The Dissertation Committee for Taesun Moon
certifies that this is the approved version of the following dissertation:

Word meaning in context as a paraphrase distribution: Evidence, Learning, and Inference

Committee:

Katrin Erk, Supervisor

Jason Baldridge

Colin Bannard

Inderjit Dhillon

Raymond Mooney

Word meaning in context as a paraphrase distribution: Evidence, Learning, and Inference

by

Taesun Moon, B.A.; M.A.; M.A.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2011

Acknowledgments

I thank Prof. Katrin Erk for her unwavering support, insight, rigor, and guidance over the years I have been a graduate student at the university. No less gratitude is due to Prof. Jason Baldridge, who first introduced me to and got me hooked on computational linguistics and who has given me endless support and guidance in my research. I thank the other members on my committee, Profs. Colin Bannard, Inderjit Dhillon, and Raymond Mooney for their time and the helpful feedback they have provided. I thank Megan Biesele and the trainees of the Kalahari Peoples Fund for the three productive summers I have spent in the Kalahari desert.

Most importantly, I thank my family for everything I am.

Word meaning in context as a paraphrase distribution: Evidence, Learning, and Inference

Publication No. ____________

Taesun Moon, Ph.D.
The University of Texas at Austin, 2011

Supervisor: Katrin Erk

In this dissertation, we introduce a graph-based model of instance-based usage meaning that is cast as a problem of probabilistic inference. The main aim of this model is to provide a flexible platform that can be used to explore multiple hypotheses about usage meaning computation. Our model takes up and extends the proposals of Erk and Padó [2007] and McCarthy and Navigli [2009] by representing usage meaning as a probability distribution over potential paraphrases. We use undirected graphical models to infer this probability distribution for every content word in a given sentence. Graphical models represent complex probability distributions through a graph. In the graph, nodes stand for random variables, and edges stand for direct probabilistic interactions between them. The lack of edges between any two variables reflect independence assumptions. In our model, we represent each content word of the sentence through two adjacent nodes: the *observed* node represents the surface form of the word itself, and the *hidden* node represents its

usage meaning. The distribution over values that we infer for the hidden node is a *paraphrase distribution* for the observed word. To encode the fact that lexical semantic information is exchanged between syntactic neighbors, the graph contains edges that mirror the dependency graph for the sentence. Further knowledge sources that influence the hidden nodes are represented through additional edges that, for example, connect to document topic. The integration of adjacent knowledge sources is accomplished in a standard way by multiplying factors and marginalizing over variables.

Evaluating on a paraphrasing task, we find that our model outperforms the current state-of-the-art usage vector model [Thater et al., 2010] on all parts of speech except verbs, where the previous model wins by a small margin. But our main focus is not on the numbers but on the fact that our model is flexible enough to encode different hypotheses about usage meaning computation. In particular, we concentrate on five questions (with minor variants):

Nonlocal syntactic context Existing usage vector models only use a word's direct syntactic neighbors for disambiguation or inferring some other meaning representation. Would it help to have contextual information instead "flow" along the entire dependency graph, each word's inferred meaning relying on the paraphrase distribution of its neighbors?

Influence of collocational information In some cases, it is intuitively plausible to use the selectional preference of a neighboring word towards the target to determine its meaning in context. How does modeling selectional preferences into the model affect performance?

Non-syntactic bag-of-words context To what extent can non-syntactic information in the form of bag-of-words context help in inferring meaning?

Effects of parameterization We experiment with two transformations of MLE. One interpolates various MLEs and another transforms it by exponentiating pointwise mutual information. Which performs better?

Type of hidden nodes Our model posits a tier of hidden nodes immediately adjacent a surface tier of observed words to capture dynamic usage meaning. We examine the model by varying the hidden nodes such that in one the nodes have actual words as values and in the other the nodes have nameless indexes as values. The former has the benefit of interpretability while the latter allows more standard parameter estimation.

Portions of this dissertation are derived from joint work between the author and Katrin Erk [submitted].

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Word sense disambiguation (WSD) is *the* dominant task in the subfield of computational linguistics that deals with the meaning of words in context [Agirre and Edmonds, 2006], i.e. computational semantics. In WSD, a system is given a naturally occurring sentence that contains a target word of interest and a disjoint candidate sense inventory, generally a list of dictionary definitions for the word. The goal of the system is then to choose the item out of the inventory that best fits the target word in the sentence. The system is evaluated based on some aggregate measure over the system's output on all the target words for all the sentences that it has been handed to disambiguate.

Consider this old and frequently cited example in WSD and machine translation:[1]

(1.1) Little John was looking for his toy box. Finally he found it. The box was in the *pen*. John was very happy.

The hypothetical situation posited by Bar-Hillel is that a machine translation system has been given the above text. Presumably, the most challenging aspect of the text

[1] The example is from Bar-Hillel [1960], an influential report that determined the course of machine translation research in the 60s and 70s. This example was cited in Gale et al. [1992] and Agirre and Edmonds [2006] among many others.

sample for the system is that *pen* is ambiguous and can mean (or map to) one of two disjoint sense items: (1) a writing implement (2) an enclosure. And Bar-Hillel's conclusion was that:

> no existing or imaginable program will enable an electronic computer to determine that the word *pen* in the given sentence within the given context has the second of the above meanings.

and that the preposterousness of a program that can properly disambiguate *pen* and the like is an insurmountable barrier to machine translation.

The assumption underlying Bar-Hillel's assertion is that WSD is a critical component to the success of any machine translation (MT) system. However, it's obvious that this isn't entirely the case when some of the most influential models in contemporary MT have been able to significantly improve on existing rule-based models of MT without incorporating WSD [Brown et al., 1993]. Similarly WSD was once assumed to be a critical component for building information retrieval (IR) systems. The overwhelming success of an Internet search engine [Brin and Page, 1998] that does not have a WSD component indicates that there are ways of effectively sidestepping the issue of ambiguity in natural language.

Nonetheless, it is clear that ambiguity is an inherent component of human language and existing systems will at some point have to address the issue if further improvements in performance are to be gained. Furthermore, some recent research has been able to show that integrating WSD into an application such as MT [Carpuat and Wu, 2007, Chan et al., 2007] or IR [Stokoe, 2005] can improve performance.

Though the original motivation for WSD lay in applications, the task has important implications that are not applied and has given birth to many subtasks and variants that ask or influence important questions regarding the definition of word meaning itself, cognitive issues about word senses, computationally oriented practical issues of data curation, modeling and evaluation and much more. These concerns are partly what initiated the first open challenge workshop for WSD systems in 1998 called SensEval [Kilgariff and Palmer, 1998, Kilgarriff and Palmer, 2000]. There is a broad consensus on what the critical issues with the dominant WSD task paradigm are and these have been highlighted since the first SensEval, at both SensEval and elsewhere [Wilks, 2000, Agirre and Edmonds, 2006]. Some of the important issues that have been raised are that: (1) the sense inventories are inconsistent [Kilgarriff, 1997, Wilks, 2000] (2) human annotators often have a hard time using the inventories [Kilgarriff and Rosenzweig, 2000] (3) the notion of disjoint word sense is too restrictive and cognitively invalid [Erk and McCarthy, 2009, Erk et al., 2009].

For the moment, we focus only on the last issue of disjoint word sense. We will discuss the other issues in more depth in Chapter 2. Consider the following example:

(1.2) This can be justified thermodynamically in this case, and this will be done in a separate *paper* which is being prepared.

The example is taken from SemCor [Fellbaum, 1998] which is a corpus that has been sense-tagged—i.e. tagged with a predefined sense inventory—for all content words.

The sense inventory comes from WordNet [Miller, 1995]. In an extensive discussion in Erk and McCarthy [2009] the seven potential sense items for *paper* according to WordNet and SemCor are defined as follows:

1. a material made of cellulose pulp derived mainly from wood or rags or certain grasses
2. an essay (especially one written as an assignment)
3. a daily or weekly publication on folded sheets; contains news and articles and advertisements
4. a medium for written communication
5. a scholarly article describing the results of observations or stating hypotheses
6. a business firm that publishes newspapers
7. the physical object that is the product of a newspaper publisher

According to SemCor, the answer is 5. Erk and McCarthy, who had no involvement in the creation of SemCor, conducted confirmatory human experiments by asking the subjects to judge the applicability of each of the seven senses above to Example 1.2 on an integer grade of 1 to 5, with 5 being the most applicable and 1 the least. It turned out that human subjects gave several items that are not 5. *scholarly article*, high scores. In other words, though the creators of SemCor labeled Example 1.2 such that *scholarly article* would be the only answer accepted as correct, people asked to judge the candidates found that other candidates such as 2. *essay* or 4. *medium* were quite plausible. Though the task for SemCor was set up under the simplifying assumption that the candidate senses are disjoint, people judged otherwise.

The problem becomes even more stark for lexical items that have only abstract definitions. Consider the following example with the target *arguments*:

(1.3) This question provoked *arguments* in America about the Norton Anthology of Literature by Women, some of the contents of which were said to have had little value as literature.

The example is taken from the corpus created for the task defined in Mihalcea et al. [2004] and is extensively examined in Erk et al. [2009]. Mihalcea et al. similarly used WordNet for labeling noun targets. In WordNet, the possible definitions/sense items for *argument* are:

1. a fact or assertion offered as evidence that something is true
2. a contentious speech act; a dispute where there is strong disagreement
3. a discussion in which reasons are advanced for and against some proposition or proposal
4. a summary of the subject or plot of a literary work or play or movie
5. (computer science) a reference or value that is passed to a function, procedure, subroutine, command, or program
6. a variable in a logical or mathematical expression whose value determines the dependent variable; if f(x)=y, x is the independent variable

7. a course of reasoning aimed at demonstrating a truth or falsehood; the methodical process of logical reasoning

Unlike SemCor, the annotators for Mihalcea et al. [2004] were allowed to assign as many senses that wanted to each target and for this particular example, they chose items 1, 2, 3 and 7. Erk et al. [2009] again independently conducted confirmatory experiments of human judgments and this time found that the overlap between their subjects and the original annotators for this example was decent.

The above methods of creating and evaluating corpora based on WSD of disjoint senses suffer from certain defects, at least one of which is that the inventories are not as disjoint as they should be [Snyder and Palmer, 2004]. That this would be a problem is obvious in a way because it is highly unlikely that all the meanings for all people of all words in all contexts can be partitioned such that (1) such a set is countable (2) such a set is pairwise disjoint. Yet that is the assumption that underlies the proposition of disjoint word sense. A more practical problem is that the sense inventories are manually compiled and this is usually expensive and time-consuming.

These problems have led researchers to investigate other representations of word meaning that are not wedded to a particular lexical inventory, are not disjoint, perhaps are not even enumerable. One such alternative definition is based on the notion of paraphrases, i.e. the use of semantically similar and syntactically valid substitutions of the target word in context to represent the meaning of the target. The benefit of this approach is that the paraphrases are generally not disjoint.

Furthermore, the paraphrase inventory can be compiled manually, automatically, or through a mixture of both as opposed to definitions in a dictionary which always require trained lexicographers to create.

We illustrate the notion of paraphrases with the following example:

(1.4) Some payments occurred after the traffickers had been indicted by federal law enforcement agencies on drug *charges*, in others while traffickers were under active investigation by these same agencies.

Here, the target of interest is *charges*. McCarthy and Navigli [2009] asked several people to propose words which could be replaced with *charges* without changing the meaning of the sentence too much. The annotators proposed *accusation*, *allegation*, *offence*, *indictment* to varying degrees which were converted to weights. The target was then labeled with all four items and the weight with which they were proposed.

This alternative proposal frees us from having to choose a single best sense even when there are several good options. It also frees us from having to provide verbal descriptions or definitions of the senses of *accusation*, *allegation*, *offence*, *indictment*. Our proposal—which is just a rehash of what has been proposed by Erk and Padó [2007] and McCarthy and Navigli [2009]—is that the meaning of a word in context is the set of paraphrases that can be proposed for it along with their weights.

This novel representation of "graded word sense" was proposed programmatically by Erk and Padó [2007] and took its motivation from prototype theories of sense representation in cognitive science [Rosch, 1975] where word types (among

Sentence	Substitutes
Some payments occurred after the traffickers had been indicted by federal law enforcement agencies on drug charges, in others while traffickers were under active investigation by these same agencies. (# 1812)	accusation 2; allegation 2; offence 1; indictment 1
We study the methods and concepts that each writer uses to defend the cogency of legal, deliberative, or more generally political prudence against explicit or implicit charges that practical thinking is merely a knack or form of cleverness . (# 1813)	accusation 2; allegation 3; criticism 1

Figure 1.1: Lexical substitution example items for *charge*. The four digit numbers at the end of each sentence are the unique identifiers in the corpus. The column on the right lists the sense items with non-zero weights in the labeling scheme, i.e. items that have been chosen by at least one annotator for the target *charge*. The integers to the left of the sense items correspond to weights—the number of annotators who have chosen the given sense item.

others) can belong to different prototypes with varying degrees of membership. It has also been established as an open task in SemEval 2007 [McCarthy and Navigli, 2007]. It has been further investigated upon in Erk et al. [2009], Erk and McCarthy [2009], Erk and Pado [2010], Thater et al. [2010] *inter alia*.

A labeled corpus that can test models of graded sense representation was created by McCarthy and Navigli [2009]. The labeled corpus is called the English Lexical Substitution dataset (LexSub). The corpus was labeled by asking multiple annotators to propose substitutes (i.e. paraphrases) for a target word in a sentence. Each annotator was allowed to propose up to three paraphrases for the target and each proposal was given a count of one. Then for each paraphrase for a target, the number of annotators who proposed it is tabulated and defined to be the weight for that paraphrase for that target in that sentence.

We reexamine Example 1.4 in Figure 1.1. The four digit numbers at the end of each sentence are the unique identifiers in the corpus. The column on the right lists the paraphrases with non-zero weights in the labeling scheme. The weights are merely the number of annotators who proposed the given paraphrase for the word in question. Only the items that have been chosen by at least one annotator for the target *charge* in the given sentences are listed. The integers to the left of the sense items correspond to weights—the number of annotators who have chosen the given sense item.

The two usages of *charge* in sentences #1821 and #1813 are highly similar and in fact share two substitutes: *accusation* and *allegation.* This representation captures subtlety that disjoint sense representation as a winner-take-all scheme is incapable of. It shows that there are elements of the paraphrase *criticism* in sentences #1813 which #1812 does not display. On the other hand, there are connotations of *offence* and *indictment* in #1812 that are not in #1813. Even among the shared *accusation* and *allegation*, we see that there are varying degrees of membership.

1.1 Graded word sense and probabilistic modeling

The notion of graded word sense is a fairly novel idea and as such most practitioners in the field of computational semantics will find it unfamiliar. Nonetheless, we hope they find the idea intuitive and useful. Here we provide a brief sketch of how the notion of graded word sense can be transformed to integrate into a probabilistic model. Once the transformation is complete, matters such as computability and inference are conventional and standard within the framework of probabilistic graphical models. There is a vast literature on probabilistic graphical models that the reader can consult [Beal, 2003, MacKay, 2003, Wainwright and Jordan, 2008]. Since our investigation adds nothing new to the field of probabilistic graphical models, we will mostly focus on the modeling aspects of graded word sense.

Because graded word sense is defined over a high-dimensional feature space that is still finite (i.e. all the paraphrases that are possible for the words in a language) and because the weights associated with each paraphrase for each context is non-negative, the program lends itself easily to probabilistic modeling. The basis elements of the feature space (again, all the paraphrases that are possible for the words in the language) can be mapped one-to-one to values of a single random variable. The non-negative weights over all paraphrases can be normalized to sum to one.

The meaning of a word can be defined to be a posterior distribution over the paraphrases for the target word in context. Furthermore, if we take this redefinition of graded word sense as posterior distribution over paraphrases given some evidence even further, we can utilize the framework of probabilistic graphical models to define what we mean by context with considerable flexibility and conduct marginal inference.

By conducting inference to discover word meaning in context, we can no longer refer to the activity of the probabilistic models we are about examine as disambiguation. Disambiguation has a firmly entrenched sense of picking the best candidate for a target word from a finite inventory. In contrast, there is no concept of choosing best or worst in marginal inference.[2] Instead, marginal inference provides a complete picture of all the potentialities of a random variable in the presence of external influence. Once the shape of the graph within which the target resides has been determined and the parameters that determine interaction between vertices in the graph have been provided, the process of marginal inference doesn't generate a best element but a complete distribution. In other words, marginal inference returns a function that is sensitive to the context.

We will not discuss the details of probabilistic graphical models and how they can relate to graded word sense any further. We leave that for Chapter 3 where it is given a more extended treatment. The important thing to note is that the representation generated by these models is highly attuned to the context that a target occurs in and builds distinct representations for each occurrence or usage of a word. This is opposed to the standard disjoint representations used in WSD where the sense inventory is permanently fixed and any chosen sense item for a given target is more or less a fixed square peg hammered into a constantly shifting hole.

The marriage of graded word sense with probabilistic graphical models gives us considerable power and flexibility to explore diverse aspects of the types of evidence that can influence word meaning. It has long been known that incorporating diverse sources of evidence such as syntactic dependency labels, bag-of-words context within some finite window, immediate left and right context, etc. helps perfor-

[2] The issue is one of terminology rather than what is or what isn't possible with probabilistic graphical models. Finding the set of best or maximum values for a set of random variables is solved through the max-product algorithm and should be distinguished from marginal inference. They are both, however, instances of probabilistic inference.

mance in WSD [McRoy, 1992, Bruce and Wiebe, 1994]. Furthermore, much of the literature hints that local information (e.g. left and right context, immediate dependency parents and children) is much more important than global information (e.g. document level bag-of-words context) [Yarowsky, 1993, Padó and Lapata, 2007].

These known facts about building WSD systems can be incorporated easily and flexibly within a probabilistic graphical framework and we will investigate accordingly. Furthermore, this framework allows us to do something that, to the best of our knowledge, has never been done before: infer the meaning of every word in a sentence in relation to the inferred meaning of every other word in the sentence. The analogy with WSD would be if every word in a sentence is disambiguated not only based on the surface evidence but also based on how the words in the sentence are disambiguated. Our experiments in terms of this type of global inference have been moderately disappointing but we believe it is because we have not yet fully explored all possibilities.

1.2 Overview of the dissertation

In this dissertation, we introduce a graph-based model of instance-based, usage meaning that is cast as a problem of probabilistic inference. Models that consider usage meaning ask fundamental questions about knowledge sources to be used in inference/computation. Therefore, the main aim of this model is to provide a flexible platform that can be used to explore multiple hypotheses about usage meaning computation. Our model takes up and extends the proposals of Erk and Padó [2007] and McCarthy and Navigli [2009] by representing usage meaning as a probability distribution over potential paraphrases. We use undirected graphical models to infer this probability distribution for every content word in a given sentence. Graphical models represent complex probability distributions through a graph. In the graph, nodes stand for random variables, and edges stand for direct probabilistic interactions between them. The lack of edges between any two variables reflect independence assumptions. In our model, we represent each content word of the sentence through two adjacent nodes: the *observed* node represents the surface form of the word itself, and the *hidden* node represents its usage meaning. The distribution over values that we infer for the hidden node is a *paraphrase distribution* for the observed word. To encode the fact that lexical semantic information is exchanged between syntactic neighbors, the graph contains edges that mirror the dependency graph for the sentence. Further knowledge sources that influence the hidden nodes are represented through additional edges that, for example, connect to document topic. The integration of adjacent knowledge sources is accomplished in a standard way by multiplying factors and marginalizing over variables.

Evaluating on a paraphrasing task, we find that our model outperforms the current state-of-the-art usage vector model [Thater et al., 2010] on all parts of speech except verbs, where the previous model wins by a small margin. But our main focus is not on the numbers but on the fact that our model is flexible enough to encode different hypotheses about usage meaning computation. In particular, we concentrate on five questions (with minor variants):

Nonlocal syntactic context Existing usage vector models only use a word's direct syntactic neighbors for disambiguation or inferring some other meaning representation. Would it help to have contextual information instead "flow" along the entire dependency graph, each word's inferred meaning relying on the paraphrase distribution of its neighbors? Consider Example 1.5 and Example 1.6. The word *class* has multiple readings, including *group of students* and *social caste.* The context *undergraduate* in Example 1.5 makes it clear that *group of students* is the intended reading. This in turn makes the *speak to* reading of *address* much more likely than the alternative *apply oneself to*; social class as an abstract concept is not a group of people and hence is not usually talked to, while student bodies often are.[3] Existing usage vector models[4] do not use information from more distant nodes in the syntactic graph, but our model can use it because its graph edges mirror the complete dependency graph.

(1.5) The teacher <u>addressed</u> the undergraduate class.

(1.6) [The parliament introduced new laws]. They <u>address</u> class as an issue.

Influence of collocational information In some cases, it is intuitively plausible to use the selectional preference of a neighboring word towards the target to determine its meaning in context. To contextualize *take* in Example 1.7, where it

[3] The presence and absence of determiners also plays a role in determining the meaning of *class* in a given context. We will also examine whether the presence of such functions words can influence inference.

[4] These are models of word meaning in context that compute individual representations for each word instance as points in vector space. For example, Kintsch [2001], Erk and Padó [2008], Erk [2009].

means something like *ride*, it will be helpful to know that *bus* has paraphrases like *autobus, coach, omnibus*, as this yields more information than the observed word *bus* alone. However, in Example 1.8, the paraphrases of *long* will be irrelevant or maybe even harmful for computing a paraphrase distribution for *take* because *take long* is a collocation. We can test the influence of collocations in our model through the graph nodes that stand for the observed words.

(1.7) They took the bus.

(1.8) It didn't take long.

Non-syntactic bag-of-words context To what extent can non-syntactic information in the form of bag-of-words context help in inferring meaning? Though it seems like this should be always, it's more the case that this information is relevant sometimes and sometimes it isn't [Leacock et al., 1998]. To examine the complex interaction between bag-of-words context, local syntactic context and non-local syntactic context, we examine two different types of bag-of-words context in relation to the remaining features. For one, we examine the effects of document level bag-of-words in the form of document topic. We do this through a standard topic model [Blei et al., 2003]. For the other, we examine the effects of bag-of-words as sentence where every content word is connected to every other content word. This is equivalent to considering a sentence as a complete graph over its content words without regard to the syntactic relations between the tokens.

Effects of parameterization We use maximum likelihood estimates derived from large, parsed corpora as parameters for potential functions between two connected nodes over a syntactic relation. However, it is well-known that raw frequency counts often have pernicious effects on inference tasks. Therefore, we experiment with two transformations of MLE. One interpolates various MLEs and another transforms it by exponentiating pointwise mutual information. Which performs better?

Type of hidden nodes Our model posits a tier of hidden nodes immediately adjacent to the surface tier of observed words to capture dynamic usage meaning. Our first formulation is simpler in that it assumes valid paraphrases constitute the value space of the hidden nodes. Any and all words that can substitute for a surface word form the value space. This makes the model output easy to interpret since the inferred meaning of a word is a probability mass function over meaningful paraphrases. However, to fit this notion of paraphrases, we take an unorthodox approach in estimating parameters over hidden paraphrase transitions[5] from surface dependency relation counts. To examine a more orthodox perspective, we investigate an alternative formulation where the hidden nodes are nameless indexes as are found in unsupervised part-of-speech tagging [Moon et al., 2010] or topic modeling [Blei et al., 2003]. For this formulation, we learn parameters from the training data through Gibbs sampling.

[5] To facilitate understanding, this is analogous to state transition parameters in state sequence models except that the graphs in our models are not sequential.

1.2.1 Plan of the dissertation

In Chapter 2, we provide more in-depth background on the task of word sense disambiguation (WSD) including its history and developments up to the current day. We address issues faced by current practice in WSD when using meaning representations that posit word sense as finite and disjoint. We then discuss the alternative meaning representation that is taken up by the dissertation—that of graded word sense—with discussions of related literature and corpora.

We unveil our model and discuss it in much greater depth in Chapter 3. First, we present how word meaning in context is represented as a probability distribution with extensive examples. We then provide a summary overview of graphical models including directed and undirected graphical models, issues regarding inference with such models, and close the section with related literature. We finally discuss the model proper in terms of the diverse graph topologies it can accommodate, the evidence it can take into account, the different parameterizations, and how marginal inference is conducted with loopy belief propagation due to the presence of loops in the graphs.

We next describe the data and software tools we use and list some implementation details in Chapter 4. We also discuss the evaluation measures for our experiments: generalized average precision and weighted accuracy.

In Chapter 5, we discuss experiments and their results. We begin by tabulating the results from our best performing model variant and contrast it with state-of-the-art benchmarks and baseline models, both of which it beats. We then show example output from the models and discuss in great detail all the variants of the models that have been experimented with—variants in terms of graph topology, evidentiary nodes, parameterization, etc.

We conclude with an overview and a discussion of directions for future work to cover some of the deficiencies in the current work Chapter 6.

1.2.2 Contributions

The primary contributions of the dissertation lie in (1) the novel application of undirected graphical models to word meaning (2) recasting the program of graded word sense as proposed in Erk and Padó [2007] and McCarthy and Navigli [2009] to one where word meaning is represented as a probability distribution. With the application of undirected graphical models to word meaning as probability distribution, the problem of resolving the meaning of a word in context becomes a problem of marginal inference. Features in vector space models of word meaning become evidentiary nodes in graphical models which provide a unified framework for conducting inference in a tractable manner over such evidence that scales well in the face of increased complexity—e.g. the model can implicitly incorporate entire dependency trees as features which is impossible for vector space models at the moment. The most expansive model that we are aware of [Thater et al., 2010] incorporates second degree vectors over dependency edges; and there are no models which incorporate third degree or higher vectors for obvious reasons of tractability.

Chapter 2

A background on word meaning

The topic of the meanings of words and the even more general issue of semantics as it arises in natural language is one of the central issues in computational linguistics and is properly called lexical semantics. It is for the almost tautological fact that words are foundational building blocks of meaning in human language. In this field of inquiry lexical inventories such as dictionaries or thesauri that were compiled by linguists and lexicographers have played a central role. As such, we discuss these inventories and the notion of discrete word sense and further examine how the use of such inventories influenced the development of word sense disambiguation (WSD) as a task in §2.1. We then look into an important alternative family of meaning representations that each derive customized representations for every different use of some target word. These representations fall under the umbrella of **word usage models** and give us considerable flexibility in the phenomena we can examine. Finally, we discuss how these new representations help us move away from fixed lexical inventories and end with graded word sense which is the representation used in our models.

2.1 Word sense disambiguation

As noted in the introduction, word sense disambiguation (WSD) is the most widely and frequently tackled task in lexical semantics. Kilgarriff [1997] defines word sense disambiguation as follows:

> Many words have more than one meaning. When a person understands a sentence with an ambiguous word in it, that understanding is built on the basis of just one of the meanings. So, as some part of the human language understanding process, the appropriate meaning has been chosen from the range of possibilities.

