics that combine logical aspects of meaning with those which are better represented distributionally.

We have so far restricted ourselves to Montague semantics, and we are thus constrained by the well-known limitations of this formalism with respect to expressing aspects of discourse. It would be interesting to investigate how well our ideas could be incorporated into a formalism such as Discourse Representation Theory.

Finally, the examples presented here deal with a very small fragment of natural language. There are many complex natural language phenomena that have been dealt with successfully within the framework of Montague semantics. This suggests that it should be possible to apply probabilistic semantics to learn about a wide range of phenomena such as quantifier scope ambiguity, intensional contexts, time and tense and indexicals, amongst others.

References

Baroni, M. and R. Zamparelli (2010). Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*.

Barwise, J. and J. Perry (1983). *Situations and Attitudes*. Cambridge, Mass.: Bradford Books. MIT Press.

Baskaya, O., E. Sert, V. Cirik, and D. Yuret (2013, June). Ai-ku: Using substitute vectors and co-occurrence modeling for word sense induction and disambiguation. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, Atlanta, Georgia, USA, pp. 300–306. Association for Computational Linguistics.

Bos, J., S. Clark, M. Steedman, J. R. Curran, and J. Hockenmaier (2004). Wide-coverage semantic representations from a ccg parser. In *Proceedings of the 20th international conference on Computational Linguistics*, pp. 1240. Association for Computational Linguistics.

Clarke, D. (2007). *Context-theoretic Semantics for Natural Language: an Algebraic Framework*. Ph. D. thesis, Department of Informatics, University of Sussex.

Clarke, D. (2012). A context-theoretic framework for compositionality in distributional semantics. *Computational Linguistics 38*(1), 41–71.

Coecke, B., M. Sadrzadeh, and S. Clark (2010). Mathematical foundations for a compositional distributional model of meaning. *CoRR abs/1003.4394*.

Cooper, R., S. Dobnik, S. Lappin, and S. Larsson (2014). A probabilistic rich type theory for semantic interpretation. In *Proceedings of the EACL 2014 Workshop on Type Theory and Natural Language Semantics (TTNLS)*, pp. 72–79. Association for Computational Linguistics.

Dagan, I., O. Glickman, and B. Magnini (2005). The pascal recognising textual entailment challenge. In *Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment*, pp. 1–8.

Erk, K. and S. Padó (2010). Exemplar-based models for word meaning in context. In *Proceedings of the ACL 2010 conference short papers*, pp. 92–97. Association for Computational Linguistics.

Fountain, T. and M. Lapata (2012, June). Taxonomy induction using hierarchical random graphs. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Montréal, Canada, pp. 466–476. Association for Computational Linguistics.

Gaifman, H. (1964). Concerning measures in first order calculi. *Israel J. Math. 2*, 1–18.

Garrette, D., K. Erk, and R. Mooney (2011). Integrating logical representations with probabilistic information using markov logic. In *Proceedings of the Ninth International Conference on Computational Semantics*, pp. 105–114. Association for Computational Linguistics.

Goodman, N. and D. Lassiter (2014). Probabilistic semantics and pragmatics: Uncertainty in language and thought. In S. Lappin and C. Fox (Eds.), *Handbook of Contemporary Semantic Theory*. Wiley-Blackwell.

Grefenstette, E. (2013). Towards a formal distributional semantics: Simulating logical calculi with tensors. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics*. Proceedings of the Second Joint Conference on Lexical and Computational Semantics.

Grefenstette, E., M. Sadrzadeh, S. Clark, B. Coecke, and S. Pulman (2011). Concrete sentence spaces for compositional distributional models of meaning. *Proceedings of the 9th International Conference on Computational Semantics (IWCS 2011)*, 125–134.

Kamp, H. and U. Reyle (1993). *From Discourse to Logic: Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*, Volume 42 of *Studies in linguistics and philosophy*. Kluwer, Dordrecht.

Kartsaklis, D., M. Sadrzadeh, S. Pulman, and B. Coecke (2014). Reasoning about meaning in natural language with compact closed categories and frobenius algebras. In J. Chubb, A. Eskandarian, and V. Harizano (Eds.), *Logic and Algebraic Structures in Quantum Computing and Information*. Cambridge University Press (to appear).

Keynes, J. M. (1921). *A treatise on probability*. Cambridge University Press.

Khapra, M., A. Kulkarni, S. Sohoney, and P. Bhattacharyya (2010, July). All words domain adapted WSD: Finding a middle ground between supervision and unsupervision. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, Uppsala, Sweden, pp. 1532–1541. Association for Computational Linguistics.

Lee, H., M. Recasens, A. Chang, M. Surdeanu, and D. Jurafsky (2012, July). Joint entity and event coreference resolution across documents. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, Jeju Island, Korea, pp. 489–500. Association for Computational Linguistics.

Lewis, M. and M. Steedman (2013). Combined distributional and logical semantics. *Transactions of the Association for Computational Linguistics 1*, 179–192.

Łoś, J. (1955). On the axiomatic treatment of probability. *Colloq. Math 3*, 125–137.

Marelli, M., L. Bentivogli, M. Baroni, R. Bernardi, S. Menini, and R. Zamparelli (2014, August). Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, Dublin, Ireland, pp. 1–8. Association for Computational Linguistics and Dublin City University.

Mikolov, T., K. Chen, G. Corrado, and J. Dean (2013). Efficient estimation of word representations in vector space. *ICLR Workshop*.

Mitchell, J. and M. Lapata (2008, June). Vector-based models of semantic composition. In *Proceedings of ACL-08: HLT*, Columbus, Ohio, pp. 236–244. Association for Computational Linguistics.

Montague, R. (1970a). English as a formal language. In B. V. et al. (Ed.), Linguaggi nella Societ e nella Tecnica, pp. 189–223.

Montague, R. (1970b). Universal grammar. *Theoria 36*, 373–398.

Montague, R. (1973). The proper treatment of quantification in ordinary english. In J. Hintikka, J. Moravcsik, and P. Suppes (Eds.), *Approaches to Natural Language*, pp. 221–242. Reidel, Dordrecht.

Nilsson, N. (1986). Probabilistic logic. *Artificial Intelligence 28*, 71–87.

Pennington, J., R. Socher, and C. Manning (2014, October). Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, pp. 1532–1543. Association for Computational Linguistics.

Richardson, M. and P. Domingos (2006). Markov logic networks. *Machine learning 62*(1-2), 107–136.

Sato, T. (1995). A statistical learning method for logic programs with distribution semantics. In *Proceedings of the 12th International Conference on Logic Programming (ICLP95)*.

Sato, T. and Y. Kameya (1997). Prism: a language for symbolic-statistical modeling. In *IJCAI*, Volume 97, pp. 1330–1339. Citeseer.

Socher, R., B. Huval, C. D. Manning, and A. Y. Ng (2012). Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 1201–1211. Association for Computational Linguistics.

van Eijck, J. and S. Lappin (2012). Probabilistic semantics for natural language. In Z. Christoff, P. Galeazzi, N. Gierasimszuk, A. Marcoci, and S. Smets (Eds.), *Logic and Interactive Rationality (LIRA)*, Volume 2, pp. 17–35. ILLC, University of Amsterdam.

Widdows, D. (2008). Semantic vector products: Some initial investigations. In *Proceedings of the Second Symposium on Quantum Interaction, Oxford, UK*, pp. 1–8.