Under this definition, WSD is also one of the oldest tasks in computational linguistics and still remains challenging today. The task itself was conceived in an influential position paper [Weaver, 1949] on using computers to automatically conduct machine translation (MT). It was obvious early on that the ambiguity of words should be a considerable challenge for MT. Since then, there has been considerable work on WSD from the 60s and on [Masterman, 1961, Weiss, 1973, Lesk, 1986]. The broad outlines of the task remained more or less the same as described previously; until a significant change occurred when the first open challenge workshop in WSD was held in 1998 called SensEval [Kilgariff and Palmer, 1998, Kilgarriff and Palmer, 2000]. It was sponsored by the Association for Computational Linguistics and modeled on "DARPA competitive evaluations for speech recognition, dialogue systems, information retrieval and information extraction." [Kilgarriff and Palmer, 2000]. Since then, four more SensEval/SemEval workshops in WSD have been held every three years[1] with each bringing new perspectives that highlighted or attempted to resolve previous limitations and problems in WSD.

Diversity was introduced to the problem by adding more languages and creating larger data sets. Each workshop no longer dealt with a monolithic WSD problem but introduced a group of smaller problems, subtasks, variations and more along with the standard WSD. For example, some problems were expanded so that there may be more than one target of interest in a given sentence [Snyder and Palmer, 2004, Agirre et al., 2010]. Or certain subtasks allowed more than one sense item to be proposed and accepted as the answer [Litkowski, 2004, McCarthy and Navigli, 2007]. The tasks were expanded even further to incorporate tasks which did not fit under the fold of WSD. Some important tasks that fall under lexical semantics but are not WSD are semantic role labeling [Litkowski, 2004] and textual entailment [Yuret et al., 2010].

Today, state-of-the-art performance on WSD for WordNet senses—a standard lexical database developed by Miller [1995] which provides the sense inventory that is used to label training and test corpora used in many WSD tasks—is at only around 70-80% accuracy [Edmonds and Cotton, 2001, Mihalcea et al., 2004]. One reason for this less than optimal performance was due to the fact that sense distinctions in WordNet are too fine-grained. This led Palmer et al. [2007] to combine fine-grained senses into coarse-grained senses. This correction has led to considerable advances in WSD performance, with accuracies of around

[1] They were held in 2001 [Edmonds and Cotton, 2001], 2004 [Mihalcea and Edmonds, 2004], 2007 [Agirre et al., 2007], 2010 [Erk and Strapparava, 2010]

90% [Pradhan et al., 2007]. But this figure averages over lemmas, and the problem remains that while WSD works well for some lemmas, others, like *leave.v*, continue to be tough [Chen and Palmer, 2009].

2.1.1 Disjoint word sense

In WSD, polysemy is typically modeled through a list of dictionary senses thought to be mutually disjoint, such that each occurrence of a word is characterized through one best-fitting dictionary sense. However, the underlying assumption that each word has clear, disjoint senses has been drawn into question by linguists, lexicographers and psychologists [Tuggy, 1993, Cruse, 1995, Kilgarriff, 1997, Hanks, 2000, Kintsch, 2007]. Nonetheless, there are many practical reasons for making such assumptions. In many cases, the discrete sense inventories came from machine readable dictionaries [Lesk, 1986] or thesauri [Masterman, 1961]. Since the late 90s, when important non-traditional lexical databases such as WordNet [Miller, 1995] or FrameNet [Baker et al., 1998] grew in popularity, the notion of discrete sense has become more expansive.

But regardless of the details of the source of the inventory, in following this program of word meaning or participating in a WSD task, one is implicitly agreeing that the sets of meanings of words are finite, discrete and pairwise exclusive. Furthermore, while the context of a word may help to disambiguate or choose among the set of candidates, any possible meanings are bound to the set that has been defined in such databases. It has often been argued that this simplistic view of words having a finite, disjoint set of meanings is restrictive [Erk and Padó, 2007, Erk, 2009], not realistic, and even from an application oriented viewpoint, very difficult to integrate into existing systems such that it enhances performance [Sanderson, 2000, Agirre and Edmonds, 2006]. It should be noted, however, that promising initial discoveries have been made very recently with regard to the utility of WSD for applications such as information retrieval [Stokoe, 2005] and machine translation [Carpuat and Wu, 2007, Chan et al., 2007].

2.1.2 Low interannotator agreement

The inadequacy of disjoint representation is evident in low agreement between annotators (more often called inter-annotator agreement or ITA in the literature) when creating labeled corpora for these tasks. To ensure the quality of the labels in training data, often a minimum of two people are independently employed to label a subset of target words with the correct sense item in the inventories for such operations. A high level of ITA generally indicates one or more of the following: (1) the task is well-defined (2) people have little difficulty learning and executing the annotation guidelines (3) the label definitions are cognitively valid. Unfortunately for sense labeling tasks, it is usually the case that the degree of agreement between annotators for the same target when sense labeling are much lower than it is for other well-defined tasks such as part-of-speech tagging [Kilgarriff, 1999]. In one experiment [Furnas et al., 1987], the rate of agreement was less than 20% between annotators. In some of the more commonly used data sets, however, the ITA has ranged from 69% [Kilgarriff and Rosenzweig, 2000] to 78.6% [Landes et al., 1998] for different corpora with their own lexical inventories. The discrepancy in

ITA between the two tasks of part-of-speech tagging and word sense labeling led Wilks [2000] to say:[2]

> . . .if humans do not have this skill [to label tokens with senses] then we are wasting our time trying to automate it. I assume that fact is clear to everyone: whatever may be the case in robotics or fast arithmetic, in the NLP parts of AI there is no point modelling or training for skills that humans do not have!

Sarcasm from one of the luminaries of the task notwithstanding, it is only valid to ask whether there might be fundamental problems with such a representation if even the people tasked with annotating the data cannot agree on the labels for a substantial portion of the corpus. And if such a representational scheme constitutes the bedrock of the most dominant task in computational lexical semantics—word sense disambiguation—then it deserves even more to be questioned.

2.1.3 All-words word sense disambiguation

All-words WSD approaches, which typically disambiguate all words in a sentence at the same time and in relation to each other, usually with little or no training data, was first attempted in Cowie et al. [1992] on a small data set of 50 sentences. It has since been expanded upon and integrated as a SensEval task in 2004 [Snyder and Palmer, 2004].

Similarly, our approach can be viewed as an all-words paraphrasing model. Among the all-words WSD approaches, the model of Nastase [2008] is most closely

[2] This is sarcasm on the part of Wilks and means the opposite of what it says.

related to ours. In the model, words that are neighbors in a dependency graph mutually disambiguate each other using word sense relatedness scores determined through a heuristic. Preferred senses are computed in two passes through the dependency graph (one top-down, one bottom-up). The setting that we use allows us to use a more principled solution for inference using loopy belief propagation, in which information is passed through the graphical model until convergence. Note that we cannot use all-words WSD datasets for evaluation for our model, as they are labeled with a single best sense for each word, while our aim is to explore alternative, more flexible ways of characterizing meaning.

2.2 Usage based models of word meaning

The difficulty of doing WSD, together with these more fundamental issues, leads to the question of whether it may be useful to consider alternative computational models of word meaning that do not represent a word instance through a single best sense but instead build dynamic, context-dependent representations for each individual instance [Erk, 2010]. There have recently been several models of word meaning in context that launch off of similar motivations. Many of these models compute individual representations for each word instance as points in vector space [Kintsch, 2001, Mitchell and Lapata, 2008, Erk and Padó, 2008, Erk and Pado, 2010, Thater et al., 2010]. We will call these models *word usage models* (where a usage is a word occurrence in a particular context).

Instead of assigning each usage a single best dictionary sense, word usage models based on the distributional hypothesis compute representations that can be distinct for each usage. All existing usage models do this by representing word usages as points in vector space. The simplest such model computes the meaning of a word w in a context c (which may consist of multiple words) by summing up the vectors for w and c [Landauer and Dumais, 1997]. Kintsch [2001] computes a representation for a predicate w in the context of an argument a by determining the near neighbors of a that are most similar to w and computing their centroid. Mitchell and Lapata [2008] propose a general framework for semantic composition through vector combination that combines the vectors u, v for two constituents in a given syntactic relation and context. The models evaluated in the paper, however, disregard syntactic relation and context, and instantiate vector combination as either addition (yielding the Landauer and Dumais model) or component-wise multiplication. Mitchell and Lapata find better performance for component-wise multiplication. Erk and Padó [2008] (below **EP08**) propose a model in which a pair of syntactic neighbors mutually contextualize each other using selectional preference vectors. Take the following examples:

(2.1) The teacher addressed the undergraduate class.

(2.2) [The parliament introduced new laws]. They address class as an issue.

A verb like *address.v* in Example 2.1 is associated with a vector that describes typical direct objects of *address.v* (computed by summing over vectors of observed direct objects in a parsed corpus), and conversely a noun like *class.n* is associated with a vector that describes predicates that typically take *class.n* as an object. The usage vector for *class.n* is then computed by combining the context-independent vector for *class.n* with the direct object preference vector of *address.v*, and conversely for the contextualization of *address.v*. Erk and Padó [2009] report that using more than one syntactic neighbor for contextualization does not improve performance of this model.

2.2.1 Vector space models of word meaning

Approaches that derive vector space representations for whole phrases either explore how to encode syntactic structure [Smolensky, 1990, Grefenstette et al., 2011] or simpler structures [Mitchell and Lapata, 2008, 2010, Baroni and Zamparelli, 2010] in a vector, and how to model phrase similarity. Vector space models for larger expressions have sometimes been used as usage vector models. For example, a vector for the phrase *address class* can also be used as a vector for *address.v* in the context of *class.n*. In fact, the Mitchell and Lapata [2008] model is a phrase model, but has been used as a benchmark in the evaluation of word usage models. The model that we present in this paper derives a separate representation for each word in context, rather than a joint representation for a phrase. It is thus a word usage model, but not a model for larger expressions.

2.2.2 Graded word sense

Thater et al. [2009, 2010] (below **TFP10**) also use selectional preferences for contextualization, but they use all syntactic neighbors instead of just one. They represent each word through two vectors. The first-order vector for a word w has dimensions $\langle \textsc{rel}\ , v\rangle$ for co-occurrence of w with v in syntactic relation REL. For example, *address.v* could have a dimension ⟨OBJ, *problem.n*⟩ showing co-occurrence of *address.n* with *problem.n* as direct object. The second-order vector for a word w consists of separate subvectors for each dependency relation REL. The subvector for REL is a combination of first-order vectors of REL-neighbors of w, similar to the selectional preference vectors of EP08. To compute a usage vector for *address.v* in Example 2.1, the TFP10 model modifies the second-order vector of *address.v* by combining its SUBJ-subvector with the first-order vector for *teacher.n*, and combining its OBJ-subvector with the first-order vector for *class.n*. This is the model if w is a verb or noun. For adjectives and adverbs, the model computes the usage vector for the headword of w in the dependency graph as the usage vector of w. This step improves performance, but having the meaning of a word be the meaning of its headword is hard to interpret. Also note that while the Thater et al. model uses all syntactic neighbors for contextualizing a word w, these neighbors act on independent sub-vectors of w rather than on a common structure. Erk and Pado [2010] (below **EP10**) argue that the whole sentence context, rather than just local syntactic context, should be used to contextualize a word. However, their model represents a sentence as a bag of words, ignoring syntax.

Like these models, our model computes an individual representation for each usage. In contrast to usage vector models, we represent meaning in context through a distribution over paraphrases. Among the models discussed above, the ones most closely related to our model are EP10 and TFP10, which both use selectional preferences for contextualization, as syntactic neighborhood is the main source that our model uses for inference. In contrast to EP10 and TFP10, we aim to provide a general, uniform mechanism for inference that uses as knowledge sources all direct syntactic neighbors, nodes at greater distance in the dependency graph, and document context. EP08 only consider a single, selected syntactic neighbor. TFP10 use all direct syntactic neighbors, but have them modify separate subvectors rather than act on a common structure. They also employ different representations depending on the part of speech of the word to be contextualized. None of them use nonlocal syntactic context.

2.2.2.1 Lexical substitution

McCarthy and Navigli [2009] proposed representing word usages through weighted paraphrases (see Figure 3.1). In the Lexical Substitution (below, **LexSub**) dataset that they introduced for the 2007 SemEval task, each paraphrase is weighted by the number of annotators who proposed it.[3] Participants had to perform two tasks: determining paraphrase candidates for each target, and ranking candidates for each usage. Participating systems mostly collected paraphrase candidates from manually created resources, mainly WordNet [Fellbaum, 1998]. The most common methods for ranking candidates (e.g., [Giuliano et al., 2007, Hassan et al., 2007, Yuret, 2007]) were to substitute the candidate for the target in the given sentence context and to search for the resulting phrase in an n-gram corpus [Brants and Franz, 2006], or to use a language model. The LexSub dataset focuses on paraphrases for single words. In contrast, approaches to learning paraphrases from text usually consider both single-word and multi-word paraphrases (e.g., Bannard and Callison-Burch [2005], Barzilay and McKeown [2001]).

[3] Annotators could also generate more than one paraphrase per item.

Approaches to learning inference rules from text consider not only (single- and multi-word) paraphrases but also other types of rules, such as *enablement* (fight → win) and *happens-before* (buy → own) [Lin and Pantel, 2001, Chklovski and Pantel, 2004, Szpektor and Dagan, 2008, Berant et al., 2010]. A related task is to determine the applicability of an inference rule in a given sentential context [Pantel et al., 2007, Szpektor et al., 2008, Poon and Domingos, 2009, Ritter et al., 2010]. Approaches to this problem use similarity in selectional preferences as well as similarity in sentence context to determine whether an inference rule applies in a given context.

2.2.2.2 A probabilistic digression on LexSub

There is a very interesting frequentist undercurrent to how both graded word sense and the lexical substitution task is defined. The motivations are implicitly frequentist in terms of how word senses are defined. The weight associated with each paraphrase for a given target is the number of people who have proposed that paraphrase. Given enough people, the definition of the weight associated with each paraphrase in the English Lexical Substitution task corresponds to "frequencies of outcomes in random experiments" [MacKay, 2003] or in the case of LexSub the frequencies at which each paraphrase has been proposed.

In this situation, it is more satisfying to state that the meaning of a word is the ratio over the aggregate counts of the paraphrases that are proposed for the word by the entire speech community. Yet, it would be interesting for no other reason than satisfying curiosity whether a strongly Bayesian approach as is implicit in our model can derive or match results that are frequentist in motivation.

Chapter 3

Model

We define word meaning in context to be a probability mass function over a set of paraphrases. This definition further defines the modeling and inferential framework to be used for inferring the meanings of words in the face of evidence: probabilistic graphical models. This framework determines how the evidence is selected and transformed and how inference is conducted over this evidence. The paraphrases are defined to be the values for the random variable that represent meaning—we will call this random variable the **paraphrase node**—and the probability mass function—we will call this the **paraphrase distribution**—over this paraphrase node in relation to any relevant evidence is taken to represent what a word means. By allowing the surrounding evidence to determine the mass function, we take this meaning to be dynamic and be influenced by its surrounding context. It therefore falls under a more general **word usage model** where individual instances of words in context and their derived representations take center stage rather than a predefined sense inventory. With such a definition, we gain several advantages in that everything in the model, from representation to learning to inference can be dealt with in the unified framework of probabilistic graphical models [Jordan, 2004].

In this chapter, we first discuss what it means to represent word meaning as a probability mass function in context—more accurately a conditionally defined probability mass function. We then provide a brief overview of probabilistic graphical models as it relates to our model. We next describe the various sources of evidence that are available and how they may be incorporated into the study. We then describe the value space of our paraphrases and the definition of our parameters.

3.1 Representing word meaning: Word meaning as probability mass function

As mentioned in the previous Chapters 1 and 2, our models will learn how to represent words in context as probability mass functions, or paraphrase distributions over paraphrase nodes, in context. We will then build concrete evidence structures and inference procedures around it.

As a concrete example, consider the following sentences where the word we're interested in is *brightest*:[1]

(3.1) In fact, during at least six distinct periods in Army history since World War I, lack of trust and confidence in senior leaders caused the so-called best and *brightest* to leave the Army in droves.

(3.2) An evening of classical symphonic music, played by the next generation stars in the American orchestral scene, can be savored at the New World Symphony, a special Miami institution that nurtures the best and *brightest* young symphonic musicians.

[1] These are sentences 5 and 6, respectively, from the English Lexical Substitution task at SemEval-2007[McCarthy and Navigli, 2007, 2009]

An important goal of our model is to resolve the meaning of the word *brightest* within the contexts that it appears in above and **represent the resolved meaning as a probability distribution**. More specifically, we represent the meaning of a given word as a conditional probability distribution that is dependent on context. The probability distribution over the meaning of the word *brightest*, whose meaning we will associate with the random variable m, is conditioned on the entire sentence $\mathbf{s}$ (or possibly some other context) in which it appears. In other words, we want to calculate the conditional probability mass function:

$$P(m|\mathbf{s})$$

We refer to the value of the context provided by sentence (3.1) as s_1 and the context provided by sentence (3.2) as s_2. Then our goal is to derive or infer the mass functions

$$P(m|\mathbf{s} = s_1)$$

and

$$P(m|\mathbf{s} = s_2)$$

respectively and **we take each of the functions themselves as representing the meaning of *brightest*** in each respective context. Also, to emphasize, the random variable m is associated with the meaning of *brightest* and is therefore hidden; it is not an observed random variable that is associated with *brightest* itself. However, we note that, for the core variant of our model, the hidden variables are not nameless indexes such as can usually be found in unsupervised models of part-of-speech tagging [Moon et al., 2010] or document topic modeling [Blei et al., 2003].

The **values of the hidden nodes are meaningful** and as such will relate to the surface statistics of observations in greater degree than is usual for a model that posits hidden nodes.

An important question for fleshing out these functions then is to define the range of values that m can assume. A first solution is to posit that the entire finite vocabulary for English according to some lexical resource constitutes the range of values for m. Listing each of the possible values that m can take on in alphabetical order, from the first word to the last word,[2] this would be:

$$P(m = \mathtt{a}|\mathsf{s} = s_1) = 0$$

$$\dots$$

$$P(m = \mathtt{zymosan}|\mathsf{s} = s_1) = 0$$

for *brightest* in the context of sentence (3.1) and where the ellipsis stands in for every word in between `a` and `zymosan`. To be thorough, we also show the probability mass function for the other sentence:

$$P(m = \mathtt{a}|\mathsf{s} = s_2) = 0$$

$$\dots$$

$$P(m = \mathtt{zymosan}|\mathsf{s} = s_2) = 0$$

for *brightest* in the context of sentence (3.2).

[2] These are the first and last words according to the online Merriam-Webster dictionary (`www.merriam-webster.com`). Also, *zymosan* is defined as "an insoluble largely polysaccharide fraction of yeast cell walls"

That is, the meaning of *brightest* within the context of some sentence is not represented by the probability of $P(m = \mathrm{a}|\mathbf{s})$ or by $P(m = \mathrm{zymosan}|\mathbf{s})$ but by the entire probability mass function itself whose domain stretches from $m = \texttt{a}$ to $m = \texttt{zymosan}$. Obviously, not all of the values of m have a probability mass of zero, but we assume that the vast majority of them do. Furthermore, taking a vaguely Bayesian stance, it seems natural to assume that no probability mass is allotted to either `a` or `zymosan` to represent the meaning of *brightest* in context.

Ignoring the majority with zero probability mass, we say that the meaning of the word *brightest* in the context of sentence (3.1) is

$$\begin{aligned}
P(m = \texttt{capable}|\mathbf{s} = s_1) &= 0.11 \\
P(m = \texttt{clever}|\mathbf{s} = s_1) &= 0.22 \\
P(m = \texttt{intelligent}|\mathbf{s} = s_1) &= 0.33 \\
P(m = \texttt{motivated}|\mathbf{s} = s_1) &= 0.11 \\
P(m = \texttt{promising}|\mathbf{s} = s_1) &= 0.11 \\
P(m = \texttt{sharp}|\mathbf{s} = s_1) &= 0.11
\end{aligned}$$

Any value of m that has not been listed above is defined to have probability mass zero.[3]

Similarly for *brightest* in sentence (3.2), the values with non-zero probability

[3] These numbers or weights are taken from the gold data of the English Lexical Substitution task [McCarthy and Navigli, 2009]. The weights are non-negative integer counts in the gold and we have normalized them here to sum to one.

are

$$P(m = \texttt{gifted}|\mathtt{s} = s_2) = 0.14$$

$$P(m = \texttt{promising}|\mathtt{s} = s_2) = 0.14$$

$$P(m = \texttt{skilled}|\mathtt{s} = s_2) = 0.14$$

$$P(m = \texttt{talented}|\mathtt{s} = s_2) = 0.43$$

$$P(m = \texttt{up-and-coming}|\mathtt{s} = s_2) = 0.14$$

To facilitate comparison, we present the mass functions again, side by side:

m	$\mathtt{s}$=sent.(3.1)	$\mathtt{s}$=sent.(3.2)
capable	0.11	0
clever	0.22	0
gifted	0	0.14
intelligent	0.33	0
motivated	0.11	0
promising	0.11	0.14
sharp	0.11	0
skilled	0	0.14
talented	0	0.43
up-and-coming	0	0.14

where the first column on the left lists some possible values of m and the second and third columns list the probability masses of the values in the context of sentences (3.1) and (3.2), respectively. The only value of m where there is any overlap in terms of both distributions having non-zero values is `promising`. This reflects that both usages of *brightest* are related semantically and are not disjoint in meaning.

There are several interpretations that can be given to the above functions. One possible interpretation is that, in the context of sentence (3.1), 33% of the meaning of *brightest* is captured by `intelligent`, 22% by `clever`, 11% by `capable` and so forth. Similarly, 43% of the meaning of *brightest* is captured by `talented`, 14% by `gifted` and so forth in the context of sentence (3.2). This can be reworded so that the above probability values represent soft cluster membership. That is, there are clusters that are labeled with words such `intelligent` and `zymosan` and *brightest* in the context of sentence (3.1) belongs 33% to the `intelligent` cluster and 0% to the `zymosan` cluster and so forth. Equivalently, it can be restated as a mixture model.

The above functions also tell us that most other words—the words that are in our vocabulary but are not listed above because they had zero probability—do not represent the meaning of *brightest* in these contexts in any way. Again, from a vaguely Bayesian viewpoint, it seems plausible that $P(m = \texttt{zymosan}|\mathbf{s}) = 0$ when $\mathbf{s} = s_1$ or $\mathbf{s} = s_2$ or even in virtually any other context that the word *brightest* can occur in.

What about words which can capture the meaning of *brightest* but had zero probability in the above examples? What about words such as *luminous* or *shiny* that are related in meaning to *bright* in the right context? Consider the following example where again the word of interest is *bright*:[4][5]

(3.3) The roses have grown out of control, wild and carefree, their *bright* blooming faces turned to bathe in the early autumn sun.

[4] Our model will treat sets of words such as *bright* and *brightest* as belonging to the same type or as instances of the same lemma

[5] This is sentence 3 from the English Lexical Substitution task [McCarthy and Navigli, 2009]

We will refer to this sentence as s_3. Here we say that the meaning representation of *bright* in the above sentence as a probability mass function is:[6]

$$P(m = \texttt{brilliant}|\mathbf{s} = s_3) = 0.2$$

$$P(m = \texttt{colorful}|\mathbf{s} = s_3) = 0.4$$

$$P(m = \texttt{gleam}|\mathbf{s} = s_3) = 0.2$$

$$P(m = \texttt{luminous}|\mathbf{s} = s_3) = 0.2$$

In this case there is no overlap with the previous meaning representations of *bright/brightest*:

m	**s**=sent.(3.1)	**s**=sent.(3.2)	**s**=sent.(3.3)
brilliant	0	0	0.2
colorful	0	0	0.4
gleam	0	0	0.2
luminous	0	0	0.2
capable	0.11	0	0
clever	0.22	0	0
gifted	0	0.14	0
intelligent	0.33	0	0
motivated	0.11	0	0
promising	0.11	0.14	0
sharp	0.11	0	0
skilled	0	0.14	0
talented	0	0.43	0
up-and-coming	0	0.14	0

Given the three instances of *bright* in sentences (3.1) $\sim$ (3.3), it seems plausible to say that the usages in sentences (3.1) and (3.2) are more closely related to

[6] Again, these weights are taken from the gold data of the English Lexical Substitution task [McCarthy and Navigli, 2009]

each other while the usage of *bright* in sentence (3.3) is quite distinct from the previous two. The argument for greater similarity is based purely on the fact that there is at least one value of m, namely m=`promising`, where $P(m|\mathbf{s}{=}s_1) \cdot P(m|\mathbf{s}{=}s_2){\neq}0$ whereas such a value doesn't exist for either $P(m|\mathbf{s}{=}s_1) \cdot P(m|\mathbf{s}{=}s_3){\neq}0$ or $P(m|\mathbf{s}{=}s_2) \cdot P(m|\mathbf{s}{=}s_3){\neq}0$. Or we could use an established measure such as Jensen-Shannon divergence or Kullback-Leibler divergence, but the presence of many zeros in the distributions involved means a tweak would be required for either to work. Though we, as humans, might intuitively grasp that one of these is not like the others, the goal of our model is to capture such intuitions by dint of numbers only.

Such a representational structure lends itself to probabilistic graphical models wherein we can manipulate not just the output of an inference but lay down the scaffolding upon which we conduct parameter learning. Before we can talk about the structure of the evidence and how information can flow, we first discuss probabilistic graphical models in general.

3.2 Probabilistic graphical models

When modeling complex stochastic phenomena, there will be differing degrees of interaction or dependence between the some of the subsets of the random variables involved. For various reasons, it is often necessary to posit that certain subsets of random variables are independent of other subsets, which lead to different factorizations over the same set of random variables. The reasons are diverse and always results in some kind of simplification of the models. The reason could be a practical issue such as computational tractability so that any calculations terminate within a reasonable period. The reason could be a formal or representational issue that has to do with the coherence and comprehensibility of the model so that humans can easily understand important interactions and correlations—or at least assumptions of such correlations—between variables. The reason could simply be that certain independence assumptions are justified by investigating and statistically testing levels of interaction between independent variables and deriving a valid model based on such exploratory analysis. In many cases, even though this last approach (where dependence and independence assumptions derive from exploratory statistical analysis) is the most valid and justified, independence assumptions must be made for the reason that the computational challenges in terms of time or space are too great.

Simplifying such concerns, such a probabilistic model can be represented through a graph where the random variables constitute the nodes or vertices, edges between pairs of nodes model statistical dependencies between such nodes, and lack thereof between pairs of nodes reflect independence assumptions or knowledge of statistical independence between such nodes. Such a composition of a probabilistic model and its representation as a graph is called a probabilistic graphical model. One can build a visual representation of the statistical dependencies and independence assumptions between random variables of interest and this facilitates understanding of the model on a global level with a larger view.

With the definition of our current model as a probabilistic graphical model, we gain benefits of standardized procedures for inference and learning. Inference refers to the process of reaching conclusions given the structure of the graph. Learning refers to the process wherein we learn the parameters necessary for conducting inference.

3.2.1 Directed versus undirected graphical models

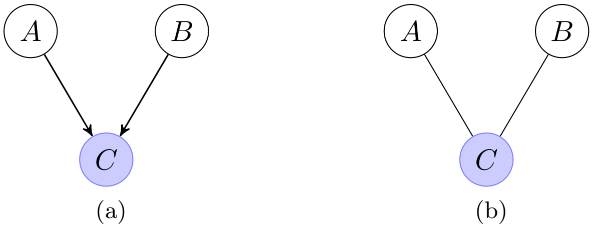


Figure 3.1: Examples of graphical models. (a) is a directed graph. (b) is an undirected graph. The only difference graphically is that the former has arrows for edges and the latter has unadorned edges.

There are two basic types of probabilistic graphical models: directed graphical models and undirected graphical models. Directed models represent assumptions of causality between relevant random variables whereas undirected models represent assumptions of dependence or correlation without implying causality between the variables. Regardless of the distinction, the shape of a graph over a set of random variables determines the factorization over some probability density or mass function associated with these random variables.

Formally, we define a graph $G = (V, E)$ where $V = \{x_1, \ldots, x_n\}$ is the set of nodes which correspond to random variables and $E \subset V \times V$ is the set of edges. If the graph is undirected, then if $(x_i, x_j) \in E$ where $x_i, x_j \in V$, then (x_j, x_i) refers to the same edge. If the graph is directed, then edges are properly treated as ordered pairs. Here and in what follows, when discussing directed graphs—including dependency parses and its transformations—we will maintain the convention that in the pair (x_i, x_j) the parent is on the left and the child is on the right, or $x_i \rightarrow x_j$.

In directed graphs, which are also called Bayesian networks, for a given node x_i, its parent nodes $\pi(x_i) \subset V$ are defined to be $\pi(x_i) = \{x_j \in V : (x_j, x_i) \in E\}$. Following convention, we define a variable $\mathbf{x} = V$ as the set of random variables that is equivalent to V, but for use within distribution functions. In a directed graph G, the distribution $p(\mathbf{x})$ factorizes as follows:

$$p(\mathbf{x}) = \prod_{x_i \in V} p(x_i | \pi(x_i)) \tag{3.4}$$

For now, we also take the liberty of defining functions by their arguments—e.g. $p(x_i)$ and $p(x_j)$ are distinct functions when $x_i \neq x_j$—and dispense with devising separate indexes for functions. With this general definition, the directed graphical model defined in Figure 3.1a factorizes to:

$$p(A, B, C) = p(C|A, B)p(A)p(B) \tag{3.5}$$

It is important to note that the factorization of a distribution also defines in practice the functions—near equivalently, the parameters that define the functions—that are expected to exist *a priori* or are expected to be learned. In the example of Figure 3.1a and its factorization above, the functions $p(A), p(B), p(C|A, B)$ are the basic building blocks from which other functions such as marginal distributions or conditional distributions are derived.

In the case of undirected models, which are also called Markov random fields (MRF), the factorization is similarly defined over nodes that are connected but without the distinction of parents or children. However, we no longer call the functions associated with groups of random variables probability mass/density functions. Instead they are called potential functions, which we denote with F. These potential functions are generally defined over the maximal cliques in the graph [Wainwright and Jordan, 2008], where a maximal clique C is a fully connected subset of V such that $(x_i, x_j) \in E$ for all $x_i, x_j \in C$ and, for all $x_k \in V \backslash C$, there is some $x_i \in C$ such that $(x_i, x_k) \notin E$. Calling the set of maximal cliques $C \in \mathcal{C}$, we can factorize $p(\mathbf{x})$ as:

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{C \in \mathcal{C}} f(x_C) \tag{3.6}$$

where Z is a suitable normalization constant such that $\sum_{\mathbf{x}} p(\mathbf{x}) = 1$. Following convention, we define a separate variable for C when referencing them within potential functions. We define x_C to be the set of random variables equivalent to C.

To illustrate this with the toy graph Figure 3.1b, the distribution factorizes into:

$$p(A, B, C) = \frac{1}{Z} f(A, C) f(B, C) \tag{3.7}$$

Again, the functions $f(A, C)$ and $f(B, C)$ form the basic building blocks for subsequent derivations and no further decomposition is defined for either $f(A, C)$ or $f(B, C)$.

As can be seen, one immediate distinction between the factorizations over directed models and undirected models is that one uses conditional distributions and the other doesn't. While this isn't a categorical distinction, it is common in practice. There is another important distinction between Bayesian networks and MRFs. This is that the product terms in the factorization of the latter, which are more commonly called potential functions, are not subject to the same summation constraints as the former. Whereas product terms in Bayesian networks need to be probability distributions that sum to one, the only constraint on individual potential functions is that they be non-negative and bounded.

One reason for the distinction between directed and undirected models is in conducting inverse inference or learning given observations. If we have little knowledge of the parameters involved in the models except perhaps the parametric form and we have observed the realizations of certain random variables, we can hold the random variables that have been observed fixed and make more informed guesses as to the random variables that have not been observed. Or if the parameters are known, conditioning on certain variables will reduce the uncertainty involved in the remaining variables [Cover and Thomas, 2005].

The difference between directed models and undirected models comes into play when attempting to factorize these conditional distributions. We can show how this is so for the simple examples in Figure 3.1, where the observed variable B which is also the conditioning variable is filled in. For the directed graph of Figure 3.1a, the factorization over its unconditioned distribution is defined as in eq. (3.4). But if conditioned on C:

$$p(A, B|C) \neq p(A|C)p(B|C)$$

in general.

In contrast, the following does hold for the undirected graph in 3.1b: [7]

$$p(A,B|C) = f(A|C)f(B|C)$$

where

$$f(A|C) = \frac{f(A,C)}{\sum_A f(A,C)}$$

and

$$f(B|C) = \frac{f(B,C)}{\sum_B f(B,C)}$$

and the normalization constant Z naturally cancels.

Another important point that will influence how calculations are conducted in our (to be defined) undirected model is the parameter tying that occurs when variables such as C are marginalized out. In the case of Figure 3.1b, assume we are marginalizing out the variable C. The operation of marginalizing over C refers to summing over the values of C such that a new function without C is derived:

$$p(A,B) = \sum_C p(A,B,C) = \sum_C \frac{1}{Z} f(A,C)f(B,C)$$

The difference between the directed and undirected version of the three node graphs in Figure 3.1 is that the factorization of

$$p(A,B) = p(A)p(B)$$

[7] For a quick proof: $p(A,B|C) = \frac{P(A,B,C)}{\sum_{A,B} P(A,B,C)} = \frac{f(A,C)f(B,C)}{\sum_{A,B} f(A,C)f(B,C)} = \frac{f(A,C)f(B,C)}{(\sum_A f(A,C))(\sum_B f(B,C))} = f(A|C)f(B|C)$. Note that this generalizes even to cases where A, B, C are sets of variables rather than single variables.

is possible for the directed graph. This is possible since $\sum_C p(C|A,B) = 1$ and therefore

$$p(A,B) = \sum_C p(A,B,C) = \sum_C p(C|A,B)p(A)p(B) = p(A)p(B)$$

In comparison, it is usually the case that

$$p(A,B) \neq p(A)p(B)$$

where

$$p(A,B) = \sum_C p(A,B,C) = \sum_C \frac{1}{Z} f(A,C)f(B,C)$$

for undirected models.

For directed models, the derivation of dependence and independence between sets of random variables due to marginalization or conditioning in a directed graphical model is a little more complicated than is implied by the simple example in Figure 3.1a. Since Bayesian networks are immaterial to our model of word meaning, we refer the reader to [Bishop, 2006, Chap. 8] for an in-depth discussion. In contrast, our discussion of 3.1b above draws a complete picture of what occurs when marginalizing or conditioning over sets of variables in an undirected graph. Given three maximal cliques C_1, C_2, C_3 and if all paths between C_1 and C_3 go through C_2, then it will hold that (1) C_1 and C_3 are independent of each other conditioned on C_2 and (2) C_1 and C_3 are not independent of each other in general if C_2 is marginalized out.

3.2.2 Undirected graphs and factor graphs

In discussing undirected graphical models in the previous section, we stated that it's "generally" the case that potential functions are defined over maximal cliques. This is because the correspondence between factorizations into potential functions and a graphical model is not one to one. Consider the example of 3.2a.

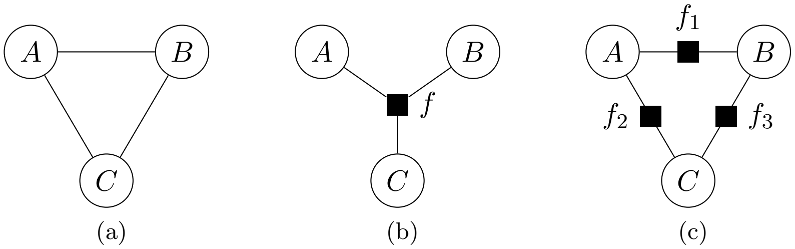


Figure 3.2: Left: a simple undirected graphical model with three nodes A, B, C. Right: Two possible factor graphs (out of many) for this undirected graphical model.

There might be situations where it is more advantageous to posit factorizations at the pairwise level such that

$$p(A, B, C) = f(A, B)f(B, C)f(C, A) \tag{3.8}$$

However, under general practice, $p(A, B, C)$ should be defined such that no factorizations are possible. And so the graph in Figure 3.2a is ambiguous between the factorization of eq. (3.8) and no factorization at all.

An alternative formalism that removes such ambiguities is that of factor graphs [Kschischang et al., 2001]. The computational machinery underlying factor graphs in terms of incorporating evidence, learning from the evidence and reaching conclusions based on this evidence is no different from what goes on with most directed and undirected graphical models. Instead, factor graphs are a more explicit formalism that require each and every product term in a factorization to be represented as individual nodes in the graphical presentation.

Formally, a factor graph is a bipartite graph with two types of nodes: factor nodes—represented as filled square nodes—and variable nodes—represented as empty round nodes. As a bipartite graph, edges exist only between factor nodes and variable nodes. Edges between factor nodes and factor nodes or between variable nodes and variable nodes are illegal within the framework. Finally, edges are established between a factor node and one or more variable nodes if and only if the variables are arguments of the factor node.

Taking the example of Figure 3.2a again, there are at least two possible factorizations. The first is when

$$p(A, B, C) = f(A, B, C) \tag{3.9}$$

where no subsequent factorization is possible over $p(A, B, C)$. The second is when

$$p(A, B, C) = f(A, B)f(B, C)f(C, A) \tag{3.10}$$

No matter which factorization represents the underlying model, the formalism of Markov random fields allows only the representation in 3.2a.

When we use the alternative formalism of factor graphs, we are able to distinguish between eq. (3.9) and eq. (3.10) in our graphical representation. Since the former has only one product term (or factor) involved, this factor is mixed into the existing set of variable nodes as a new factor node. Then edges are built between the arguments of the factor—which is all three of A, B, C—and the factor node. This becomes Figure 3.2b where the factor nodes are represented with filled square nodes. With eq. (3.10), in contrast, there are three factors, each of which takes a different set of arguments. As such, we give each of the factors different labels—f_1, f_2, f_3 for convenience—and connect each factor with its respective arguments. This is represented in Figure 3.2c.

Furthermore, we can use factor graphs to remove notational clutter. For example, if we know that some random variables are observed variables and therefore act as constants in terms of an implementation, we can remove such random variables and integrate them into the factors themselves with the understanding that such variables are held fixed within any calculations involved.

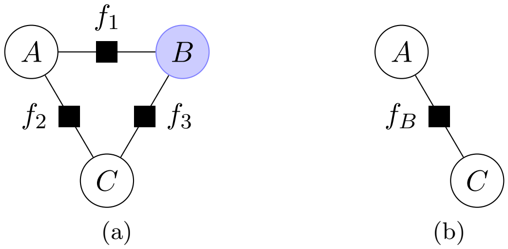


Figure 3.3: Conversion of a factor graph with explicit factors and with observed variable arguments to a smaller graph

Consider the model in Figure 3.3a where B is an observed variable. Since we know that the value of B is fixed to, say, b, all calculations over eq. (3.10) actually only vary over two variables, A and C. We therefore create a new factor that incorporates the following

$$p(A, B = b, C) = f_1(A, B = b) f_2(A, C), f_3(B = b, C)$$

into a new function

$$f_B(A,C) \triangleq p(A,B=b,C) = f_1(A,B=b)f_2(A,C), f_3(B=b,C)$$

and we obtain a simpler factor graph in Figure 3.3b which will help us remove some clutter later on where more variables are involved or the model is presented in an abstract manner.

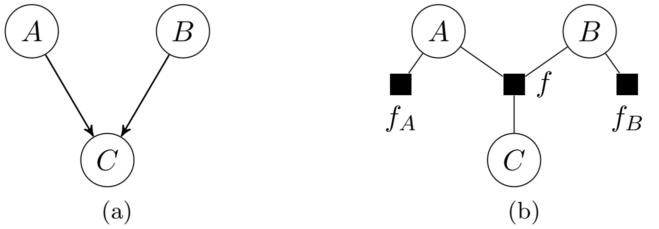


Figure 3.4: Conversion of directed graphical model to factor graph

As a final note, we show how a directed graph may be converted to a factor graph. We consider again the simple directed graph in Figure 3.1a and its factorization in eq. (3.5):

$$p(A,B,C) = p(C|A,B)p(A)p(B)$$

The graphical representation dictates that $p(A,B,C)$ be factorized into the three terms of $p(C|A,B), p(A)$ and $p(B)$. One part of the conversion to factor graphs is straightforward since all that is required is that each of the product terms should be represented through its own factor node and that each factor node should be connected to its argument. We show the conversion of the directed model to a factor graph in the two graphs of Figure 3.4 where eq. (3.5) is now

$$p(A,B,C) = f(A,B,C)f_A(A)f_B(B)$$

It is obvious that there is some amount of information loss when moving from directed graphs to factor graphs since what was explicitly a conditional probability distribution (i.e. $p(C|A,B)$) is now a more general function which need not necessarily follow the summation constraint of probability distributions which must sum to one.

With these preliminaries in place, we are now ready to discuss inference over graphs, i.e. the process of making informed decisions with probabilistic graphical models. Another way in which this distinction between directed and undirected models is critical when conducting inference over large sets of random variables such as in our model. We discuss this in the following section.

3.2.3 Inference, belief propagation and loopy belief propagation

Once a graphical model has been formulated, there are standardized procedures of conducting calculations over the nodes involved. In many cases, the goal of the calculations is to conduct inference. Given that there are certain nodes that are considered to be evidence and other nodes that can be considered to capture the information emerging from the evidence to which they are connected, inference is a general catch-all term for methods that either process or summarize that information. Within these inference methods, our specific interest is in marginal inference.

More formally, the goal is to conduct marginal inference, which can be posed as follows. Given a set of random variables $\mathbf{m}$ and some measure of its information $P_{\mathbf{s}}(\mathbf{m})$ in relation to another set of random variables $\mathbf{s}$, what do we know about a specific random variable $m_i \in \mathbf{m}$ solely in relation to $\mathbf{s}$? The answer to this question is obtained by marginalizing out (i.e. summing over) the random variables that are not m_i:

$$P_{\mathbf{s}}(m_i) = \sum_{\mathbf{m} \setminus \{m_i\}} P_{\mathbf{s}}(\mathbf{m}) \tag{3.11}$$

The above is often formulated by placing the evidentiary random variables $\mathbf{s}$ as arguments to the functions involved—e.g. $P(\mathbf{m}, \mathbf{s})$—but we will maintain the convention of incorporating evidence as subscript to reduce the length of notation involved and simplify some notation.

The left hand side of eq. (3.11) is known as the marginal of m_i and such functions as a whole in relation to $P_{\mathbf{s}}(\mathbf{m})$ (called the **global function**) are referred to as **marginals**. In this most general form, this is an intractable problem, but if some random variables are independent of each other or are assumed to be independent, the ordering of the summation can be carefully arranged so as to make this pliable. This is a result of the very simple fact that multiplication is distributive over addition [Aji and McEliece, 2000]. Graphical representations of these independence assumptions such as factor graphs help formalize and visualize the models that derive from them.

There are standardized procedures for conducting exact marginal inference such as the sum-product algorithm [Kschischang et al., 2001] or belief propagation [Yedidia et al., 2001]. The caveat is that such procedures are guaranteed to be exact only for graphs without loops such as those in Figure 3.1. Such a guarantee cannot be made for arbitrary graphs including some of the graphs built from dependency trees that we use in our model because of the presence of loops. In such situations, an alternative approach called the junction tree algorithm exists which permits exact inference in spite of the presence of loops [Cozman, 2000]. However, an internal step in the application of this algorithm—namely, the ordering of variables to be eliminated for message passing—is known to be an NP-hard problem for arbitrary graphs [Beal, 2003]. Therefore, to conduct inference within our models, we use loopy belief propagation [Murphy et al., 1999] which is a modified application of the general belief propagation algorithm in spite of the presence of loops. This has been shown to work well in practice [Murphy et al., 1999].

3.2.4 Graphical models in computational linguistics

In computational linguistics, undirected graphical models have mainly been used in the shape of conditional random fields [Lafferty et al., 2001, Sutton et al., 2007] and Markov Logic Networks [Riedel and Meza-Ruiz, 2008, Yoshikawa et al., 2009, Poon and Domingos, 2009]. Such models have also occasionally been used for structural tasks such as morphology-based word generation [Dreyer and Eisner, 2009], noun-phrase chunking [Sutton et al., 2007], and dependency parsing [Smith and Eisner, 2008]. Directed graphical models have seen much more use in computational linguistics (e.g. topic models for semantics or HMMs for low-level syntax). In the context of modeling word meaning, Brody and Lapata [2009] use topic models for sense induction. They rely mainly on context word and word n-gram features, finding dependency features to be very sparse. Deschacht and Moens [2009] define a language model as a Hidden Markov Model in which observed words are generated by hidden variables ranging over the whole vocabulary. We cannot directly compare to either of those models: The Brody and Lapata model cannot be mapped to paraphrases in any straightforward way, and the Deschacht and Moens model does not constrain the hidden word to be a paraphrase. We will evaluate a variant of the Deschacht and Moens model that only considers paraphrases (below called the sequential model).

3.3 Probabilistic modeling of graded word sense

The model that we introduce is a *usage model* of word meaning, where each word representation is vector valued, context dependent and inferred dynamically. Such a model contrasts with models in WSD which assume a fixed, discrete sense inventory and are dominant in practice. While existing usage models represented a target word in context through a vector of contextual co-occurrence dimensions, we use a distribution over potential paraphrases of the target. In our case, the task of computing a usage representation is defined as a probabilistic inference task over graphs. We examine several probabilistic models to investigate how different knowledge sources and graph topologies affect predicted word meaning.

Though all the models we investigate have slightly different graphs, they share a common foundational node-node pair. As the building block for all our models, we construct two adjacent nodes for each content word of the sentence: one node (the *observed* node) represents the surface form of the word itself and the other node (the *hidden* node) represents its usage meaning. We call the distribution inferred over the hidden node given the evidence from its observed context the *paraphrase distribution* of the observed word. This observed context may include evidence as diverse as the word's heads and dependencies, its left and right context, the entire sentence, or even the entire document. The integration of such knowledge sources is accomplished in a standard way by summing over all hidden variables and multiplying the corresponding factors. This process of integration is inference, and contextualization is the inference of paraphrase distributions.

To briefly recap the previous sections, we will be using factor graphs to represent our models. A factor graph is a bipartite graph over two types of nodes, nodes that correspond to variables and nodes that correspond to *factors*. A factor is a function whose arguments are the variable nodes adjacent to the factor node. The factor graph as a whole represents the product of all the factors in it.

3.3.1 Evidence and graph transformations

Depending on the types of evidence that we want to use for contextualization, we use different topologies in the graph over which we conduct inference. The first piece of evidence that we consider is the sentence.

3.3.1.1 The sentence as evidence: Sequential order

When considering a sentence as the basic frame of evidence, the simplest option is to take the surface left-to-right order of the sentence as given. Furthermore, if we make the assumption that all information is strictly local, i.e. the meaning of a word is only influenced by the words immediately to the left and right of it, we have even simpler graphs which are star shaped graphs centered on the paraphrase node of interest.

Illustrating with the sentence of Example 1.1

the box was in the pen

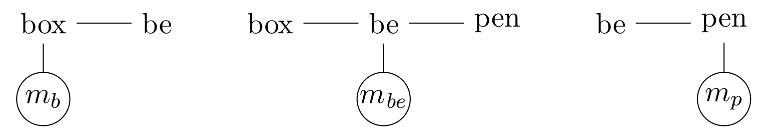
we tag the words for part-of-speech and lemmatize the tokens:

the.DT box.NN be.VB in.IN the.DT pen.NN

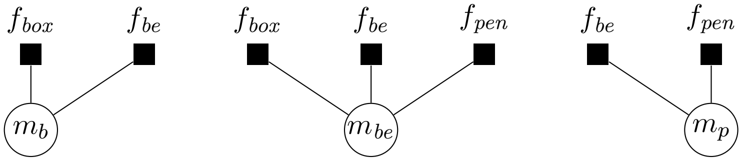
We make the assumption that function words have no influence on paraphrase distributions and so remove them from consideration:

box.NN be.VB pen.NN

We do not consider non-auxiliary verbs such as `be` or `do` to be function words and so retain them. The same applies to pronouns such as `she` or `I`. Then we duplicate the nodes such that star shaped graphs centered on a content word of interest can be created (we leave the POS-tags out to remove clutter) and we add the paraphrase nodes attached to each content word of interest:



Then, because we want adjacent words to directly affect the paraphrase nodes, we move the edges so that the adjacent words are directly connected to the paraphrase nodes:



And the paraphrase distribution for each paraphrase node is computed as:

$$p(m_b) \propto f_{box}(m_b)\ f_{pen}(m_b)$$

$$p(m_{be}) \propto f_{box}(m_{be})\ f_{be}(m_{be})\ f_{pen}(m_{be})$$

$$p(m_p) \propto f_{be}(m_p)\ f_{pen}(m_p)$$

All that is involved in calculating the paraphrase distribution of, say, the paraphrase node of *be* (m_{be}) is multiplying each of the three product terms that model the influence that each of the adjacent observations have on the possible paraphrases for *be*: $f_{box}(m_{be})$, $f_{be}(m_{be})$, and $f_{pen}(m_{be})$. This reflects our assumption for this variant of our model that information is strictly local.

More generally, for a given string of space delimited tokens $(w_1, \ldots, w_n)$—which have been suitably POS-tagged, lemmatized and stripped of function words—we create a coindexed string of paraphrase nodes $(m_1, \ldots, m_n)$ and graph components centered on each of those paraphrase nodes that only include the observation immediately to the left (if it exists) and to the right (if it exists) within a sentence boundary. Therefore, for some paraphrase node m_i, the only nodes of the component centered on this node are m_i, the factor node to the left f^l_{i-1} (if it exists), the coindexed factor f_i, and the factor node to the right f^r_{i+1} (if it exists). The superscripts ^{l} and ^{r} are to distinguish left and right factors since $f^l_{i-1}(m_i)$ and $f^r_i(m_{i-1})$

are different functions. Then for the paraphrase node m_i, its paraphrase distribution is defined to be:

$$p(m_i) \propto f^l_{i-1}(m_i)\ f_i(m_i)\ f^r_{i+1}(m_i)$$

If either the left or right observations don't exist because the word happens to occur at a sentence boundary, then the corresponding left or right factor should be removed.

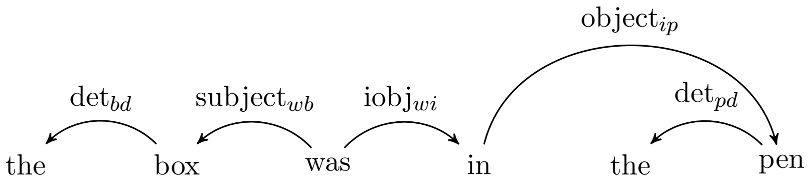
3.3.1.2 The sentence as evidence: Dependency parses

The model variant in the previous section reflected the reductive assumption that meaning is local and is determined by left-to-right surface order. One can make a more informed assumption, instead, and assume that meaning is determined by syntactic structure rather than left-to-right surface order. For the moment, we still hold on to the assumption that meaning is local.

There are many different types of syntactic formalisms with attendant structures but we will use dependency parses. A dependency parse is a graph g where $g = (V_g, E_g, R_g)$. V_g is the set of words. We use the letters i, j, k for vertices/words in the dependency parses. E_g is the set of directed edges defined as pairs over vertices. R_g is the mapping from edges to dependency labels over edges or $R_g : E_g \rightarrow R$ where R is the set of dependency labels such as *subject, object, modifier*, etc. Elements of R_g are indexed by variables such as r_{ij}: the first index i is the head and the second index j is the dependent.

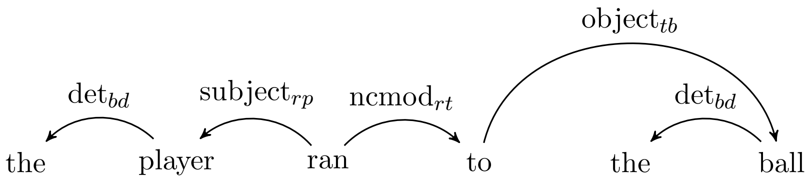
For the model variants described in this section, the basic frame of evidence is the dependency parse. Take the case of the dependency parse of Example 1.1

below:



What this shows is structural dependencies in natural language that are obscured by the left-to-right surface order.

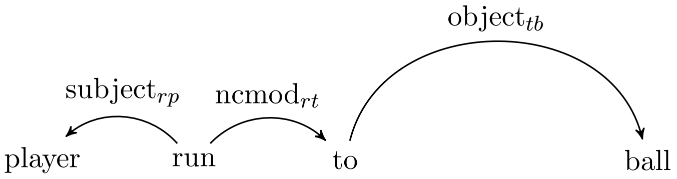
We show the dependency parse of the example that we will actually be working with:



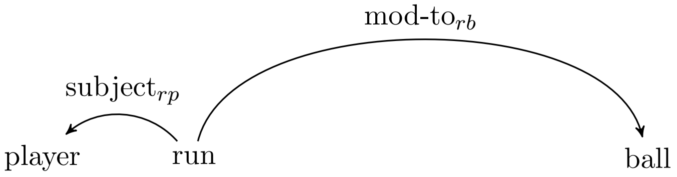
Compared to the previous sequential variant, we have more information on hand. First, we have relation labels which further specify the joint distributions that we work with. Also, the edges are directed and can point either left-to-right or right-to-left with no constraints on the length of the edges as long as the edge is contained within a sentence. Compare this with the sequential model where the edges are strictly left-to-right and are constrained to only connect words that immediately adjacent each other on the surface. Finally, with dependency edges, it is quite common to have a word have more than one dependency child (also called a *dependent*) and, while less common, it is possible for a node to have more than one dependency parent (also called a *head*).

So here's a broad outline of how we will be creating graphs: (1) Generate dependency parse (2) Remove function words and simplify dependency edges (3) Add paraphrase nodes (4) Add edges and create factor nodes (5) Remove original edges (6) Remove original observed nodes

As with the previous sequential model, we first drop most function words from the graph since we reductively assume they do not contribute to the meanings of words:

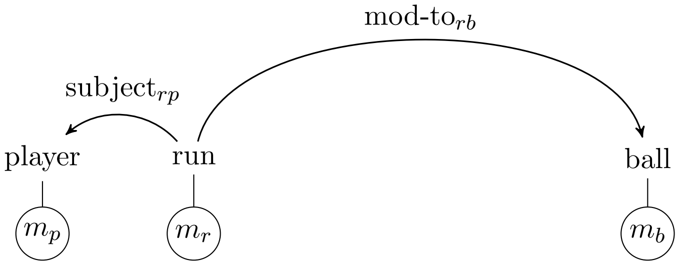


Then we modify dependency relations so that prepositions are incorporated into labels and edges:

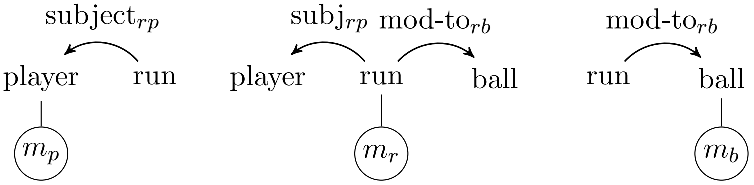


Therefore, though prepositions do not have the status of nodes, they still contribute information by specifying dependency labels. For example, we discarded information on how the box is *in* the pen by reducing the sentence to "box be pen," but we would be able to avoid this information loss by having an edge labeled *mod-in* between *be* and *pen*. And similarly, with the current example, we can distinguish between "player run ball" and "player run to ball" by having an edge labeled *mod-to* between *run* and *ball*.

Next, we add the paraphrase nodes.

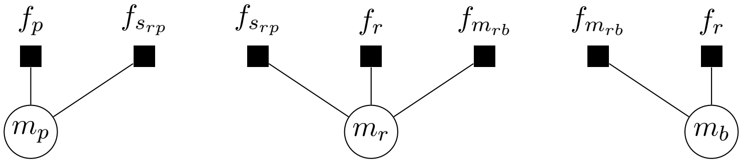


Then, same as the sequential model, we duplicate the observed words and create component graphs centered on each of the content words:



Note that, unlike the sequential model where there can be at most four nodes (three factor nodes and one paraphrase node) to a component, there can be an arbitrary number of factor nodes in a component for graphs based on dependency parses.

Finally, we take the observations and convert them to factor nodes. Two different types of conversions are involved. For the immediate observation that is paired with the paraphrase node (i.e. *player* for m_p, *run* for m_r, *ball* for m_b), it is converted immediately to f_p for *player*, f_r for *runner* and f_b for *ball*. For edges with relation labels, the conversion needs to retain more information: namely the relation labels themselves, the identity of the head, and the identity of the dependent. Formally, we do this by writing the relation label and its head first and dependent second as a subscript to the factor. For example, since *player* is the dependent of *run* under the relation *subject*, we indicate this factor by $f_{s_{rp}}$ where s stands for subject, r on the left denotes that *run* is the head in the subject relation and p on the right denotes that *player* is the dependent in the subject relation. Once such conversions are complete, all factors are connected to the paraphrase node in their respective component graphs:



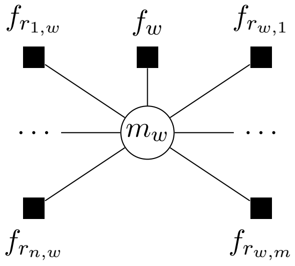
where s=*subject*, m=*mod−to*, p=*player*, r=*run*, b=*ball*. The paraphrase distribution for each paraphrase node is then:

$$p(m_p) \propto f_p(m_p)\ f_{s_{rp}}(m_p)$$

$$p(m_r) \propto f_r(m_r)\ f_{s_{rp}}(m_r)\ f_{m_{rb}}(m_r)$$

$$p(m_b) \propto f_p(m_p)\ f_{m_{rb}}(m_r)$$

Adjacency transformation (at) To distinguish it from the sequential variant that came before and the model variants that will follow, we give a name to this particular transformation of dependency parses. We call it the **adjacency transformation (at)**. Formally, we describe each component centered on the content word w and its paraphrase node m_w as follows:



where we have placed the factor nodes derived from the heads of w on the left and the factor nodes derived from the dependents of w on the right. The distinction between w being either the head or dependent in the relation is maintained by placing w second in the subscript under r if w is a dependent (e.g. $f_{r_{1,w}}$) and placing it first under r if it is a head (e.g. $f_{r_{w,1}}$). We call factors such as f_w which capture associativity between an observed word and its paraphrase node **word factors**. Factors such as $f_{r_{w,1}}$ which capture associativity between paraphrase nodes and adjacent observations in the dependency parse **word selectional factors**.

Given some dependency parse, the paraphrase distribution for some component centered on w is defined as:

$$p(m_w|\mathbf{s}) \propto f_w(m_w) \left(\prod_{j \in \Gamma^h} f_{r_{j,w}}(m_w) \right) \left(\prod_{k \in \Gamma^d} f_{r_{w,k}}(m_w) \right)$$

where Γ^h is the set of heads of w and Γ^d is the set of dependents of w.

Though it is not necessary for this particular model variant, the factorization of the **global function** over the set of all paraphrase nodes $\mathbf{m}$ within the full sentence as dependency parsed graph G is given as:

$$P_G(\mathbf{m}) \triangleq \prod_{i \in V_G} \left(f_i(m_i) \prod_{j \in \Gamma_i^h} f_{r_{ji}}(m_i) \prod_{k \in \Gamma_i^d} f_{r_{ik}}(m_i) \right)$$

where Γ_i^d is the set of nodes that are dependents of node i in the dependency graph, and Γ_i^h are nodes that are heads of node i.

Later in §3.3.3, we will discuss how we estimate the parameters for the factors in our models. We experiment with two types of estimation. In one, we do not learn the parameters for factors like f_w, $f_{r_{ij}}$ using an iterative procedure. Instead, we determine parameters using a simple surface count approach, based on the assumption that interactions involving hidden values follow the same parameters as interactions between observed words. In the other, we work with a different value space for the parameter nodes and learn the parameters through Gibbs sampling.

Canonical transformation (ct) The previous variant made the simplifying assumption that information was strictly local in the dependency parse. This is unsatisfactory since it is obvious that most people generally do not forget or ignore words that occur in the same utterance. Take the following examples:[8]

(3.12) The player ran to the ball.

(3.13) The debutante ran to the ball.

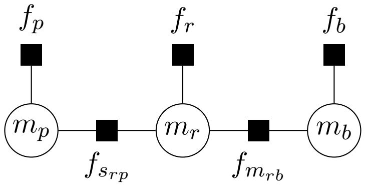
To correctly resolve the meaning of *ball* in these two contexts, it is necessary to know that *player* or *debutante* is the the subject of *run*. However, in the two

[8] These examples are due to Raymond Mooney

variants that we defined above—the sequential model and the adjacency transformed model—*player* and *debutante* are ignored because they exist outside the immediate neighborhood of *ball*. Thus *ball* means the same thing in the two sentences above according to our previous two model variants.

To correct this, we examine a new variant, one where paraphrase nodes are connected to other paraphrase nodes instead of being connected to observations. We call this transformation the **canonical transform (ct)** since one of the most canonical structures in NLP—the hidden Markov model—is a special case of this transform.

For our example, "the player ran to the ball," the transformation of its dependency parse generates the following factor graph:



where edges between content words are established only for the hidden paraphrase nodes. This models the assumption that semantic information in a sentence flows through a dedicated layer that is not observed, but mirrors the structure of the observed dependency parse. For this particular example, the shape of the graph is nearly identical to that of an HMM. Then the graph G corresponds to the following factorization of the global function over all paraphrase nodes $\mathbf{m}$:

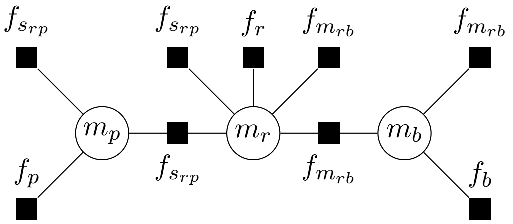
$$F_G(\mathbf{m}) \propto f_p(m_p)\ f_{s_{rp}}(m_r, m_p)\ f_r(m_r)\ f_{m_{rb}}(m_r, m_b)\ f_b(m_b)$$

It should be noted that factors such as $f_{s_{rp}}$ and $f_{m_{rb}}$ with dependency relation subscripts have been overridden so that they are binary factors (i.e. factors that take two arguments). Contrast this with the previous definition of $f_{s_{rp}}$ and the like in §3.3.1.2 where factors that involved dependency relations were unary factors. Furthermore, the order of the arguments is important and is not commutative. For example, $f_{s_{rp}}(m_r, m_p)$ cannot be written as $f_{s_{rp}}(m_p, m_r)$. This is to maintain the convention of heads preceding dependents. While we called the unary factors involving dependency relations in the previous at variant **word selectional factors**, we will call the binary factors defined for the current variant **selectional factors**.

More generally, for some set of paraphrase nodes $\mathbf{m}$ transformed from some dependency parse G, the canonical transformation of a dependency graph results in the following factorization of the global function:

$$P_G(\mathbf{m}) \propto \prod_{i \in V_G} f_{w_i}(m_i) \prod_{(j,k) \in E_G} f_{r_{jk}}(m_j, m_k) \tag{3.14}$$

Canonical+Adjacency transformation (cat) This is a combination of the canonical transformation and the adjacency transformation. It models both collocational strength of adjacent observations (i.e. at) as well as generalized information from the entire sentence (ct). The example transformation of "the player ran to the ball" is given below:



This particular variant is the reason why we avoided giving observed words their own nodes in the graph representations and in the functions for the factorizations. See the following factorization of the transformed graph:

$$\begin{aligned} P_G(\mathbf{m}) \propto & f_p(m_p)\ f_r(m_r)\ f_b(m_b) && \text{(word factors)} \\ & f_{s_{rp}}(m_p)\ f_{s_{rp}}(m_r)\ f_{m_{rb}}(m_r)\ f_{m_{rb}}(m_b) && \text{(word selectional factors)} \\ & f_{s_{rp}}(m_r, m_p)\ f_{m_{rb}}(m_r, m_b) && \text{(selectional factors)} \end{aligned}$$

Given that there are far more terms involved, we have labeled sets of product terms in parentheses on the right. The first line corresponds to the unary **word factors** that capture associativity between an observed word and its own paraphrase node. The second line lists the unary **word selectional factors** that capture associativity between a dependency connected observation and a paraphrase node. The final line lists the binary **selectional factors** that capture associativity between hidden paraphrase nodes and allow information to flow throughout the entire sentence. Note how, similar to argument-dependent lookup in programming languages, we give the same name to factors such as $f_{s_{rp}}(m_p)$ and $f_{s_{rp}}(m_r, m_p)$ but allow the number of arguments to disambiguate which is being used. Finally, to indicate that $f_{s_{rp}}(m_p)$ takes a dependency parent as its argument while $f_{s_{rp}}(m_r)$ takes a dependency child as its argument, we rely on the fact that the subscript of m_p is the same as the second subscript of s_{rp} and that the subscript of m_r is the same as the first subscript of s_{rp}.

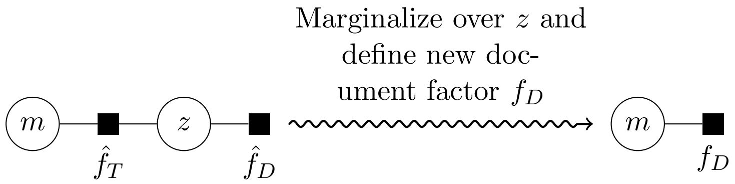


Figure 3.5: A hidden paraphrase distribution node m augmented by a topic variable z specific to document D. By marginalizing out z, we can define a new unary factor f_D over m.

In full generality, the canonical+adjacency transformation of a dependency graph results in the following factorization of the global function:

$$P_G(\mathbf{m}) \propto \left(\prod_{(i,j)\in E_G} f_{r_{ij}}(m_i, m_j) \right) \prod_{i \in V_G} \left(f_{w_i}(m_i) \prod_{j \in \Gamma_i^h} f_{r_{ji}}(m_i) \prod_{k \in \Gamma_i^d} f_{r_{ik}}(m_i) \right)$$

where Γ_i^h denotes the set of heads for node i in the original dependency parse and Γ_i^d denotes the set of dependents.

3.3.1.3 Wider document context (lda)

It was established early on that modeling bag-of-words context at the document level can help in word sense disambiguation for certain words [Yarowsky, 1995]. Given this evidence, and not quite convincing recent work that incorporate document level information through more sophisticated topic models [Boyd-Graber et al., 2007], we also examine the effects of document topic in inferring graded word sense. We include evidence on wider document context through a topic model [Blei et al., 2003]. Given a document D, the topic model defines a document specific distribution over topics $\hat{f}_D(z)$ and a distribution over words given a topic, $\hat{f}_T(m, z)$ (fig. 3.5). By marginalizing over z, we characterize the likelihood of each paraphrase candidate

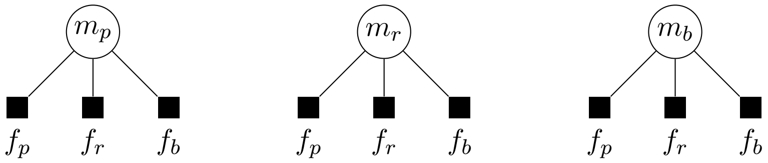


Figure 3.6: Sentence level bag-of-words representation

given the document and thus define a unary factor for a paraphrase distribution:

$$f_D(m) \triangleq \sum_z \hat{f}_T(m,z)\hat{f}_D(z)$$

All graph transformations above can be augmented with this unary factor. For example, the nodes m_p, m_r, m_b in the transformations of "the player ran to the ball" can be linked to the additional document factor f_D. Such a joint model incorporates lexical and syntactic evidence from the local sentence as well as topical evidence from the global document context.

3.3.1.4 Sentence bag-of-words context

It has been found in Erk and Pado [2010] that modeling graded word sense based only on sentence level bag-of-words features can help performance. To examine such evidence within our model, we consider a model where all content words influence all other content words without regard for dependency relations. In line with our previous models, observed content words are incorporated into unary factors and all such factors deriving from content words are connected to a paraphrase node. If we had pursued a different formulation with Markov random fields where surface tokens were given their own nodes, then this model would be represented with a complete graph. Since we have chosen to work with factor graphs and to incorporate constant-valued observed variables into any connected factors, we cannot define bag-of-words context as a complete graph. Instead, we follow with our current practice and create separate sets of factor nodes corresponding to observations for every paraphrase node. Using the example of "the player ran to the ball," there are three paraphrase nodes associated with each observation, and for each paraphrase node we create three new factors which correspond to *girl*, *catch*, and *ball* and attach them to their respective paraphrase node. This is laid out in a diagram in Figure 3.6.

3.3.2 Inference

In graphs that are trees or polytrees, the sum-product algorithm can be used for inference. However, some dependency parsers (including the one that we use) generate graphs that are not polytrees, so we assume that the graphical models over which we conduct inference may contain loops. Therefore, we use loopy belief propagation [Murphy et al., 1999] to approximate marginals. For graphs free of loops, loopy BP will converge to the correct marginal, and for graphs with loops the algorithm is known to perform well in practice [Weiss, 2000].

Because it is not possible to perform exact inference of the marginal for m_i (eqn. (3.11)) given a transformed dependency parse with loops, loopy BP instead approximates the marginal of m_i at some iteration $t{+}1$ based only on the values of the approximate marginals of its neighbors from the previous iteration t. In the sequence, we indicate this approximate, loopy marginal at iteration t by $P^{(t)}(m_i)$, dropping the subscript from P_S for notational clarity. We will simply call this "the marginal" in what follows, but it should not to be understood as an exact marginal.

Because of the variety of our models, we present the loopy BP update formula for the most specific model, cat+lda. The update equations for all other models can be derived from this by removing unnecessary terms from the formulas.

The update equation for the marginal of m_i for the cat+lda model variant at iteration $t{+}1$ is given by

$$P^{(t+1)}(m_i) = C(m_i) \prod_{j \in \Gamma_i^h} \left(\sum_{m_j} f_{r_{ji}}(m_j, m_i) P^{(t)}(m_j) \right) \prod_{k \in \Gamma_i^d} \left(\sum_{m_k} f_{r_{ik}}(m_i, m_k) P^{(t)}(m_k) \right)$$

where Γ_i^h and Γ_i^d are as above, and we define

$$C(m_i) \triangleq f_D(m_i) f_{w_i}(m_i) \prod_{j \in \Gamma_i^h} f_{w_j, r_{ji}}(m_i) \prod_{k \in \Gamma_i^d} f_{w_k, r_{ik}}(m_i)$$

$C(m_i)$ is merely the product of unary factors that do not change values over iterations: the document factor f_D, the word factor f_{w_i}, and the word selectional factors $f_{w_j, r_{ji}}$ and $f_{w_k, r_{ik}}$. The two terms that involve $P^{(t)}(m_j)$ and $P^{(t)}(m_k)$ above (marginals for head Γ_i^h and dependent Γ_i^j paraphrase nodes, respectively, of m_i) do not require messages from their neighbors as would be the case for exact sum-product updates. Instead, they approximate this by having incorporated at iteration t the marginal values that their neighbors had at iteration $t{-}1$. These values are then marginalized over at iteration $t{+}1$ for the node m_i.

Before the first iteration, all values for all nodes are set to one, so $P^{(0)}(m_i) = 1$. Then loopy BP is run until convergence or until a fixed number of maximum iterations is reached. In our case, we tested convergence by examining whether all probability values of the paraphrase distributions for all nodes changed less than a certain threshold over a single iteration. For the at model, a truncated version of the above loopy BP algorithm is the same as exact inference so it “converges” in one iteration.

To ensure numerical stability, the paraphrase distributions were renormalized to sum to one at each iteration.

3.3.3 Defining factors

A distinct advantage of using factors that derive from an undirected graphical model is that there are few restrictions on how the parameters for such factors are defined. It would be senseless to set the parameters with random values—though we could—but we are also not constrained by any requirement to abide by asymptotic notions of occurrence as would be for models with frequentist motivations—though, again, we could.

In our model, all that is required of the factors is that they reflect some form of associativity between the arguments involved: between an observed word and its paraphrases, between a paraphrase and another paraphrase connected through a dependency edge (as we do with the ct transform), or between a paraphrase and an adjacent observation connected through a dependency edge (as we do with the at transform).

In the following subsections, we discuss two different approaches to estimating these parameters. The first is estimated in a straightforward way from token counts. It is associated with the notion of paraphrase nodes as actual paraphrases.

The second is estimated through Gibbs sampling. Here, the value space of the paraphrase nodes is nameless indexes and have no inherent meaning.

3.3.3.1 Surface injected paraphrases

When the value space of the paraphrase nodes comprises real vocabulary items instead of nameless indexes, a simple solution to defining interactions between paraphrase nodes and paraphrase nodes (or paraphrase nodes and adjacent observed nodes) is to assume that the interactions model the selectional preferences of the relevant paraphrases or observations over some dependency relation. Under this interpretation, the paraphrase nodes are not hidden nodes in the conventional sense that they generate the observations or that they model some class label for the observations. Instead, the paraphrase nodes, which have the entire vocabulary as value space, can be understood as instantiating an alternative realization of selectional preference as constrained by the surface observations.

For example, consider the example of "the player ran to the ball," specifically the transformation as we defined it with the adjacency transformation in §3.3.1.2. Assume we are interested in learning the paraphrase distribution over the paraphrase node of *run* in this sentence. For convenience, there are only two valid paraphrases for *run*: *move* and *manage*. Then, straying from the notation of the previous sections, the paraphrase distribution in full is defined as follows:

$$\begin{aligned} P(\text{move}) &\propto f(\text{move}, \text{run}) f(\text{move}, \text{subj}, \text{player}) f(\text{move}, \text{mod-to}, \text{ball}) \\ P(\text{manage}) &\propto f(\text{manage}, \text{run}) f(\text{manage}, \text{subj}, \text{player}) f(\text{manage}, \text{mod-to}, \text{ball}) \end{aligned}$$

The word factors $f(\text{move}, \text{run})$ and $f(\text{manage}, \text{run})$ where *run* is the observed constant define the associativity between *run* and *move* and *run* and *manage* in the absence of context. The remaining factors instantiate the alternative realization of selectional preference defined above. Ignoring the observation *run* for the moment and the dependency relations over the edges, the word selectional factors assign different weights to the sequences "player move ball" and "player manage ball" through the weights associated with each of the factors and the different paraphrases: *move* and *manage*.

There are differing degrees of constraints placed on the paraphrase nodes by the surface observation. With the ct transformation of dependency trees, paraphrase nodes are constrained only by their corresponding word factors (i.e. observations) and adjacent paraphrase nodes whose inference is complete. For the at and cat transformations, in contrast, there is a selectional constraint placed on the paraphrases by the adjacent observations, in addition to, or instead of some of the constraints that are placed on the ct transform.

We then make the assumption that the selectional preferences that are reflected in the surface counts over our training corpora are legitimate parameters for modeling the alternative realization of selectional preference over paraphrase nodes or between paraphrase nodes and observations. As such, these **surface injected parameters** derive directly from plain surface counts over observed (*head*, *dependency relation*, *dependent*) triple counts in our training corpora. There is no iterative estimation procedure involved for learning these parameters.

Selectional factors: interpolated surface counts (int) We model the interaction between two paraphrase distributions m_i, m_j with respect to a relation r_{ij} as the maximum likelihood estimate for values of m_i occurring in relation r_{ij} to values of m_j. Because this estimate is likely to be sparse, we interpolate with bigram and unigram MLEs.

$$f_{r_{ij}}(m_i, m_j) \triangleq \lambda_1 P(m_i, m_j|r_{ij}) + \lambda_2 P(m_i, m_j) + \lambda_3 P(m_i|r_{ij}) P(m_j|r_{ij}) + \lambda_4 P(m_i) P(m_j)$$

where the weights λ_i sum to one. We describe how the interpolation parameters λ_i are determined in §4.5.

Selectional factors: exponentiated PMI (epmi) Raw frequency counts are known to adversely affect selectional preferences and are often transformed through pointwise mutual information, the log of two likelihood ratios. However, because factors/potential functions must be non-negative, we take the exponential of pointwise mutual information and end up with the original ratio:

$$f_{r_{ij}}(m_i, m_j) \propto \frac{P(m_i, m_j|r_{ij})}{P(m_i|r_{ij}) P(m_j|r_{ij})} + \beta_s$$

where the Ps are MLEs and β_s is some small smoothing constant.[9]

We derive the unary word selectional factor $f_{w_i, r_{ij}}$ directly from $f_{r_{ij}}$. This factor is merely one where the paraphrase node in either the head or dependent position has been swapped out for an observation and thus becomes a constant.

[9] Note that this definition for selectional factors uses $P(m_i, m_j|r_{ij})$ without interpolation.

Word factors: using vector space similarity We design the unary factor $f_{w_i}(m_i)$ to model semantic similarity between the observation w_i and the paraphrase distribution m_i, but only for actual paraphrase candidates. Let P_i be a set of known paraphrase candidates for w_i, let $v_i \in M$ (where M is the set of all words) be a value of m_i, and let $\vec{w_i}$, $\vec{v_i}$ be unit-length vectors for the two words, if they exist. (v_i may not have a vector due to insufficient attestations.) Then we define the factor, with smoothing constant β_w, as

$$f_{w_i}(v_i) \propto \begin{cases} \exp(\vec{w_i}^T \vec{v_i}) + \beta_w & \text{if } v_i \in P_i \text{ and } \vec{v_i} \text{ exists} \\ \beta_w & \text{if } v_i \in P_i \text{ and } \vec{v_i} \text{ does not exist} \\ 0 & \text{else} \end{cases}$$

3.3.3.2 Factors over nameless hidden nodes and parameter estimation

In this subsection, we discuss an alternative formulation of the value space for paraphrase nodes and how the parameters will be estimated. In contrast with the previous subsection, the paraphrase nodes defined here have a nameless set of indexes as its value space instead of words. With this approach, we lose the interpretability that came with using real words as the values of paraphrase nodes. On the other hand, our model is now more coherent in terms of parameter inference and learning, since the learned parameters derive from the same graphical structure to which the inference procedure is applied. Because our model is defined over arbitrary graphs, no closed form procedure exists for estimating the parameters. Therefore, we use Gibbs sampling for estimation.

We define some collection G of labeled dependency parses qua graphs $g \in G$ where $g = (V_g, E_g, R_g)$. V_g is the set of vertices. We use the letters i, j, k for vertices in the dependency parses. E_g is the set of directed edges defined as pairs over vertices. R_g is the mapping from edges to dependency labels over edges or $R_g : E_g \to R$ where R is the set of dependency labels. Elements of R_g are indexed by variables such as r_{ij}: the first index i is the head and the second index j is the dependent.

Then we can perform the usual canonical (ct), adjacency (at), or canonical+adjacency (cat) transforms on these parses. For concreteness, we only discuss cat. As a reminder, with such a transformation, each i generates two nodes, an observed node w_i and a hidden node m_i. An edge is established between the two. Relation edges are inserted between (m_i, m_j) pairs, (w_i, m_j) pairs, and (m_i, w_j) pairs iff $r_{ij} \in R_g$. Then the observed words w_i, w_j are incorporated into (1) the word selectional factor nodes $f_{r_{ij}}$ and $f_{r_{ij}}$ and (2) the word factors f_{w_i} and f_{w_j}. For convenience, we will refer to this transformed graph as g also. All subsequent mentions of g refer to the transformed graph and not the original unless explicit mention is made of sets such as V_g or E_g.

The probability mass function for the transformed graph g is defined as follows:

$$p(g) = \prod_{(p,q) \in E_g} f_{r_{pq}}(m_p, m_q) \prod_{i \in V_g} \left[f_{w_i}(m_i) \prod_{j \in \Gamma_i^h} f_{r_{ji}}(m_i) \prod_{k \in \Gamma_i^d} f_{r_{ik}}(m_i) \right] \qquad (3.15)$$

where Γ_i^d is the set of nodes that are dependents of node i in the dependency graph, and Γ_i^h are nodes that are heads of node i.

We are taking a Bayesian approach so there exists a set of hyperparameters

$\mathbf{h}$. Therefore, there's also a corresponding set of prior distributions $\boldsymbol{\Theta}$ such that

$$p(g|\mathbf{h}) = \int p(g|\boldsymbol{\Theta})p(\boldsymbol{\Theta}|\mathbf{h})d\boldsymbol{\Theta}$$

We will be assuming Dirichlet priors and using collapsed Gibbs sampling for learning, so the prior $\boldsymbol{\Theta}$ will not be an issue in implementation. For concision, we leave out hyperparameter $\mathbf{h}$ from all probability statements above and below. In full, they should all read $p(g|\mathbf{h})$.

Given the factorization in (3.15), we have four different factors/parameters with a different hyperparameter for each:

Factor	Hyper	Description
$f_{r_{pq}}(m_p, m_q)$	δ	A transition parameter from paraphrase node to paraphrase node
$f_{w_j,r_{ji}}(m_i)$	β	A transition parameter from observed node to paraphrase node
$f_{w_k,r_{ik}}(m_i)$	γ	A transition parameter from paraphrase node to observed node
$f_{w_i}(m_i)$	α	An emission parameter between observed node and its paraphrase node

Because the probability mass function applies over the entire corpus, the full statement is as follows:

$$p(G) = \prod_{g \in G} p(g)$$

where G is the collection of dependency parses over the entire corpus.

For the collapsed Gibbs sampler, we are interested in the following conditional distribution, the conditional probability of a paraphrase random variable given all other states and observations:

$$\begin{aligned}p(m_i^g|g\backslash\{m_i^g\}) \propto & \frac{\#(w_i, m_i) + \alpha_{w_i}}{\#(m_i) + \sum_w \alpha_w} \\ & \prod_{j \in \Gamma_i^h} \left((\#(w_j, r_{ji}, m_i) + \beta_{m_i}) \, (\#(m_j, r_{ij}, m_i) + \delta_{m_i}) \right) \\ & \prod_{k \in \Gamma_i^d} \left(\frac{\#(m_i, r_{ik}, w_k) + \gamma_{w_k}}{\#(m_i, r_{ik}) + \sum_w \gamma_w} \, \frac{\#(m_i, r_{ik}, m_k) + \delta_{m_k}}{\#(m_i, r_{ik}) + \sum_m \delta_m} \right)\end{aligned}$$

where $\#(\cdot)$ indicates the counts of the variables. m_i^g is the random variable corresponding to the paraphrase node for observation i in the graph g.

Then from the above posterior sampling step, we derive the following parameters:

$$f_w(m) = \frac{\#(w, m) + \alpha_w}{\#(m) + \sum_{w'} \alpha_{w'}} \tag{3.16}$$

$$f_{r_{hd}}(m_h, m_d) = \frac{\#(m_h, r_{hd}, m_d) + \delta_{m_d}}{\#(m_h, r_{hd}) + \sum_m \delta_m} \tag{3.17}$$

$$f_{w_d, r_{hd}}(m_h) = \frac{\#(m_h, r_{hd}, w_d) + \gamma_{w_d}}{\#(m_h, r_{hd}) + \sum_w \gamma_w} \tag{3.18}$$

$$f_{w_h, r_{hd}}(m_d) = \frac{\#(w_h, r_{hd}, m_d) + \beta_{m_d}}{\#(w_h, r_{hd}) + \sum_m \beta_m} \tag{3.19}$$

Chapter 4

Data and Evaluation measures

4.1 Test sets

Word usage models are typically evaluated on a paraphrasing task, often using the LexSub dataset (illustrated in ex 3.1). While the original Lexical Substitution task involved both the generation of paraphrase candidates and the computation of their weights for a given usage, EP08 and subsequent approaches focus on the second half of the task. They take the list of paraphrase candidates as given and weight them in a given context. Another paraphrasing dataset has been provided by Mitchell and Lapata [2008] (below **M/L**). It has human ratings for paraphrase appropriateness (on a scale of 1-7) for verbs in the context of different subject nouns. Given a target verb and subject noun, for example *discussion* *strayed*, participants rated the goodness of a paraphrase for the verb in this context, for example *digress*. Bieman and Nygaard [2010] provide a dataset of paraphrases for nouns in context (below twsi) collected on Amazon Mechanical Turk as a first step towards grouping usages into discrete senses. It contains paraphrases for the most frequent nouns of the English language, with sentence contexts taken from the English Wikipedia. The format of this dataset is similar to LexSub.

We use all three datasets for evaluation. For LexSub, we follow EP08 in focusing on the second half of the task, paraphrase weighting, taking the list of paraphrase candidates as given.[1] LexSub consists of 2000 instances of 200 target words (verbs, nouns, adjectives, and adverbs) in sentential contexts, which were taken from the English Internet Corpus [Sharoff, 2006]. To compile the list of potential paraphrases for a target, we proceed as follows: We first pool all paraphrases that LexSub annotators proposed for the target, and add all synonyms in all synsets of the target in WordNet 3.0. For *address.v*, the list of potential paraphrases contains, among others, *speak.v*, *direct.v*, *call.v* and *handle.v*. For use with a topic model, we use the full documents containing the LexSub sentences.[2] The M/L dataset comprises a total of 3,600 human similarity judgements for 120 experimental items. Mitchell and Lapata split the dataset by participants into a development and a test portion. For comparability, we evaluate on the test portion that they used. To the best or our knowledge, the twsi dataset has not so far been used to evaluate paraphrase ranking or word usage models. We use version 1 of the data,[3] using the raw data with substitutions for all sentence contexts rather than only the contexts that were assigned to senses later.[4] This dataset comprises 7577 sentences with paraphrases for 392 nouns. We compile lists of paraphrase candidates in the same way as for LexSub. The twsi dataset contains a high number of multi-word expressions (about 20%) among paraphrase candidates. Since our model currently cannot deal with multi-word paraphrases, we omit them for now.

[1] This means that we cannot compare our results directly with those of participants of the SemEval Lexical Substitution task.

[2] We thank Diana McCarthy for making the full documents of the LexSub sentences available to us.

[3] Maintained at the `aclweb repository`

[4] We thank Chris Biemann for making the raw data available to us.

4.2 Parsing

We use the C&C parser [Clark and Curran, 2007] to parse the LexSub and twsi datasets as well as the corpora from which we estimate probabilities. We transform prepositions from nodes to edge labels, and we retain only content words. All words are lemmatized.

4.3 Parameter estimation

For estimating selectional factor parameters, we use C&C parses of three corpora—the British National Corpus (bnc, 100 million words), the English Gigaword corpus (LDC2003T05, Giga, 1 billion words), and ukWaC [Baroni et al., 2009] (2 billion words)—and combine them (u+b+g). Giga and bnc also serve as training corpora for benchmark purposes. The TFP10 model computes its vector space on Giga while EP10 computes on bnc.

All words are lemmatized and paired with their part of speech. Word factor parameters are estimated based on the paraphrase lists described in the previous paragraph. Vectors for these paraphrases are computed using the DependencyVectors package[5] with log-likelihood ratio transformation. To learn topic model parameters, we randomly take 26000 documents from ukWaC and combine them with the full LexSub documents. This is a total of 14,227,219 tokens. We then learn topic parameters with MALLET [McCallum, 2002].

[5] `http://www.nlpado.de/~sebastian/dv.shtml`, Padó and Lapata [2007]

4.4 Testing convergence of inference

To test convergence, we examine whether all probability values of the paraphrase distributions for all nodes change less than 1e-4 over a single iteration. If there is no convergence by 1000 iterations, we terminate inference and collect the values at the last iteration. In the majority of sentences, the algorithm converges in less than 20 iterations.

4.5 Smoothing constants

For smoothing constant β_s, we take the smallest non-zero value of the respective unsmoothed factor and multiply that by 1e-4. We set β_w to 0.1. The interpolated smoothing parameters are set to $\lambda_1 = 0.9999, \lambda_2 = 9\text{e}{-5}, \lambda_3 = 5\text{e}{-6}, \lambda_4 = 5\text{e}{-6}$. The values were set after a few experiments indicated that the model performed better as the value for λ_1 increased but still required a very small amount of smoothing with interpolated values to prevent all inferred probabilities from collapsing to zero.

4.6 Evaluation measures

In this section, we discuss the evaluation measures used in the dissertation. The first two measures, generalized average precision (GAP) and precision out of ten (P10), we use are measures that reflect recall. The third, weighted accuracy (wAcc), is a more stringent one intended to reflect how precisely the model reflects the human counts on LexSub. The fourth and final is a modified precision and recall and is designed to capture precision and recall performance over thresholded probability values.

4.6.1 Generalized Average Precision (GAP)

For a LexSub, M/L, or twsi target word w, we use as our model's prediction the probabilities computed for the matching paraphrase distribution node, restricted to the paraphrase candidates for that target word in the dataset. Like previous papers, we evaluate performance on M/L using Spearman's rho, a nonparametric rank correlation measure. For LexSub, Thater et al [Thater et al., 2009] use Generalized Average Precision (GAP). Let A be a list of gold paraphrases for a given sentence, with gold weights $\langle a_1, \ldots, a_m \rangle$. Let $B = \langle y_1, \ldots, y_n \rangle$ be the list of model predictions as ranked by the model, and let $\langle b_1, \ldots, b_n \rangle$ be the *gold* weights associated with the model predictions (assume $b_i = 0$ if $y_i \notin A$). Let $I(b_i) = 1$ if $y_i \in A$, and zero otherwise. We write $\overline{b_i} = \frac{1}{i} \sum_{k=1}^{i} b_k$ for the average gold weight of the first i model predictions, and analogously $\overline{a_i}$. Then

$$\text{GAP}(A, B) = \frac{1}{\sum_{j=1}^{m} I(a_j)\overline{a_j}} \sum_{i=1}^{n} I(b_i)\overline{b_i} \qquad (4.1)$$

We report macro-averaged GAP.[6]

4.6.2 Precision out of ten (P10)

Earlier, the SemEval task defined a "precision out of ten" (P_{10}) measure for LexSub [McCarthy and Navigli, 2009]. It uses the model's ten top-ranked paraphrases as its prediction, and scores them by their gold weights. Let A and B be as

[6] Since the model may rank multiple paraphrases the same, we averaged over 10 random permutations of equally ranked paraphrases.

above. Let $B_{10} = \langle y_1, \ldots, y_{10} \rangle$ be the model's 10 top predictions. Then

$$P_{10}(A, B) = \frac{\sum_{y_i \in A \cap B_{10}} b_i}{\sum_{i=1}^{m} a_i}$$

We report macro-averaged P_{10}. However, even though both evaluation measures carry the name "precision", they have more in common with recall measures, as they report the gold weight recovered by the model relative to the full gold weight. Also, they both take gold weights into account, but not model weights, using only the ranking predicted by the model.

4.6.3 Weighted Accuracy (wAcc)

Therefore we propose the use of additional evaluation measures. The first is a measure of *weighted accuracy* (**wAcc**), which compares gold and model weights, testing how much of the model-assigned weight is allocated to the right paraphrases. Let $\langle b_1^m, \ldots, b_n^m \rangle$ be the model weights associated with the prediction list B such that the sum of weights is the same for gold and model: $\sum_i a_i = \sum_i b_i^m$. Then we define weighted accuracy as

$$\text{wAcc}(A, B) = \frac{\sum_{j=1}^{m} \min(a_i, b_i^m)}{\sum_i a_i} \tag{4.2}$$

When computing wAcc below, we normalize gold paraphrase weights to sum to one. Weighted accuracy is a variant of the weighted precision and recall scores defined by Erk and McCarthy [2009]. When the sum of weights is the same for gold and model, both their weighted precision and recall reduces to our weighted accuracy. We report macro-averaged wAcc. Note that wAcc is a stricter evaluation measure than GAP and P_{10}, as it considers the weights that the model assigns, not just the ranking that it produces.

4.6.4 Precision and Recall

In addition, we use precision and recall, computed at different model weight thresholds θ. Model predictions at θ are the paraphrases whose model weight is at or above θ: $B_\theta = \{y_i \in B \mid b_i^m \geq \theta\}$. Ignoring gold weights, we then compute precision and recall of B_θ with respect to A as usual.

4.6.5 Evaluating model with nameless hidden nodes and parameters

Here, we describe how we transform and evaluate output from the model with nameless hidden nodes (which we will call nh) and concordant parameters described in §3.3.3.2.

Because the value space of the hidden nodes are nameless, it is not possible to evaluate directly on LexSub whose gold data is composed of meaningful lexemes. Therefore, we transform the output from nh into a form that is compatible with the entries in LexSub.

Each target lemma w has a set of possible paraphrases M_w that we derive from either WordNet or LexSub. For each test sentence that a target lemma occurs in, we generate $|M_w|$ new sentences from this by removing the target from the test sentence, then placing the paraphrases in the spot vacated by the target. We then conduct standard loopy BP inference on this new sentence.

Taking the example of Example 3.2, again, we have the original sentence with the target *brightest*:

An evening of classical symphonic music, played by the next genera-

> tion stars in the American orchestral scene, can be savored at the New World Symphony, a special Miami institution that nurtures the best and *brightest* young symphonic musicians.

For the lemma *bright*, WordNet and LexSub provide the paraphrases (among others): `intelligent`, `luminous`, `clear`.

. . . that nurtures the best and *brightest* young . . .

We replace the original target with the paraphrases and generate a new sentence for each one:

. . . that nurtures the best and `intelligent` young . . .
. . . that nurtures the best and `luminous` young . . .
. . . that nurtures the best and `clear` young . . .

We conduct inference on each of the new sentences and compare the distribution inferred over *brightest* with each of the paraphrases in terms of Jensen-Shannon divergence (JS), since the values of JS are non-negative and the function is symmetric over its arguments. To be specific, we define $\mathbf{s}_{\texttt{int}}$ to be the sentence with `intelligent` in place of *brightest* and $\mathbf{s}_{br}$ to be the original sentence. Then we define $P(m_{\texttt{int}}|\mathbf{s}_{\texttt{int}})$ and $P(m_{br}|\mathbf{s}_{br})$ to be the probability mass functions inferred for the paraphrase nodes of `intelligent` and *brightest*, respectively, given the observed sentences. Then we measure JS between the two distributions. We also measure distances between *brightest* and `luminous`, *brightest* and `clear` and so forth for all possible paraphrases $M_{brightest}$. Therefore, the final list of probability values that will be evaluated in terms of GAP, P_{10} and wAcc could be:

$$\begin{aligned}
P(m = \texttt{intelligent}|\mathbf{s}_{br}) &= \mathrm{JS}(P(m_{\texttt{int}}|\mathbf{s}_{\texttt{int}}), P(m_{br}|\mathbf{s}_{br})) \\
P(m = \texttt{luminous}|\mathbf{s}_{br}) &= \mathrm{JS}(P(m_{\texttt{lum}}|\mathbf{s}_{\texttt{lum}}), P(m_{br}|\mathbf{s}_{br})) \\
P(m = \texttt{clear}|\mathbf{s}_{br}) &= \mathrm{JS}(P(m_{\texttt{cle}}|\mathbf{s}_{\texttt{cle}}), P(m_{br}|\mathbf{s}_{br}))
\end{aligned}$$

This is actually incorrect. JS, being a measure of difference, will place larger values on more dissimilar distributions whereas we want larger values on more similar distributions. Therefore, we flip the above values by finding the maximum of all JS between *brightest* and each paraphrase in $M_{brightest}$ and subtracting each JS from the maximum. With examples, we call this maximum value $\max_{\mathrm{JS}}$:

$$\begin{aligned}
P(m = \texttt{intelligent}|\mathbf{s}_{br}) &= \max_{\mathrm{JS}} - \mathrm{JS}(P(m_{\texttt{int}}|\mathbf{s}_{\texttt{int}}), P(m_{br}|\mathbf{s}_{br})) \\
P(m = \texttt{luminous}|\mathbf{s}_{br}) &= \max_{\mathrm{JS}} - \mathrm{JS}(P(m_{\texttt{lum}}|\mathbf{s}_{\texttt{lum}}), P(m_{br}|\mathbf{s}_{br})) \\
P(m = \texttt{clear}|\mathbf{s}_{br}) &= \max_{\mathrm{JS}} - \mathrm{JS}(P(m_{\texttt{cle}}|\mathbf{s}_{\texttt{cle}}), P(m_{br}|\mathbf{s}_{br}))
\end{aligned}$$

and normalize it to sum to one.

We now formalize the above. Given a target word w, a set of possible paraphrases M_w and a context $\mathbf{s}_w$, for each pa**r**aphrase $r \in M_w$, we generate $|M_w|$ new sentences $\mathbf{s}_r$ for each $r \in M_w$. We infer a probability distribution $p(m_r|\mathbf{s}_r)$ over each paraphrase node m_r in context $\mathbf{s}_r$. We also infer the usual probability distribution $p(m_w|\mathbf{s}_w)$ over paraphrase node m_w for the original target sentence with the original target word w. We then calculate JS values between $p(m_w|\mathbf{s}_w)$ and each $p(m_r|\mathbf{s}_r)$ for each $r \in M_w$. We also find the maximum JS value

$$\max_{\text{JS}} \triangleq \max_{r \in M_w} \text{JS}(p(m_w|\mathbf{s}_w), p(m_r|\mathbf{s}_r))$$

Then the final "probability distribution" that is evaluated is defined as

$$P(m = r|\mathbf{s}_w) \propto \max_{\text{JS}} - JS(p(m_w|\mathbf{s}_w), p(m_r|\mathbf{s}_r))$$

Chapter 5

Experiments and Results

In this section we discuss experimental results on the task of predicting paraphrase appropriateness on the LexSub, twsi and M/L datasets. We refer to our group of models as **pd** for *paraphrase distribution* models. As mentioned before, our focus is not on numerical comparison to existing systems, but on the ability to test many knowledge sources and their interactions.

5.1 LexSub: Evaluation against benchmark and baseline models

Table 5.1 uses the LexSub dataset to compare the best performing variant (the at variant) of the pd model and the sequential variant (§3.3.1.1) to benchmark

	GAP					wAcc
	all	verb	noun	adj	adv	
seq (u+b+g)	45.88	41.74	45.89	46.46	51.98	25.73
pd (u+b+g)	**47.76**	44.90	**48.51**	**47.60**	51.49	**26.70**
pd (Giga)	46.68	42.92	46.86	46.18	**53.58**	24.94
pd (bnc)	43.42	38.88	44.39	43.61	48.98	22.28
singl	36.5	30.8	37.1	38.5	41.5	21.72
rand	30.0	27.4	30.3	28.1	36.3	21.34
TFP10 (Giga)	-[1]	**45.17**	46.38	43.21	51.43	-
EP10 (bnc)	38.6	36.9	41.4	37.5	-	-

Table 5.1: LexSub data: GAP and wAcc scores. Evaluation on the full dataset (all), and by target POS. Condition for pd parameters: epmi, at.

[1] TFP10 do not provide a joint GAP across all parts of speech.

and baseline results. We list two benchmarks, TFP10 and EP10. TFP10 reports better results than any previous syntax-based usage vector model, including EP08, so it constitutes the current state of the art. We also list EP10 because of its different modeling choice: It is based on a bag-of-words representation of the sentence rather than syntactic neighborhood. Note that TFP10 uses GIGA as a basis, while EP10 uses BNC, so those two approaches do not compare directly, and should be compared to variants of pd trained on GIGA and BNC respectively. The pd conditions shown are the best model variant (at+epmi) with parameters from the joint U+B+G (UKWAC+BNC+GIGA) corpus, from the BNC corpus only, and from GIGA. The sequential variant also uses epmi parameters.

We list two baselines: singleton and random. The singleton baseline (**singl**) assumes that the target paraphrase distribution is connected only to its observation, i.e. there is no contextual information. The random baseline (**rand**) assigns random probabilities to the paraphrases.

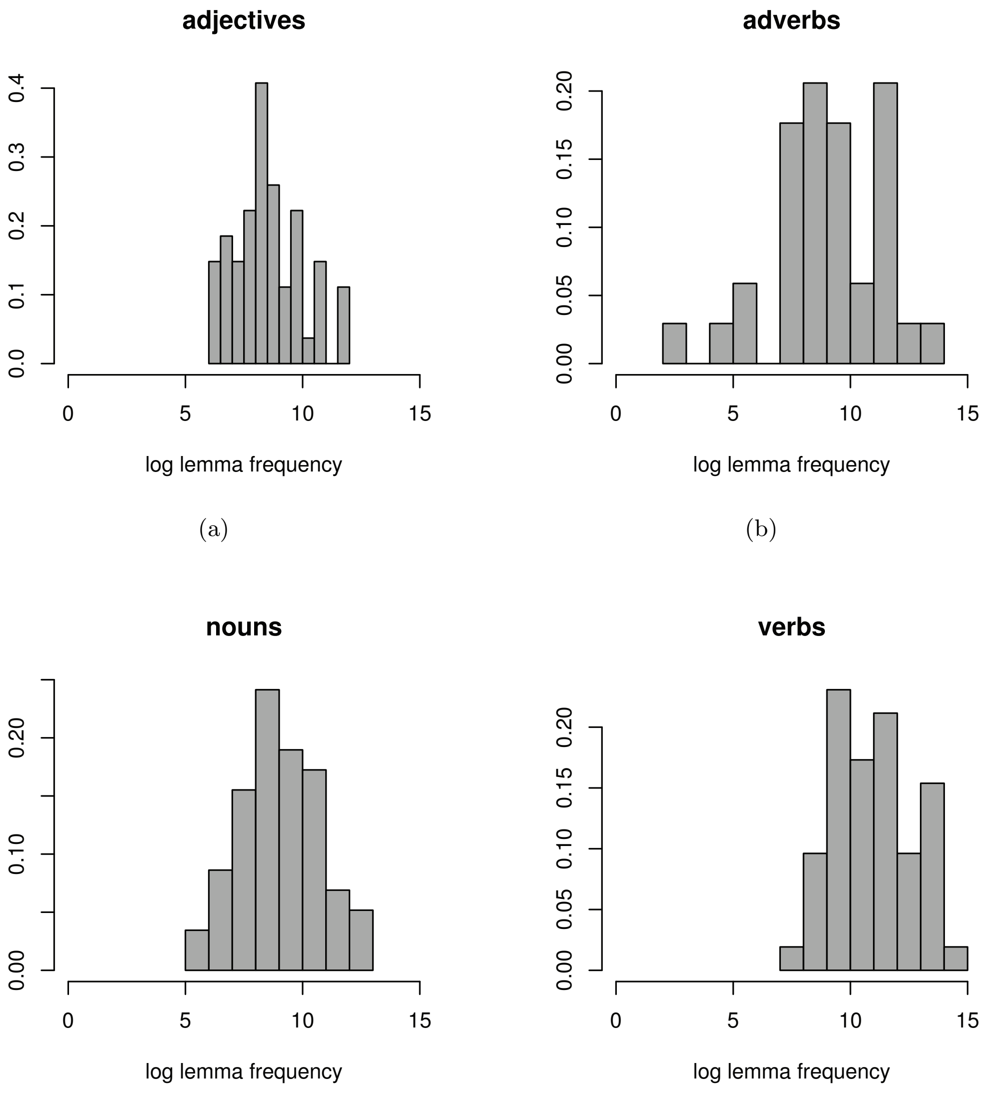
On verbs, TFP10 shows the best performance as measured by GAP. On all other parts of speech, the pd variants with U+B+G parameters and GIGA parameters have the best performance, with an especially large advantage on adjectives. Compared to the BNC and GIGA conditions, U+B+G shows better results for nouns and verbs in particular. The GIGA condition performs best for adverbs. The contrast of BNC with GIGA and U+B+G indicates that the use of more data to estimate selectional factors has a considerable impact. Comparing BNC and seq, we see that using surface structure with more data improves over using dependency structure with less data. The singleton baseline is higher than the random baseline, but lower than any other model.

Weighted accuracy scores (given only for pd as this is a new evaluation measure not reported in previous papers) show that the best pd model allocates about a quarter of its weight correctly. One reason for this is that pd assigns nonzero weights to many more paraphrase candidates than are listed among the gold paraphrases. The scores also confirm that wAcc is a very strict measure, as the random baseline, at 21.34, is not far below the best result of 26.70.

We also report P_{10} scores for completeness, but do not show them in further analyses, as they evaluate more or less the same properties of the models as GAP. The pd variant with U+B+G parameters attained the highest overall score with 69.46. The sequential variant scored 67.34. For the baselines, the scores are 62.54 (singl) and 59.61 (rand). TFP10 report a P_{10} of 75.43 for verbs, but do not give the score for other parts of speech. For comparison, the pd variant with GIGA achieves a P_{10} of 67.97 on verbs. EP10 do not report P_{10}.

The GAP scores by part of speech follow a familiar pattern – for all approaches except TFP10 – in that results for verbs are lower than for all other parts of speech. However, the figures in Figure 5.1 suggest another explanation besides the general difficulty of verb contextualization. It shows log-transformed frequencies for LEXSUB lemmas by part of speech. Frequencies for LEXSUB verbs go higher than those for any other part of speech. As is well known, high-frequency lemmas tend to be more ambiguous, which makes them more difficult to contextualize. This is a problem for all approaches that evaluate on the LEXSUB data.

One possible reason why TFP10 achieves the highest scores for verbs is that



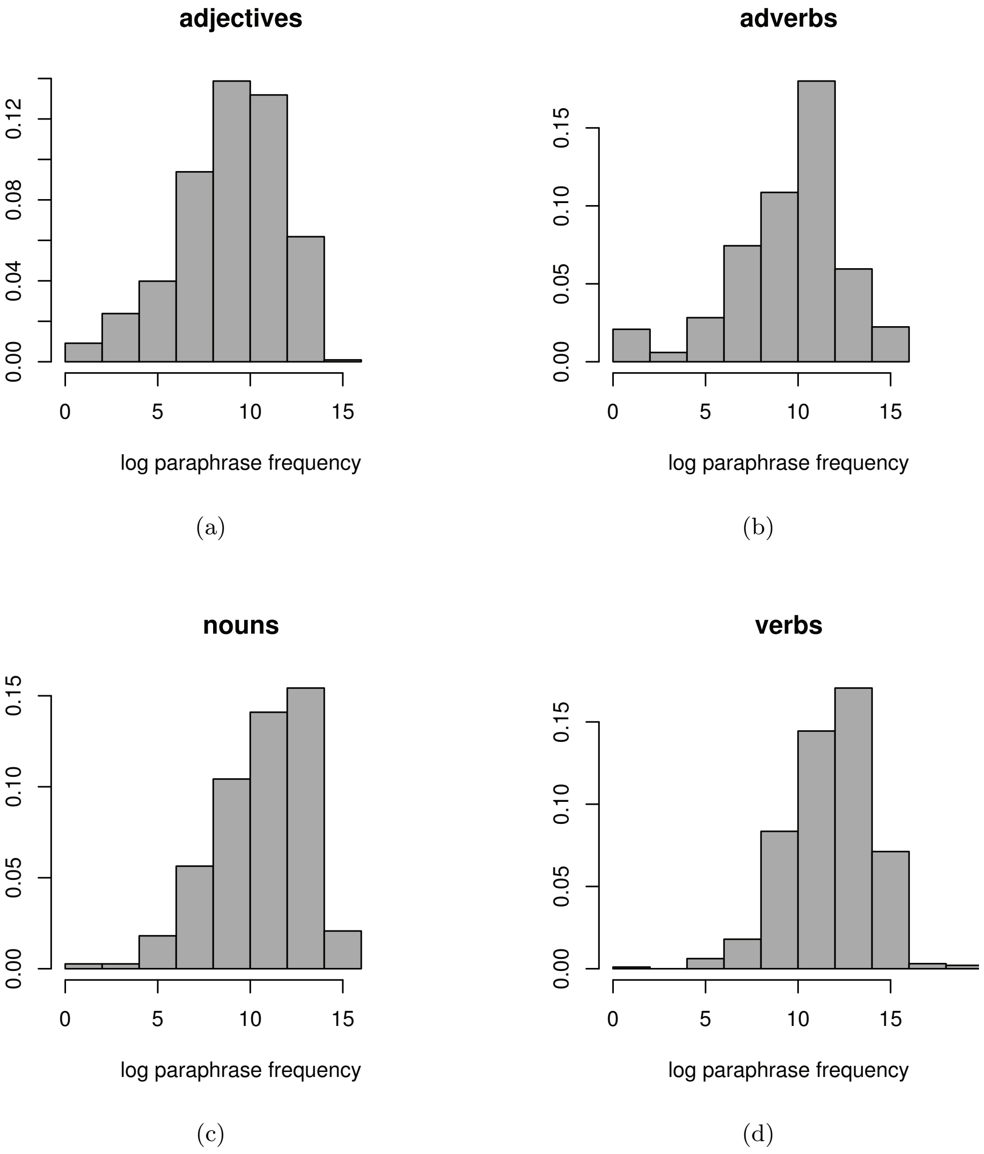
(a)

(b)

(c)

(d)

Figure 5.1: LexSub log lemma frequency by parts-of-speech



(a)

(b)

(c)

(d)

Figure 5.2: LexSub log paraphrase frequency by parts-of-speech

	lemma freq.		paraph. freq.	
corpus	GAP	wAcc	GAP	wAcc
BNC	10 (83%)	6 (50%)	10 (83%)	2 (16.7%)
U+B+G	10 (83%)	5 (41.7%)	9 (75%)	8 (66.7%)

Table 5.2: LexSub: Number of conditions for which there is a significant negative correlation between lemma or paraphrase frequency and model performance ($p \leq 0.05$)

the vectors used by TFP10 have much higher dimensionality than the paraphrase state space of pd. Perhaps having more dimensions is especially beneficial for verbs in modeling fine sense distinctions. It would be interesting to test whether modeling a larger state space that includes more than paraphrases improves pd performance. Another possible explanation is that TFP10 uses the target's headword as a stand-in for the target for adjectives and adverbs. This may be suboptimal for estimating similarity to the target's paraphrases. A third possible reason lies in paraphrase frequencies. The figures in Figure 5.2 show log-transformed frequencies for paraphrases (both WordNet- and LexSub-derived) of LexSub lemmas. Verb lemmas not only tend to be of higher frequency, but often have higher-frequency paraphrases as well. This may make things especially difficult for our model, as selectional preference parameters can be expected to be of lower quality for high-frequency paraphrases. We tested whether this is indeed the case by measuring correlation (using Spearman's rho) between average paraphrase frequency and performance. Table 5.2 shows the results. Here and below, we concentrate on BNC and U+B+G and omit Giga, which was only included for comparability with TFP10. Of the 12 conditions for each corpus (at, ct, cat with and without lda), many show a significant correlation between lemma frequency and paraphrase frequency on the one hand, and model performance on the other hand. The correlation is negative: Performance goes down as lemma or paraphrase frequency rises. The correlation is more pronounced for GAP analysis than for wAcc.

5.2 Model output examples

Figures 5.3 and 5.4 show examples of the `pd` model's output for the sentences from Figure 1.1. The distributions are sorted in descending order. We are listing LexSub paraphrase candidates only, omitting paraphrases from WordNet. Gold paraphrases for each datapoint are boldfaced. Here and in general, the model produces highly skewed distributions with few high-probability items. For both sentences, the three model variants produce similar paraphrase rankings, no matter whether the selectional information comes from observed words or hidden variables. For sent. 1812, there is a modifier relation between *drug* and *charge*, and thus words typically modified by *drug* (and its paraphrases) are ranked highly. In sent. 1813, *charge* is the dependent of a MOD-AGAINST relation, so words which are typical dependents of MOD-AGAINST such as *criticism* and *accusation* are highly ranked by the model. In sent. 1813, the probability distributions produced by `at` and `ct` happen to coincide for *charge*, even though they differ for other content words in the sentence.

sent. 1812 . . . by federal law enforcement agencies on drug charges, in others while traffickers . . .

ct		at		cat	
issue	3.26e-01	control	5.00e-01	issue	5.73e-01
control	1.38e-01	issue	3.96e-01	control	2.98e-01
authority	8.35e-02	payment	3.00e-02	payment	6.37e-04
power	6.69e-02	**allegation**	2.38e-07	authority	1.16e-11
payment	4.95e-03	**offence**	2.14e-08	power	5.95e-12
offence	5.66e-08	expense	1.54e-09	**offence**	6.76e-15
cost	4.95e-10	authority	2.34e-11	**allegation**	1.07e-23
allegation	1.17e-14	power	2.10e-11	expense	2.98e-27
expense	7.66e-17	tariff	1.57e-17	cost	1.83e-27
tariff	4.14e-17	cost	1.25e-18	tariff	4.45e-33
command	1.10e-17	prosecution	7.91e-19	prosecution	7.46e-36
prosecution	2.09e-18	**accusation**	5.81e-19	**accusation**	2.73e-36
accusation	1.53e-18	fee	4.35e-20	fee	2.51e-38
fee	1.15e-19	**indictment**	8.11e-27	command	2.61e-43
indictment	2.14e-26	criticism	7.81e-27	**indictment**	5.08e-52
criticism	2.06e-26	command	5.46e-27	criticism	4.90e-52

Figure 5.3: LexSub: Sample pd model output (U+B+G, epmi) on the sentences of Figure 1.1

sent. 26 If you don't take the risk of dying by driving to the store, your house could collapse on you and kill you anyway.

ct		at		cat	
tolerate	8.21e-01	assume	1.99e-01	assume	4.27e-01
consider	4.21e-02	run	1.29e-01	run	1.79e-01
get	1.43e-02	accept	9.48e-02	accept	9.72e-02
run	8.58e-03	tolerate	6.93e-02	tolerate	5.21e-02
be	7.28e-03	consider	6.53e-02	consider	4.62e-02
include	3.34e-04	happen	5.60e-02	happen	3.40e-02
risk	3.87e-07	risk	3.56e-02	risk	1.39e-02
happen	7.40e-08	be	1.53e-02	be	2.54e-03
grasp	6.51e-08	grasp	1.23e-02	grasp	1.64e-03
assume	5.59e-08	get	7.12e-03	get	5.50e-04
accept	3.30e-08	include	2.11e-03	include	4.83e-05
grow	1.28e-09	occur	2.02e-03	occur	4.40e-05
occur	1.22e-09	grow	1.60e-03	grow	2.77e-05
last	6.46e-10	start	9.55e-04	start	9.88e-06
start	3.80e-10	collect	3.67e-04	collect	1.46e-06
collect	2.79e-10	undergo	3.46e-04	undergo	1.29e-06
undergo	2.28e-10	begin	1.48e-04	begin	2.37e-07
begin	2.19e-10	occupy	9.97e-05	occupy	1.08e-07
gather	1.34e-10	gather	8.72e-05	gather	8.24e-08
occupy	2.78e-11	last	9.81e-12	last	8.25e-17

Figure 5.4: LexSub: Sample pd model output (u+b+g, epmi) on the sentence at top.

5.3 Influence of collocational information

Table 5.3 compares different variants of the pd model on the LexSub dataset. Best bnc and best u+b+g scores are boldfaced. at and cat GAP scores are consistently higher than those achieved by ct, and at consistently outperforms ct in terms of wAcc. So we can conclude that in the context of these experiments, at least, including collocational information improves performance. Comparing at and cat, while there are only negligible differences in GAP between the two, there is a noticeable difference in terms of wAcc. This means that at is better at apportioning probability mass in the paraphrase distributions so that it more closely aligns with LexSub.

Table 5.4 shows GAP and wAcc results by POS for models trained on u+b+g. For both evaluation measures, we see that at shows the best performance across the board. The only exception is that cat achieves better performance for nouns under epmi factors. Comparing only between at variants, the epmi+at condition beats the int+at condition. Surprisingly, it is at with int that performs best on adverbs in terms of GAP. Both tables show that cat does better than ct on ranking paraphrases (GAP), but has more problems than at and ct when assigning weights

	u+b+g				bnc			
	int		epmi		int		epmi	
GT	GAP	wAcc	GAP	wAcc	GAP	wAcc	GAP	wAcc
ct	42.16	22.11	46.13	24.24	42.28	20.62	42.45	20.85
at	43.14	22.91	47.76	**26.70**	42.56	20.73	**43.48**	**22.28**
cat	42.82	20.73	**47.77**	23.22	42.52	17.48	43.44	20.23

Table 5.3: LexSub data: GAP and wAcc scores by corpus, graph transformation and factor type. GT=graph transformation.

		int				epmi			
	GT	verbs	nouns	adj	adv	verbs	nouns	adj	adv
GAP	ct	36.36	42.19	42.86	50.78	42.97	48.00	45.23	50.01
	at	37.24	43.76	43.86	50.77	**44.91**	48.51	**47.60**	51.49
	cat	36.52	43.06	43.77	51.45	44.34	**48.70**	47.50	**52.37**
wAcc	ct	18.60	22.65	21.94	27.68	22.51	24.31	24.61	27.14
	at	19.58	24.09	22.25	27.99	**25.56**	**26.10**	**27.21**	**29.69**
	cat	15.39	21.72	21.44	26.83	21.33	23.57	22.76	27.16

Table 5.4: LexSub data: GAP and wAcc scores by POS with U+B+G. GT=graph transformation.

(wAcc).

5.3.1 Collocation isolated from semantic vector space

Here, we examine the contributions of collocational information isolated from the influence of the semantic vector space factor—i.e. the word factor—defined in §3.3.3.1. Instead of a sophisticated vector space model[2] defining the associativity between words and their potential paraphrases, the variant examined here merely assumes a uniform association between words and their potential paraphrases that derive from WordNet and LexSub. For example, if the paraphrases for the content word *bright* are `promising, luminous, shiny`, then the word factor $f_{bright}(m)$ is defined to be:

$$f_{bright}(m = \texttt{promising}) = 1/3$$

$$f_{bright}(m = \texttt{luminous}) = 1/3$$

$$f_{bright}(m = \texttt{shiny}) = 1/3$$

[2] derived from the DependencyVectors package (`http://www.nlpado.de/~sebastian/dv.shtml`)

That is, the uniform probability mass function over the three possible paraphrases for *bright*.

	GT	all	verbs	nouns	adj	adv
GAP	ct	44.43	42.37	45.38	43.63	47.56
	at	46.49	**44.94**	47.00	45.94	49.15
	cat	**46.62**	44.24	**47.07**	**46.05**	**50.86**
wAcc	ct	23.45	22.22	23.07	23.65	26.54
	at	**26.68**	**25.84**	**26.29**	**26.92**	**29.30**
	cat	23.07	21.32	23.21	22.64	27.13

Table 5.5: Collocational baseline scores for GAP and wAcc by POS with U+B+G. GT=graph transformation. Selectional factors are epmi

Table 5.5 shows the collected results by part-of-speech with GAP and wAcc scores for this variant. The only difference between our best, standard model and this collocational variant is that this variant assumes a uniform distribution whereas the standard model does not, instead deriving its parameters from a sophisticated vector space model of word meaning. Comparing this with the standard model allows us to determine how much in performance we are gaining through such a vector space model. In Table 5.1, the score is 47.76 for the best model and 46.49 for the collocational variant. While the difference is statistically significant, the difference is far less than that between the best model and the singleton baseline. The singleton baseline is the complement of the collocational baseline, where the sophisticated vector space model has been retained but all selectional factors reflecting collocation are removed. The situation is similar by part-of-speech and by graph transformation. The removal of the vector space model decreases performance but in the limited range of one to two percentage points.

5.4 Parameters for selectional factors

Comparing performance for U+B+G and BNC in Tables 5.3 and 5.4, we find that scores for U+B+G are better than corresponding BNC scores across the board, so estimating selectional factors from more data is consistently helpful. In comparing int and epmi, we see that epmi consistently outperforms int. This shows that it is important to dampen frequency-related noise when using selectional factors.

While TFP10 used a cutoff on both counts and pmi values, we do not apply any sort of cutoff.[3] This indicates that the core model with epmi is capable of effectively incorporating very small counts to make reliable inferences.

5.5 Analysis of precision and recall

The plots in Figures 5.5 and 5.6 show precision and recall at different weight thresholds θ for epmi parameters derived from the U+B+G and BNC corpora respectively. Points are shown in the order from highest to lowest θ, 0.9 - 0.0, in steps of 0.05. At θ between 0.0 and 0.2 (rightmost points), we have high recall of close to 90% at a precision of around 20%. This underscores again that pd model variants tend to report nonzero probabilities for many more paraphrase candidates than are listed among the gold paraphrases. (Note, though, that we evaluate only on words that are LexSub paraphrase candidates, not paraphrase candidates from WordNet.) With higher θ, recall drops fast, showing that most paraphrases in the models have probability between 0.0 and 0.1, as can also be seen for some examples

[3] We leave out details of experiments where we varied cutoff values and found that no cutoff of any sort performed the best.

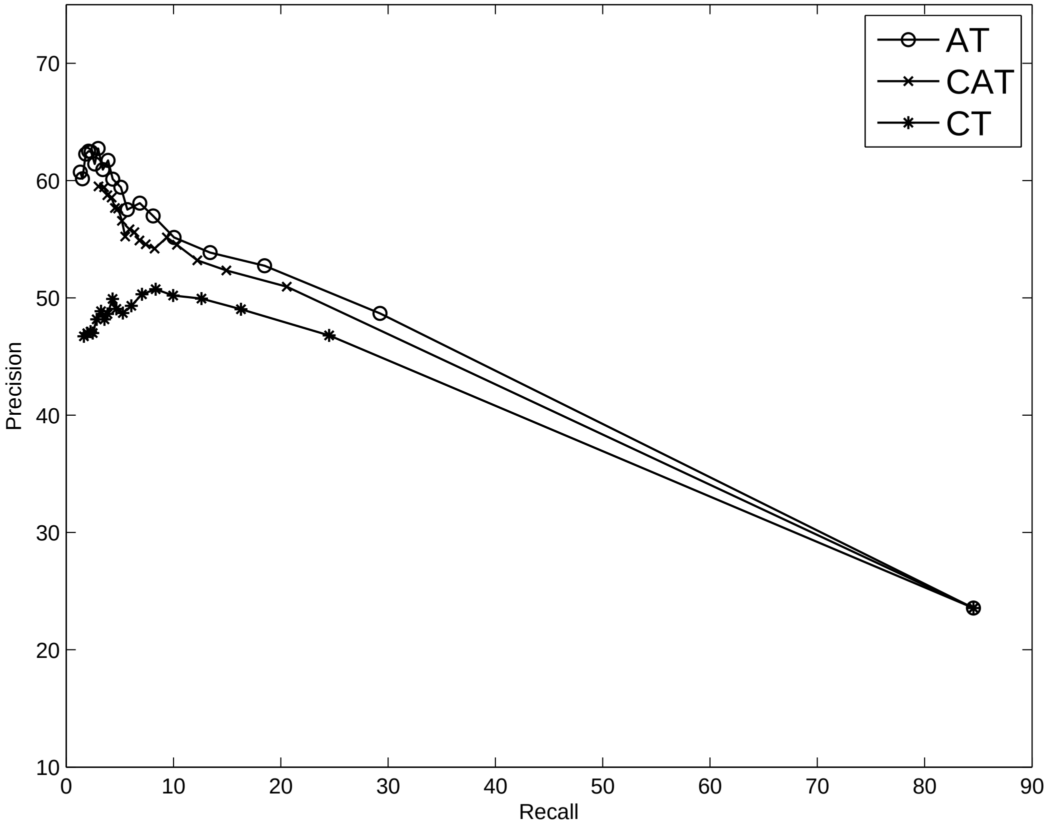


Figure 5.5: LexSub data: Precision/recall graphs over threshold by graph transformation. Results from U+B+G epmi parameters.

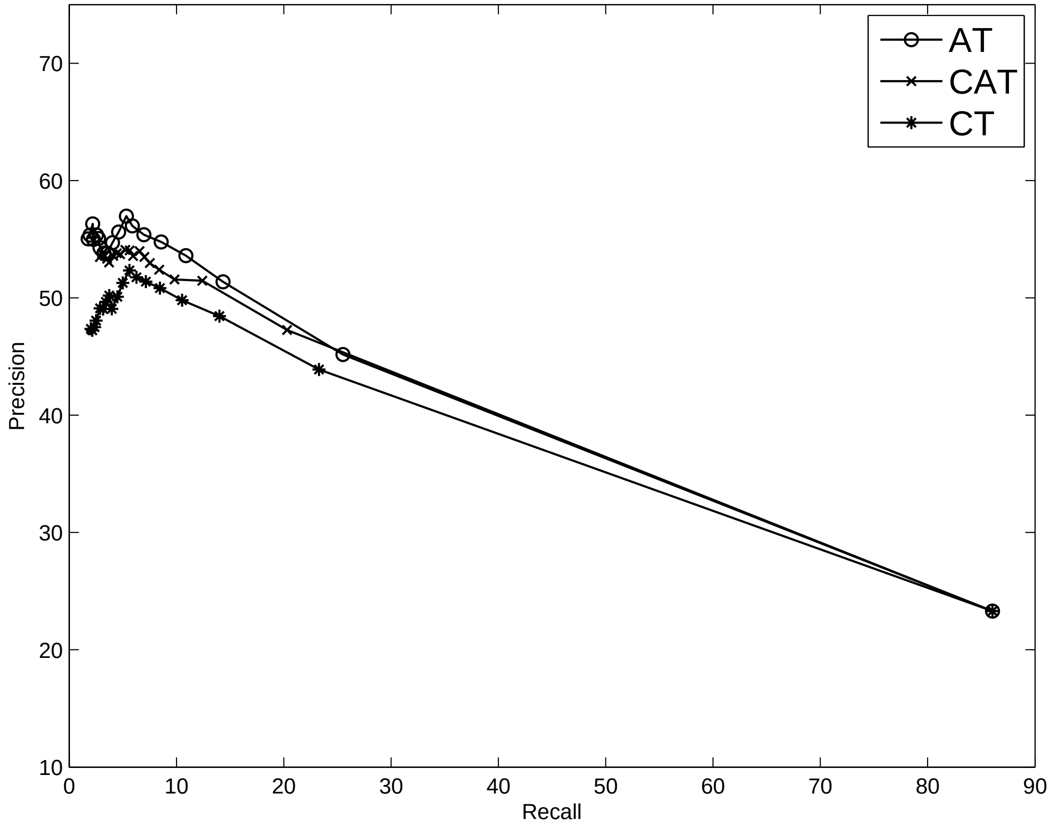


Figure 5.6: LexSub data: Precision/recall graphs over threshold by graph transformation. Results from BNC epmi parameters.

in Figures 5.3 and 5.4. Precision keeps rising as the threshold rises, indicating that the paraphrases with particularly high predicted probability tend to be correct. As in the GAP analysis, we get a performance ordering of at > cat > ct, in particular for high thresholds.

5.6 Document topic

We next compare results of pd models with and without lda derived factor nodes. We present results in Table 5.6 only for the at condition; results for other conditions are comparable. The first line shows the result of a baseline lda experiment where each target node was given no syntactic context (i.e. equivalent to the singleton baseline) and augmented only with the document based lda factor. We note that it is even lower than the singleton baseline. Combining selectional information through int with topic information via lda provides stronger results on GAP for BNC over parameters derived from U+B+G. With U+B+G, there is a insignificant numerical improvement over the results from BNC in terms of wAcc. Outside of the slight anomaly where BNC beats U+B+G in terms of GAP with int+lda, the only observation that holds across experiment settings is that lda detracts from

	BNC		U+B+G	
Model	GAP	wAcc	GAP	wAcc
lda only			32.24	15.41
int	42.56	20.73	44.56	22.98
int+lda	37.32	16.49	36.68	16.69
epmi	43.47	22.28	47.13	26.53
epmi+lda	42.21	19.20	45.10	22.57

Table 5.6: Models with and without document topic factors. All models (except lda only) shown only in at condition.

performance, though less so when the parameters are epmi.

Overall, the contribution of LDA-derived factor nodes is disappointing, a marked difference to the usefulness of document context features in traditional WSD. One possible reason is that LDA topics may not provide much information for the words in our paraphrase distributions. LDA information will be most useful for words whose probability differs strongly across topics. We compute the entropy of a word across topics as a measure for its "topicality". Figure 5.7 shows density plots for the entropies of the paraphrases of LexSub lemmas, as well as for the top 30 words in all LDA topics that we used. The dotted line plots the entropy for the top LDA topic words, and the solid line plots the entropy of the LexSub paraphrase candidates. We see that paraphrases have two modes, one at high entropy, which indicates low "topicality", and another in the middle but still higher than a sizable portion of the highly ranked words. So using a topic model, though intuitively the most obvious approach to including document topic information, might not be the most suitable for this data set, but a different model of document context still may be.

To evaluate whether this discrepancy in entropy between words of high topicality according to the topic model and the entropy of the targets in LexSub has an influence on model performance, we conducted a rank correlation test (Spearman's rho) comparing performance for each datapoint with the average entropy of the paraphrases for that datapoint's target. We found no significant correlation for GAP, but did find highly significant correlation ($p < 0.01$) for wAcc. In a plot of entropy by GAP (Figure 5.8) and a plot of entropy by wAcc (Figure 5.9), a negative

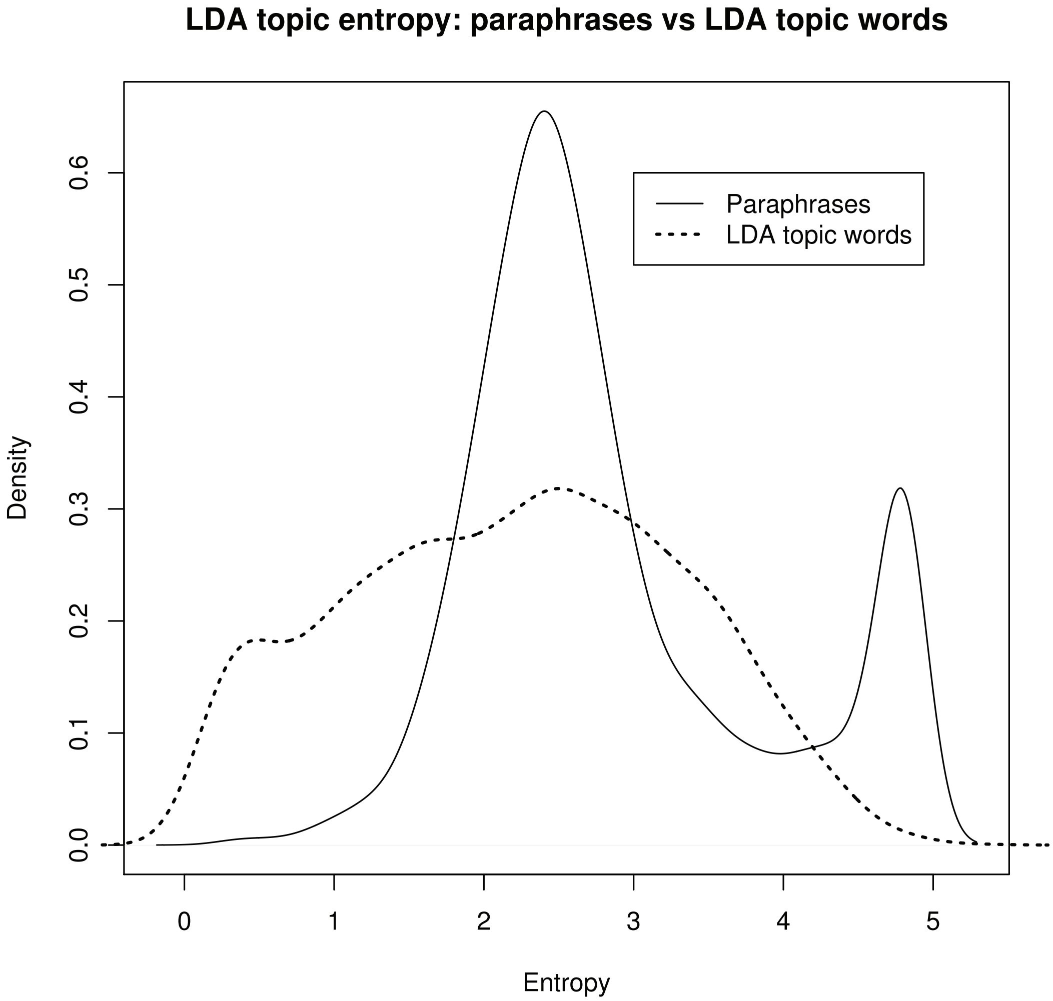


Figure 5.7: Entropy of LexSub paraphrases and top 30 words of all LDA topics.

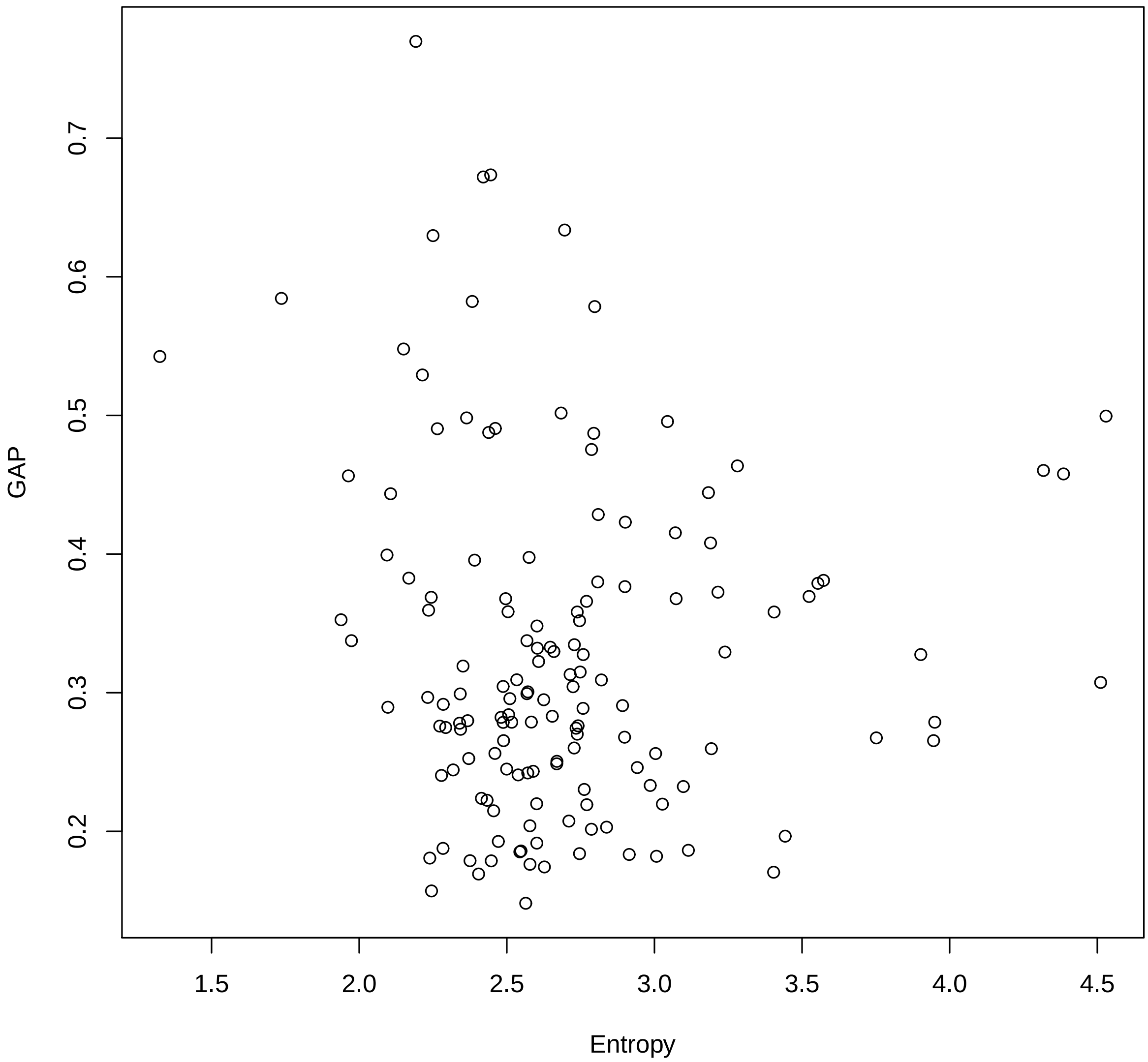


Figure 5.8: Plot of GAP score by average entropy of paraphrases for given target words. Correlation, while negative is insignificant with Spearman's $\rho = -0.04353575$ (p=0.6044)

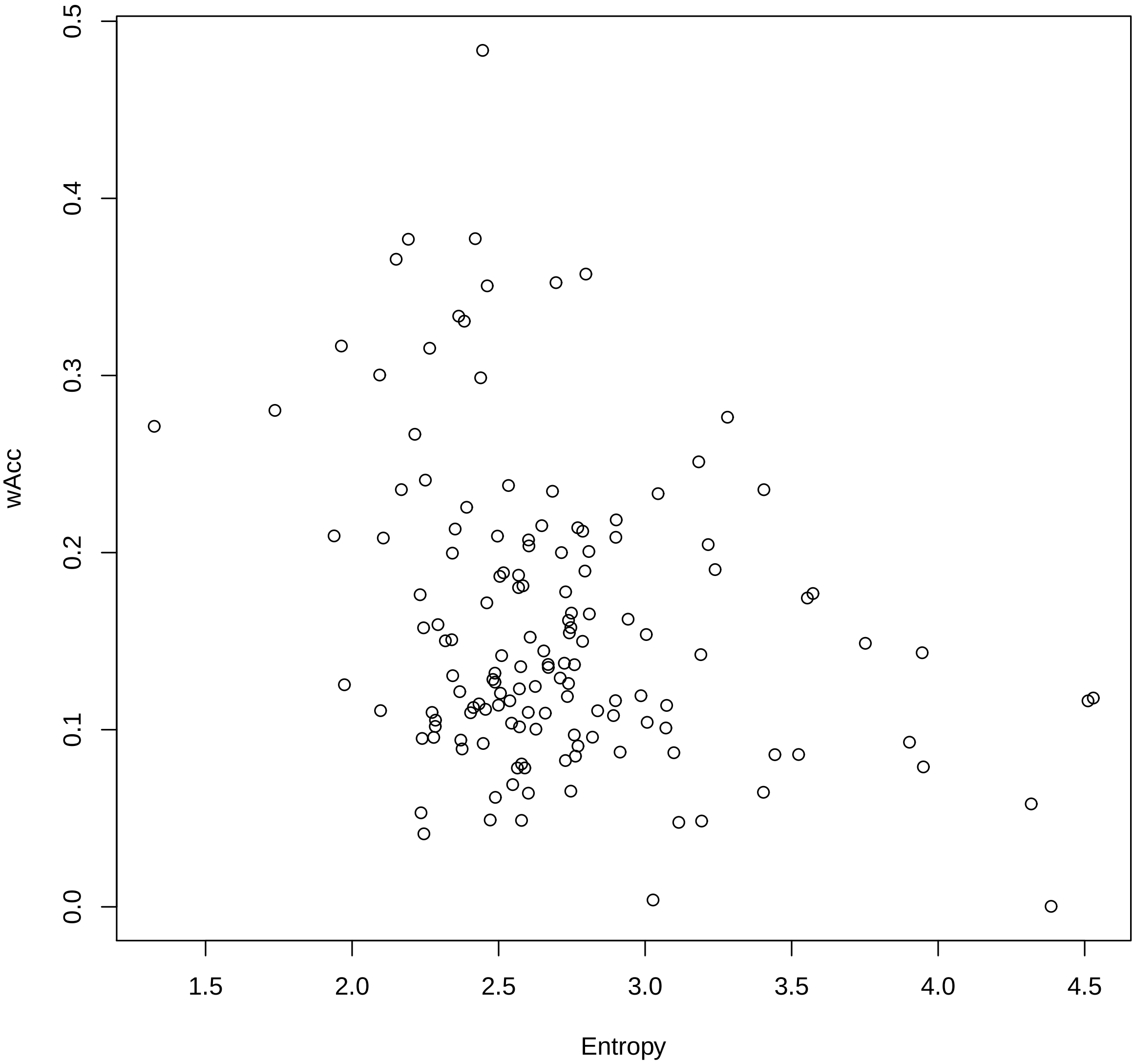


Figure 5.9: Plot of GAP score by average entropy of paraphrases for given target words. Correlation is negative and significant with Spearman's $\rho = -0.2317619 (p=0.005189)$

correlation is visible in both plots. Nonetheless, the correlation is only significant between wAcc and entropy. In related work [Kilgarriff and Rosenzweig, 2000], a strong correlation between entropy and performance in a pure WSD was noted. In fact, entropy was more indicative of the difficulty of a target word than the degree of polysemy for the target. Though there are non-trivial differences between the work in Kilgarriff and Rosenzweig and our model, we believe that a similar argument can be made in our case. The relatively higher entropy of the targets in LexSub compared to the entropy of highly topical words in the topic models makes an LDA based factor less than effective for LexSub.

To examine whether performance could be improved by varying the number of topics, we conducted experiments where we trained the topic models with 100, 200, 300, ..., 2000 topics. We then ran baseline models where only the document factor and vector space word factor were retained. The selectional factors were removed. These results are plotted in Figure 5.10 for GAP and Figure 5.11 for wAcc. As can be seen, there is no relation between the number of topics and performance.

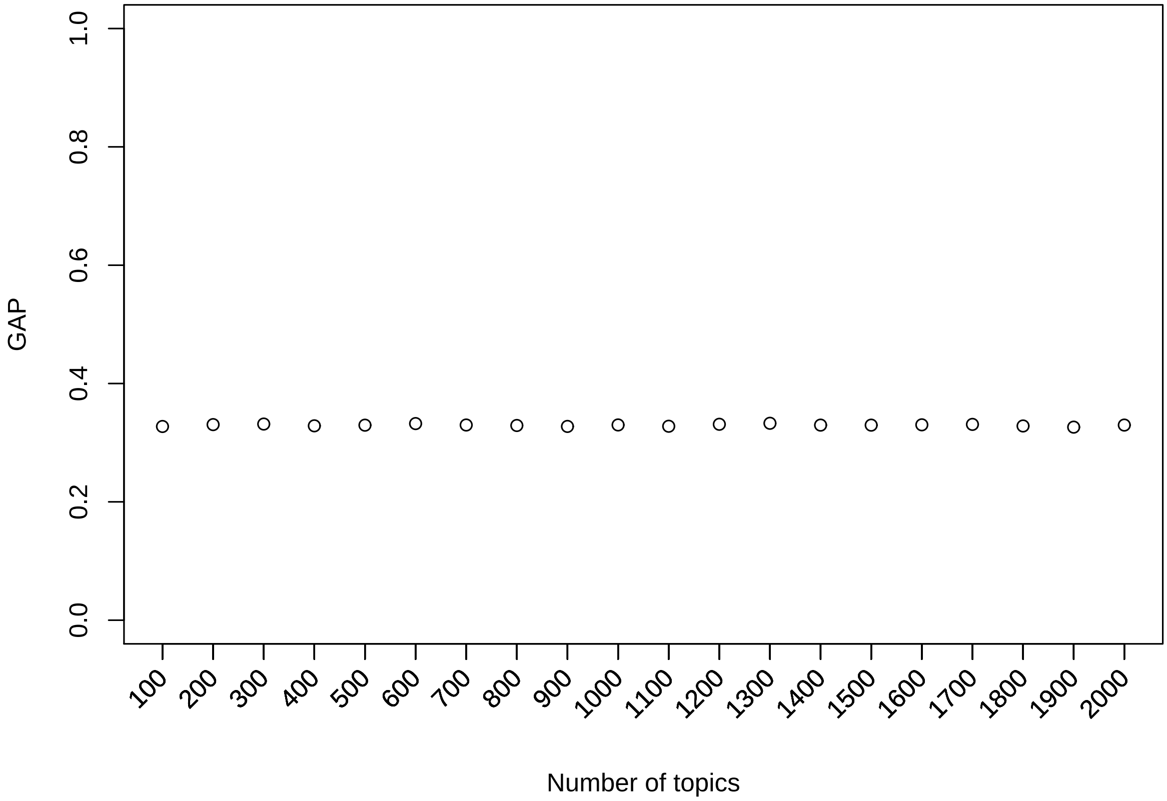


Figure 5.10: Plot of LDA baseline experiments where only number of topics is varied. x-axis is number of topics and y-axis is GAP score.

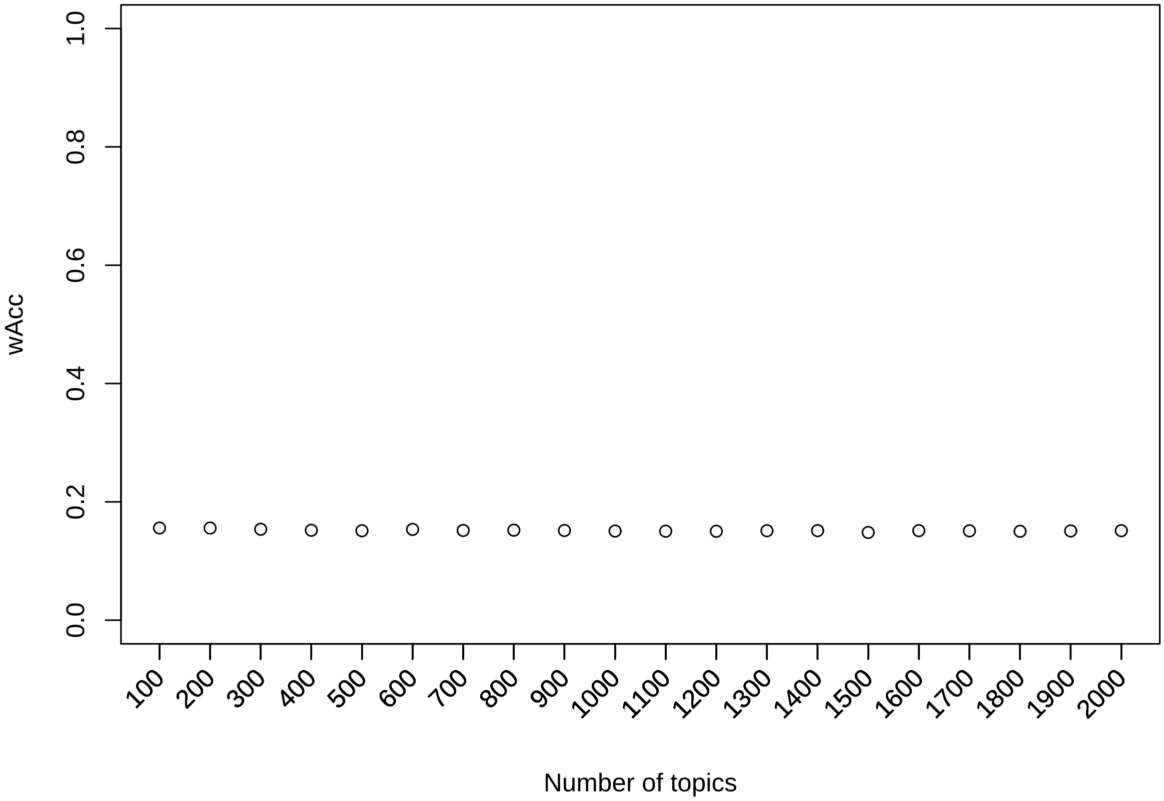


Figure 5.11: Plot of LDA baseline experiments where only number of topics is varied. x-axis is number of topics and y-axis is wAcc score

5.7 Bag-of-words sentence context

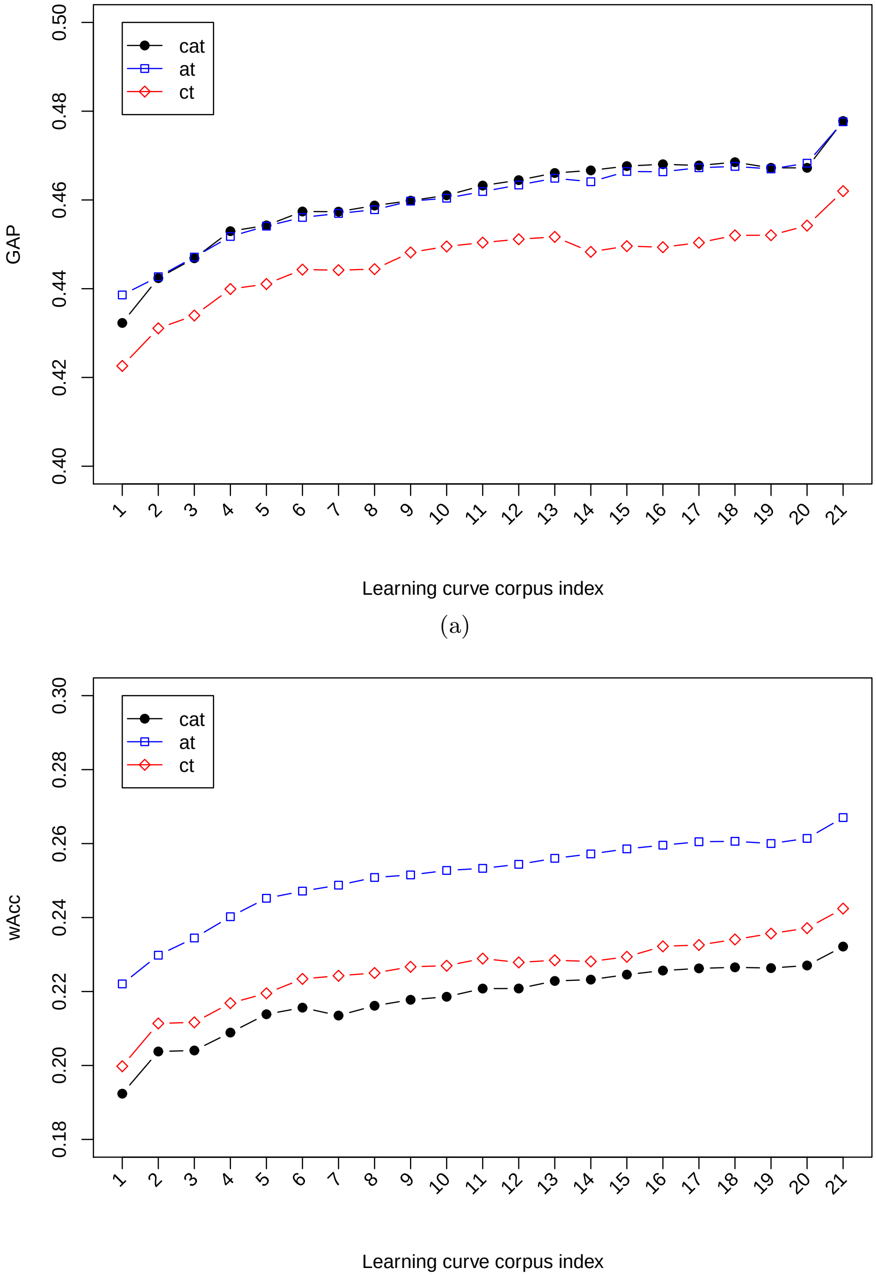
	GAP	wAcc
baseline	34.00	12.27
ct	36.89	12.30
at	38.19	13.60
cat	39.17	13.40

Table 5.7: Sentence bag-of-words factor experiment results

Sentence bag-of-words features were shown to be effective in Erk and Pado [2010] within a vector space approach. Here, we examined whether such features could be incorporated within our framework successfully as evidence. The results are tabulated in Table 5.7 and it is clear that they are not helpful. The baseline results are based on experiments using only the bag-of-words factors described in §5.7 and the vector space word factor in §3.3.3.1. The remaining ct, at, and cat results incorporate the previous bag-of-words factors and the vector space word factors and use epmi selectional factors. There is evident deterioration in performance across the board compared to the results in Table 5.1. The most severe decrease is in wAcc where the scores underperform even the random baseline. One point of interest in these results is that the cat result is one percentage point higher than at for GAP. This is the widest difference between cat and at experiments among all experiments, all other things being equal.

5.8 Learning curve experiments

It is clear from the preceding discussion that the selectional factor is the most important component in the performance of our model. As such, we examine it



(a)

(b)

Figure 5.12: Learning curve for GAP and wAcc by training corpus size for selectional factors. By **at**, **ct**, and **cat** condition. Tick marks on x-axis from 1 to 20 represent approximately 0.1 billion to 2 billion words of ukWaC. 21st tick mark represents combined corpus of ukWaC, bnc, and Giga.

	GAP		wAcc	
GT	10&20	20&21	10&20	20&21
at	0.79	**0.93**	0.87	0.56
ct	0.47	**0.78**	1.02	0.53
cat	0.62	**1.05**	0.85	0.51

Table 5.8: Gain in performance in terms of GAP and wAcc in learning curve experiments by transformation type (GT=graph transformation). The "10&20" header indicates absolute performance gain from corpus 10 to corpus 20. The "20&21" header indicates absolute performance gain from corpus 20 to 21.

further. In §5.4, we examined whether int or epmi is more effective in inferring graded word meaning. In addition to these two different transformations of maximum likelihood estimates, we can see from 5.1 that there are noticeable performance gains as we increase the size of our training corpus from BNC to GIGA to U+B+G. To investigate the effect of corpus size in more detail, we conducted learning curve experiments, where everything was held constant except the size of the training corpus. The word factor was defined as the vector space parameter in 3.3.3.1. No document or sentence level bag-of-words context was incorporated. Selectional factors used the epmi parameters. The results of these experiments are plotted for GAP and wAcc and for at, ct and cat in Figure 5.12. The tick marks on the x-axis from 1 to 20 indicate subdivisions of UKWAC. From 1 to 20, each represents approximately 0.1 billion to 2 billion words, the last of which is the full UKWAC corpus. Finally, the 21st corpus combines all of UKWAC, BNC, and GIGA; the last two add some 1.1 billion words for a total of 3.1 billion words. First, we can see that more data is better. The second thing we notice is that the jump from corpus 20 to 21—where GIGA and BNC are added—is considerable. In fact, the performance gain for GAP that comes from adding the last one billion words is actually greater than the gain that comes from adding one billion words of ukWaC to an existing one billion words of ukWaC—i.e. the jump from corpus 10 to 20. Though there is a similar performance jump in wAcc from corpus 20 to 21, the amount gained here does not exceed the change from 10 to 20. The results are tabulated in Table 5.8. The former result on GAP improvement shows that not all data is equal and strongly suggests that diversity of the domain is an important consideration as well when building training corpora.

		BNC		U+B+G	
GT	Model	GAP	wAcc	GAP	wAcc
cat	int	L**	L**	-	-
ct	int	-	-	-	-
cat	int+lda	L**	L**	-	-
ct	int+lda	L**	-	-	-
cat	epmi	G**	-	-	-
ct	epmi	G**	G**	G**	-
cat	epmi+lda	-	L**	-	-
ct	epmi+lda	-	G**	-	-

Table 5.9: Comparing global models to models restricted to local syntactic context: L=local model better, G=global model better. GT = graph transformation. **: difference significant at $p < 0.01$. Only results with performance distance ≥ 0.05

5.9 Nonlocal syntactic context

The ct and cat models receive information not only from their syntactic neighbors. They infer paraphrase distributions by marginalizing over connected paraphrase nodes, which eliminates d-separation[4] between any given paraphrase node and observed variables. This allows each paraphrase node to infer a para-

[4] For undirected graphs, two connected nodes X and Z are d-separated by node Y iff all paths between X and Z pass through Y. By marginalizing over Y, X and Z are no longer d-separated and thus any existing factorization properties for connected nodes X and Z no longer hold.

phrase distribution based on evidence from the entire parsed sentence as well as the hidden paraphrase nodes. Thus, from a linguistic modeling viewpoint, this is more satisfying than at, which only requires knowledge of its immediately adjacent, observed environment and does not incorporate its neighbors' paraphrase distributions.

This raises the question of whether the cat and ct variants successfully incorporate observed evidence from the entire sentence. We test this by using a pruned dependency graph consisting of only the LexSub target word and its immediate neighbors. We then transform this pruned dependency graph via cat or ct as before. In this **local** model, we still have mutual disambiguation between the target and its syntactic neighbors, but no influence from the wider syntactic context in the sentence.[5] We call the original graph the **global** model. We compare the local and global model by computing 99% confidence intervals with bootstrap resampling.

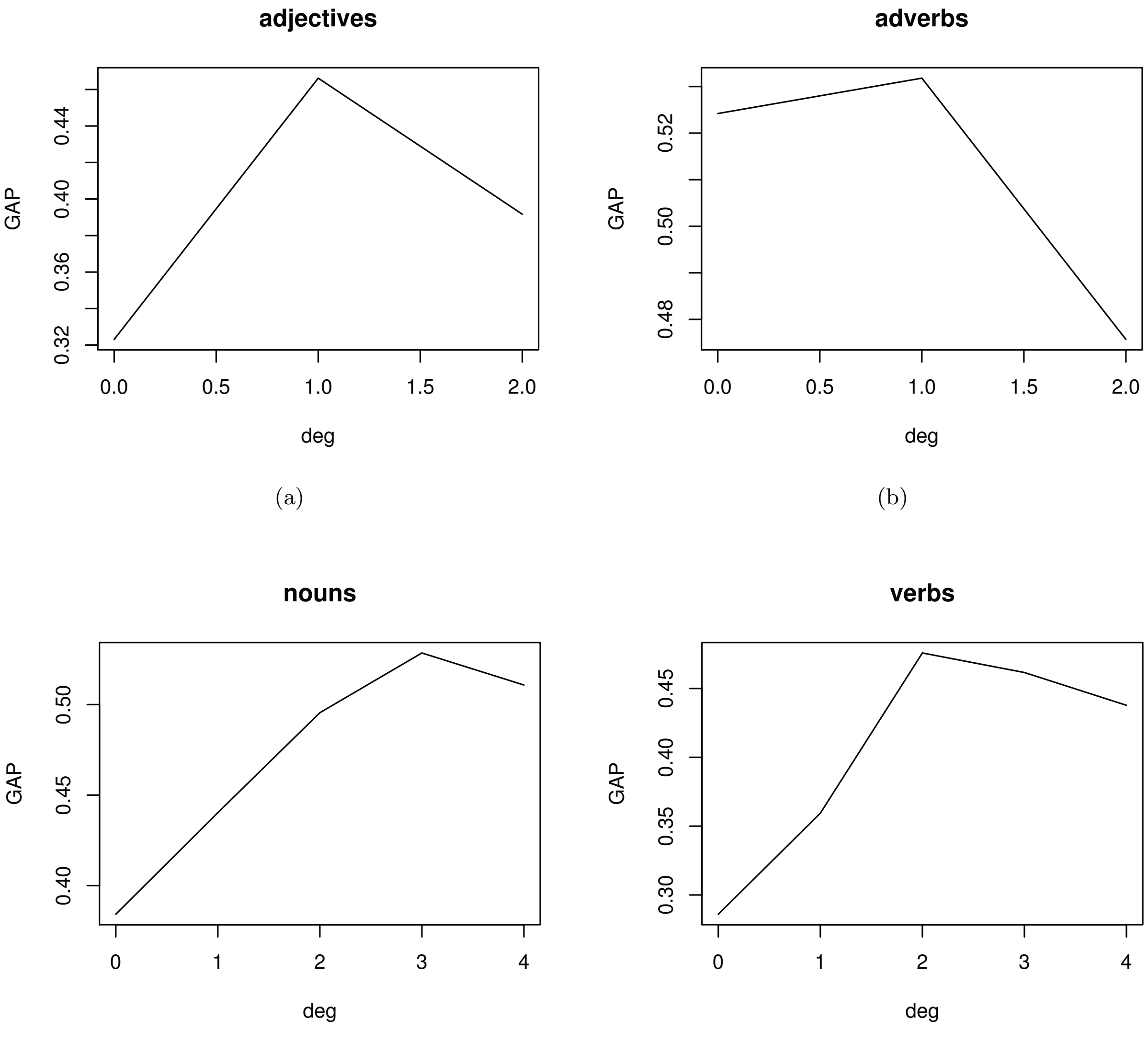
The results of these experiments are shown in Table 5.9: L are conditions where the local model is significantly better with an absolute difference in performance of ≥ 0.05. G are conditions where the global model is significantly better with the same minimum difference in performance. Looking at the BNC parameters, the local model shows better performance with int factors, while the global model works better for epmi factors. This could indicate that selectional factors computed from raw co-occurrence counts do not yield a clear enough signal for non-local syntactic context to be usable, while epmi-based selectional factors do. The local model

[5] Note that in this subgraph the LexSub target's neighbors are still contextualized, but are contextualized only by the target. This is in contrast to existing approaches like EP08 and TFP10, which always contextualize the target based on non-contextualized context vectors.

again achieves better scores for int combined with lda, so lda factors may yield some of the same information that the non-local syntactic context would have supplied. With epmi plus lda, we may be getting mixed signals, resulting in one case where the local model works better and one where the global model does. With parameters from U+B+G, the picture is very different, where with the exception of one condition—namely, GAP for epmi, where the global model outperforms the local model—there is no significant difference between the performance of the local and global models. We hypothesize that any differentiating factors that come from the local vs. global topology of the graphs are overridden by the size of the corpora from which the parameters are derived.

5.10 Number of syntactic neighbors

Next we examine whether the model is able to successfully integrate contextualizing information from multiple syntactic neighbors. Erk and Padó [2009] found that use of multiple syntactic neighbors did not improve the EP08 model. We find that this is different in the pd model. Figures 5.13 through 5.14 plot performance against the degree of the LexSub target node for different parts of speech, based on parameters from U+B+G with the cat+epmi condition. Other conditions show a similar picture. The x-axes indicate the degree of the target node in the trimmed dependency parse with only context words. The y-axes show either GAP or wAcc. We can see that for verbs and nouns, there is a large increase in GAP for nodes with at least two neighbors. For wAcc, the most pronounced increase is from zero to one neighbors, with a small increase for two neighbors, again underscoring the



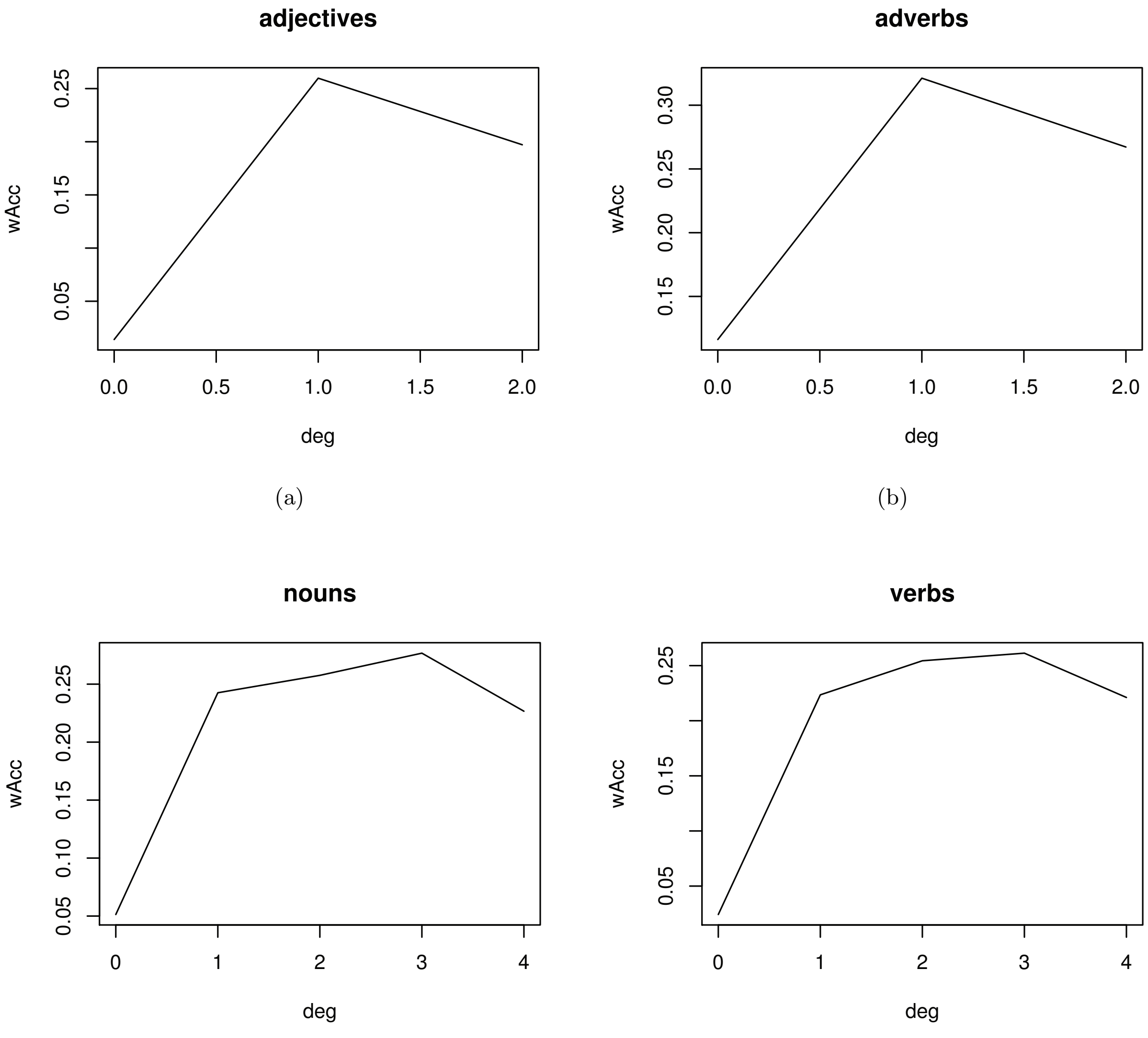
(a)

(b)

(c)

(d)

Figure 5.13: GAP by degree of target node in graph. By part-of-speech.



(a)

(b)

(c)

(d)

Figure 5.14: wAcc by degree of target node in graph. By part-of-speech.

difficulty of achieving an improvement in weighted accuracy. In contrast, the model performs best for adjectives and adverbs when there is exactly one neighbor, which makes sense, as they typically have fewer dependency neighbors. We conclude that having multiple neighbors helps our model in terms of GAP for verbs and nouns, and doesn't harm it in terms of wAcc.

5.11 Experiments with nameless hidden nodes

	GAP	wAcc
at	28.39	20.32
cat	28.93	20.37
ct	29.23	20.30

Table 5.10: Experiment results on model with nameless hidden nodes

In this section, we present the results of the model with nameless hidden nodes (nh). The training of the model was described in §3.3.3.2 and its evaluation was described in §4.6.5. The models were all trained on the 1.15M word Brown corpus [Francis et al., 1982]. 50 states were posited for all models. All hyperparameters $\alpha = \beta = \gamma = \delta$ were set to 5. Parameters were determined from 1000 samples that were collected with a lag of 5 iterations after a burn-in period of 500 iterations. Once the parameters had been learned, inference was conducted identically to the named models.

The results from these experiments are tabulated in Table 5.10. As can be seen, the results are worse than the random baseline for GAP and wAcc. One interesting result is that both ct and cat outperform at (with at least $p < 0.05$) though the difference between ct and cat itself is not significant.

For comparison's sake, we conducted the standard pd variant of at+epmi with meaningful paraphrases values. The parameters were derived from Brown. The results were 38.51 (GAP) and 18.22 (wAcc). So there is a substantial 10 point gain in terms of GAP when actual paraphrases are used as values for the paraphrase nodes. More surprising is that nh displays a 2 point gain over pd in terms of wAcc. For the moment, we can only say that we will conduct further investigations into the issue.

5.12 Miscellaneous experiments on LexSub

Here, we describe additional experiments that we conducted but are not critical for the overall evaluation of the model.

5.12.1 Blocked information flow in paraphrase nodes

	GAP	wAcc
at	47.78	26.70
cat	47.77	23.22
ct	46.13	24.24

Table 5.11: Augmented paraphrase set experiment results

When we create a paraphrase node for a given observation, the valid paraphrase values over which the distribution can have non-zero values is restricted by a predefined set that derive from LexSub and WordNet. However, because we are defining such paraphrase nodes for every content word node in a sentence, the coverage is generally insufficient for most words and therefore when a word does not have an entry in either database, we take the stopgap measure of only allowing one valid paraphrase value for such words, namely its own self. This has the unfortunate result of "blocking" information flow between paraphrase nodes that are *d*-separated. To overcome this limitation, we tried augmenting the non-zero values for paraphrase nodes given some target word. This was done by taking some frequent items that shared the same part-of-speech with the target and adding them to set of paraphrases in addition to the synonyms in WordNet and the paraphrases from LexSub. Specifically, we conducted unigram counts over u+b+g, then for all word types in a given part-of-speech, we took the 200 types which had frequency rank 2001 to 2200 and added them to existing sets of paraphrases if there was only one valid paraphrase for some target with a given part-of-speech. The results are tabulated in Table 5.11. There is a numerical, but not significant, increase in performance for cat.

5.12.2 Retention of function words

	GAP	wAcc
cat	47.72	26.54
at	47.65	24.12
ct	46.03	23.03

Table 5.12: Experiment results for when function words have not been discarded

The current set of models remove non-content words from the dependency graph. While justified from the viewpoint of practice and practicality, we wondered whether there is information loss and examined alternatives that retain non-content words. Specifically, all function words that were not prepositions attached to *mod* relations were retained. Prepositions attached to *mod* relations were transformed as with all other models examined in this dissertation.[6] Part of the motivation for this investigation is that one of the relevant variables in distinguishing between the *address* in Example 1.5 and the *address* in Example 1.6 is the existence of a determiner that specifies the former *address*. The results of these experiments are tabulated in 5.12. This time, there is a slight numerical deterioration in performance compared to 5.11 and 5.1 but the differences were not significant.

5.13 twsi dataset

It is to be expected that the twsi dataset will be harder to model than LexSub. It focuses on the most frequent nouns, which will in general be harder to contextualize than medium-frequency lemmas. However, we consider it important to access additional datasets for the evaluation of usage models to avoid overfitting the LexSub data. Table 5.13 shows GAP and wAcc scores on twsi by selectional factor type, graph transformation, and parameter source. Table 5.14 shows baseline results. The parameters are based on counts from ukWaC and bnc. Giga was

[6] For example, the dependency parse output of the C&C parser [Clark and Curran, 2007] on the noun phrase "measure of height" would be *measure* $\xrightarrow{\text{ncmod}}$ *of* $\xrightarrow{\text{dobj}}$ *height*. As input to our current models, we remove the non-content word *of* and transform this graph to *measure* $\xrightarrow{\text{mod-of}}$ *height*

	U+B				BNC			
	int		epmi		int		epmi	
GT	GAP	wAcc	GAP	wAcc	GAP	wAcc	GAP	wAcc
ct	32.48	19.18	33.26	16.67	32.62	19.24	32.53	13.88
at	33.06	**19.88**	34.01	19.45	33.04	**19.85**	33.01	16.03
cat	32.88	16.40	**34.31**	15.22	32.80	16.26	**33.08**	12.76

Table 5.13: twsi data: GAP and wAcc scores by corpus, graph transformation and factor type. Experiment parameters are derived from counts in ukWaC and bnc

	GAP	wAcc
seq	32.48	16.24
singl	33.96	19.72
rand	23.62	18.57

Table 5.14: TWSI data: GAP and wAcc scores for baselines. seq uses epmi. Experiment parameters are derived from counts in UKWAC and BNC

excluded. The scores confirm that this dataset is relatively hard to model. The random baseline in Table 5.14 is considerably lower than for LEXSUB, indicating that there is on average a greater number of paraphrase candidates for TWSI datapoints. The best pd model variants improve over the random baseline by about 11 points in GAP, and 1 point in wAcc, while the improvement on the LEXSUB dataset is 17 points for GAP and 5 points for wAcc. The singleton baseline is exceptionally strong: The difference in GAP between that baseline and the best model variant is only 0.3 points, and the difference in wAcc is negligible. So on this dataset, context-aware models barely manage to rise above a model that ranks paraphrases just by similarity to the target without taking sentential context into account. Focusing on the left side of Table 5.13, the results confirm the trend in the LEXSUB experiments where cat and at models perform the strongest on GAP, while at models show the best performance in terms of wAcc. However, on this dataset, cat actually outperforms at in terms of GAP. What is surprising is that for wAcc, the best factor type is not epmi but int for both U+B and BNC parameters.

SF	U+B+G			BNC		
	ct	at	cat	ct	at	cat
int	0.193	0.207	0.195	14.10	14.93	14.55
epmi	0.305	**0.311**	**0.311**	13.82	19.82	19.14

M/L	0.24
E&P	0.27

Table 5.15: M/L data: Spearman's ρ. pd parameters estimated using U+B+G. ρ for prior M/L and EP08 models on right. All results significant at $p < 0.01$.

5.14 M/L dataset

Table 5.15 shows the results on the M/L dataset. Mitchell and Lapata estimate the ceiling (inter-rater agreement) at $\rho = 0.4$. The correlation of the best pd model condition, at with epmi parameters, at 0.311 exceeds the best results reported by M/L and EP08. However, these results are based on the U+B+G corpus, while both M/L and EP08 use the BNC, Using BNC parameters, the pd model's performance is lower than those of M/L and EP08, maybe due to the impoverished syntactic context of the M/L datapoints. Even though the dataset is different and the evaluation measure is different, the results mainly confirm our findings on LexSub: using epmi transformation on the selectional factor parameters strongly improves performance, and the at condition results in the best performance.

Chapter 6

Conclusion

In this dissertation, we have introduced a usage model of word meaning that is inference-based, characterizing a word's meaning within context as a distribution over potential paraphrases. The main aim has been to create a model that is general and flexible enough for testing and integrating multiple knowledge sources for the task of contextualization. The model framework of probabilistic graphical models is itself dependent upon a novel choice of representation: graded word sense over paraphrases. We normalized this to create paraphrase distributions. This representation granted us considerable flexibility in capturing word meaning as well as more cognitive validity [Erk and McCarthy, 2009]. Furthermore, though it does not apply to the current work, the creation of the main data set we used—LexSub—required far less lexical expertise than for a lexical inventory such as WordNet. The lower threshold allows and will allow convenient creation or expansion of such inventories such as LexSub. Given these positives, we believe that graded word sense is a meaningful and lasting contribution to the study of lexical semantics.

We summarize our findings in a format that mirrors the list of questions in the introduction:

- **Influence of collocational information**

Modeling collocation in the form of edges between paraphrase nodes and adjacent observations is significantly more useful than solely having edges between paraphrase nodes

- **Nonlocal syntactic context**

  Model variants that laid edges between paraphrase nodes implicitly allow information to flow throughout an entire sentence. The conclusion is that this information flow is not harmful in terms of GAP as long as local, collocational information is also considered.

- **Non-syntactic bag-of-words context**

  Variants that incorporated bag-of-words context at the document level through topic models or at the sentence level through complete graphs severely underperformed models that did not incorporate this information.

- **Effects of parameterization**

  Comparing parameters based on normalized frequency counts and epmi transformed parameters, it is clear that the latter performs much better.

- **Type of hidden nodes**

  In terms of the value space over the paraphrase nodes, actual paraphrases are highly more beneficial compared to nameless indexes.

Happily, it was found that many variants of our core model outperformed the state-of-the-art model [Thater et al., 2010] on the LexSub task for all parts-of-speech except verbs (i.e. nouns, adjectives and adverbs).

Beyond the above summarization, the fact that the adjacency transform (at) variant with exponentiated pointwise mutual information (epmi) performed best in terms of GAP and wAcc on LexSub strongly indicates that (1) the choice of lexical targets in LexSub is skewed towards words which have high correlation with words which occur in dependency adjacent nodes (2) incorporating all the arbitrary number of observed neighbors around a target through simple product rules is much more effective than most vector space models which can only consider a fixed number of neighbors (usually one or two). Further experiments are required to find whether similar performance gains would hold under a different data set.

6.1 Future work

In this section, we discuss some of the deficiencies of the current work and what must be further studied to develop a more complete picture of the model and validate its potential.

6.1.1 Automatic extraction of paraphrase sets

For reasons of tractability, the set of real word paraphrases that have non-zero values for a given target varies. For *charge* this will include *accusation*, *allegation* and a few more words. For *bright*, this would be *light*, *promising* and a few more. Compared to the overall vocabulary, these are much smaller subsets. And we obtained these subsets from WordNet and the LexSub corpus. To reduce the level of supervision required even further, removing this reliance on previously compiled sense inventories is critical. We believe this to be an important issue and plan on dealing with it in the long term. One possible solution is to evaluate the pairwise similarity between words in some vector space and only consider words which exceed some similarity threshold in relation to some other word to be valid paraphrases of this word.

6.1.2 Applications

The experiments in the dissertation concentrated on data sets that evaluated either graded word sense—LexSub [McCarthy and Navigli, 2009] and twsi [Bieman and Nygaard, 2010]—or selectional preference—M/L [Mitchell and Lapata, 2008]. While important for examining the properties and capabilities of our model in itself in relation to other existing models, and while it is important for examining the phenomenon and representation of graded word sense, there is a pressing need to investigate its performance in terms of real-world applications such as information retrieval or question answering. One fruitful application we have in mind is in machine translation. For example, language models used to generate a target sentence could be strengthened by incorporating information from the adjacency transformed version of our model.

6.1.3 More flexible notions of evidence

It is clear from the results here (§5.6) as well as elsewhere [Leacock et al., 1998] that different types of words are dependent on different sources of evidence for disambiguation. For example, our experiments on incorporating document topic model as evidence suggest that the target words in LexSub are poorly suited for inferring the meanings of words with low topicality. In contrast, topic models can be used profitably in word sense disambiguation given the right set of words [Boyd-Graber et al., 2007].

An effective model should be able to distinguish which pieces of evidence are relevant or irrelevant and effectively promote or downgrade such evidence when inferring meaning. Our hope was that the process of factorization and marginalization in a graphical model would be able to implicitly conduct such promotion and demotion of evidence. Our experiments show that, for the data we have and under the parameterizations and estimation procedures that we used, this is not the case. We will require more experiments under more diverse settings to conclusively decide whether a new model is necessary and how this new model may be defined if this is the case.

6.1.4 Further exploration of model with nameless hidden nodes

The results presented in §5.11—based on the model with a nameless set of indexes as its value space described in §3.3.3.2—are clearly of a preliminary nature. Different hyperparameter values as well as different model sizes (i.e. the number of nameless hidden states K) need to be examined extensively with larger corpora. The restriction of the size of the hidden states to $K = 50$ states was due to memory issues. For example, experiments with $K = 100$ proved untenable on machines with 50G of memory. Once solutions have been devised and more experiments have been conducted, we will be able to make more conclusive statements regarding the performance of these models.

Bibliography

Eneko Agirre and Philip Edmonds. Introduction. In Eneko Agirre and Philip Edmonds, editors, *Word Sense Disambiguation: Algorithms and Applications*, pages 1–28. 2006.

Eneko Agirre, Lluís Màrquez, and Richard Wicentowski, editors. *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007).* Association for Computational Linguistics, Prague, Czech Republic, June 2007. URL `http://www.aclweb.org/anthology/S/S07/S07-1`.

Eneko Agirre, Oier López de Lacalle, Christiane Fellbaum, Shu-Kai Hsieh, Maurizio Tesconi, Monica Monachini, Piek Vossen, and Roxanne Segers. Semeval-2010 task 17: All-words word sense disambiguation on a specific domain. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 75–80, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/S10-1013`.

S.M. Aji and R.J. McEliece. The generalized distributive law. *Information Theory, IEEE Transactions on*, 46(2):325–343, mar. 2000. ISSN 0018-9448. doi: 10.1109/18.825794.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The berkeley framenet project. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguis-*

tics - Volume 1, ACL '98, pages 86–90, Stroudsburg, PA, USA, 1998. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/980845.980860. URL `http://dx.doi.org/10.3115/980845.980860`.

Colin Bannard and Chris Callison-Burch. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 597–604, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/1219840.1219914. URL `http://dx.doi.org/10.3115/1219840.1219914`.

Yehoshua Bar-Hillel. The present status of automatic translation of languages. volume 1 of *Advances in Computers*, pages 91 – 163. Elsevier, 1960. doi: DOI:10.1016/S0065-2458(08)60607-5. URL `http://www.sciencedirect.com/science/article/pii/S0065245808606075`.

M. Baroni, S. Bernardini, A. Ferraresi, and E. Zanchetta. The wacky wide web: A collection of very large linguistically processed web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226, 2009.

Marco Baroni and Roberto Zamparelli. 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*, pages 1183–1193, Cambridge, MA, October 2010. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/D10-1115`.

Regina Barzilay and Kathleen R. McKeown. Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting on Association for Compu-*

tational Linguistics, ACL '01, pages 50–57, Stroudsburg, PA, USA, 2001. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/1073012.1073020. URL `http://dx.doi.org/10.3115/1073012.1073020`.

Matthew J. Beal. *Variational algorithms for approximate Bayesian inference*. PhD thesis, University College London, 2003.

Jonathan Berant, Ido Dagan, and Jacob Goldberger. Global learning of focused entailment graphs. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1220–1229, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P10-1124`.

Chris Bieman and Valerie Nygaard. Crowdsourcing wordnet. In *Proceedings of the 5th Global WordNet conference*, Mumbai, India, 2010.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006. ISBN 0387310738.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003. ISSN 1532-4435. doi: http://dx.doi.org/10.1162/jmlr.2003.3.4-5.993. URL `http://dx.doi.org/10.1162/jmlr.2003.3.4-5.993`.

Jordan Boyd-Graber, David Blei, and Xiaojin Zhu. A topic model for word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Em-*

pirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), pages 1024–1033, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D/D07/D07-1109.

T. Brants and A. Franz. Google web 1t 5-gram corpus, version 1. Technical report, Linguistic Data Consortium, Philadelphia, 2006. Catalog Number LDC2006T13.

Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(1-7):107 – 117, 1998. ISSN 0169-7552. doi: DOI:10.1016/S0169-7552(98)00110-X. Proceedings of the Seventh International World Wide Web Conference.

S. Brody and M. Lapata. Bayesian word sense induction. In *Proceedings of EACL*, Athens, Greece, 2009.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. The mathematics of statistical machine translation: parameter estimation. *Comput. Linguist.*, 19:263–311, June 1993. ISSN 0891-2017. URL http://portal.acm.org/citation.cfm?id=972470.972474.

Rebecca Bruce and Janyce Wiebe. Word-sense disambiguation using decomposable models. In *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*, ACL '94, pages 139–146, Stroudsburg, PA, USA, 1994. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/981732.981752. URL http://dx.doi.org/10.3115/981732.981752.

Marine Carpuat and Dekai Wu. Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 61–72, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/D/D07/D07-1007`.

Yee Seng Chan, Hwee Tou Ng, and David Chiang. Word sense disambiguation improves statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 33–40, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P07-1005`.

Jinying Chen and Martha Palmer. Improving english verb sense disambiguation performance with linguistically motivated features and clear sense distinction boundaries. *Language Resources and Evaluation*, 43:181–208, 2009. ISSN 1574-020X. URL `http://dx.doi.org/10.1007/s10579-009-9085-0`. 10.1007/s10579-009-9085-0.

Timothy Chklovski and Patrick Pantel. Verbocean: Mining the web for fine-grained semantic verb relations. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 33–40, Barcelona, Spain, July 2004. Association for Computational Linguistics.

Stephen Clark and James R. Curran. Wide-coverage efficient statistical parsing with ccg and log-linear models. *Comput. Linguist.*, 33:493–552, December

2007. ISSN 0891-2017. doi: http://dx.doi.org/10.1162/coli.2007.33.4.493. URL http://dx.doi.org/10.1162/coli.2007.33.4.493.

Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley & Sons, Inc., 2005. ISBN 9780471748823.

Jim Cowie, Joe Guthrie, and Louise Guthrie. Lexical disambiguation using simulated annealing. In *Proceedings of the 14th conference on Computational linguistics - Volume 1*, COLING '92, pages 359–365, Stroudsburg, PA, USA, 1992. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/992066.992125. URL http://dx.doi.org/10.3115/992066.992125.

F.G. Cozman. Generalizing variable elimination in bayesian networks. In *Proceedings of the IBERAMIA Workshop on Probabilistic Reasoning in Artificial Intelligence*, Sao Paulo, Brazil, 2000.

D. A. Cruse. Polysemy and related phenomena from a cognitive linguistic viewpoint. In P. Saint-Dizier and E. Viegas, editors, *Computational Lexical Semantics*, pages 33–49. Cambridge University Press, 1995.

Koen Deschacht and Marie-Francine Moens. Semi-supervised semantic role labeling using the Latent Words Language Model. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 21–29, Singapore, August 2009. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D/D09/D09-1003.

Markus Dreyer and Jason Eisner. Graphical models over multiple strings.

In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 101–110, Singapore, August 2009. URL http://cs.jhu.edu/~jason/papers/#emnlp09-multimorph.

P. Edmonds and S. Cotton, editors. *Proceedings of the SensEval-2 Workshop*, Toulouse, France, 2001. ACL.

Katrin Erk. Representing words as regions in vector space. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 57–65, Boulder, Colorado, June 2009. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W09-1109.

Katrin Erk. What is word meaning, really? (and how can distributional models help us describe it?). In *Proceedings of the 2010 Workshop on GEometrical Models of Natural Language Semantics*, pages 17–26, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/W10-2803.

Katrin Erk and Diana McCarthy. Graded word sense assignment. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 440–449, Singapore, August 2009. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D/D09/D09-1046.

Katrin Erk and Sebastian Padó. Towards a computational model of gradience in word sense. In *Proceedings of the Seventh International Workshop on Computational Semantics*, 2007.

Katrin Erk and Sebastian Padó. A structured vector space model for word meaning in context. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 897–906, Honolulu, Hawaii, October 2008. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/D08-1094`.

Katrin Erk and Sebastian Padó. Paraphrase assessment in structured vector space: Exploring parameters and datasets. In *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*, pages 57–65, Athens, Greece, March 2009. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/W09-0208`.

Katrin Erk and Sebastian Pado. Exemplar-based models for word meaning in context. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 92–97, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P10-2017`.

Katrin Erk and Carlo Strapparava, editors. *Proceedings of the 5th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, Uppsala, Sweden, July 2010. URL `http://www.aclweb.org/anthology/S10-1`.

Katrin Erk, Diana McCarthy, and Nicholas Gaylord. Investigations on word senses and word usages. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 10–18, Suntec, Singapore, August 2009. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P/P09/P09-1002.

C. Fellbaum. *WordNet: An electronic lexical database*. The MIT press, 1998.

W.N. Francis, H. Kučera, and A.W. Mackie. *Frequency analysis of English usage: Lexicon and grammar*. Houghton Mifflin, Boston, 1982.

G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. The vocabulary problem in human-system communication. *Commun. ACM*, 30:964–971, November 1987. ISSN 0001-0782. doi: http://doi.acm.org/10.1145/32206.32212. URL http://doi.acm.org/10.1145/32206.32212.

William Gale, Kenneth Church, and David Yarowsky. A method for disambiguating word senses in a large corpus. *Computers and the Humanities*, 26:415–439, 1992. ISSN 0010-4817. URL http://dx.doi.org/10.1007/BF00136984. 10.1007/BF00136984.

Claudio Giuliano, Alfio Gliozzo, and Carlo Strapparava. Fbk-irst: Lexical substitution task exploiting domain and syntagmatic coherence. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 145–148, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/S/S07/S07-1029.

Edward Grefenstette, Mehrnoosh Sadrzadeh, Stephen Clark, Bob Coecke, and Stephen Pulman. Concrete compositional sentence spaces. In *Proceedings of IWCS-9*, Oxford, UK, 2011.

Patrick Hanks. Do word meanings exist? *Computers and the Humanities*, 34:205–215, 2000. ISSN 0010-4817. URL `http://dx.doi.org/10.1023/A:1002471322828`. 10.1023/A:1002471322828.

Samer Hassan, Andras Csomai, Carmen Banea, Ravi Sinha, and Rada Mihalcea. Unt: Subfinder: Combining knowledge sources for automatic lexical substitution. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 410–413, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/S/S07/S07-1091`.

Michael I. Jordan. Graphical models. *Statistical Science*, 19(1):pp. 140–155, 2004. ISSN 08834237. URL `http://www.jstor.org/stable/4144379`.

Adam Kilgariff and Martha Palmer, editors. *Proceedings of the Pilot SensEval*. Association for Computational Linguistics, Hermonceux Castle, Sussex, UK, September 1998. URL `http://www.aclweb.org/anthology/S98-1`.

A. Kilgarriff and M. Palmer. Introduction to the special issue on senseval. *Computers and the Humanities*, 34:1–13, 2000. ISSN 0010-4817. URL `http://dx.doi.org/10.1023/A:1002619001915`. 10.1023/A:1002619001915.

A. Kilgarriff and J. Rosenzweig. Framework and results for english senseval. *Computers and the Humanities*, 34:15–48, 2000. ISSN 0010-4817. URL `http://dx.doi.org/10.1023/A:1002693207386`. 10.1023/A:1002693207386.

Adam Kilgarriff. "I don't believe in word senses". *Computers* *and the Humanities*, 31:91–113, 1997. ISSN 0010-4817. URL http://dx.doi.org/10.1023/A:1000583911091. 10.1023/A:1000583911091.

Adam Kilgarriff. 95% replicability for manual word sense tagging. In *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, EACL '99, pages 277–278, Stroudsburg, PA, USA, 1999. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/977035.977084. URL http://dx.doi.org/10.3115/977035.977084.

W. Kintsch. Meaning in context. In T.K. Landauer, D. McNamara, S. Dennis, and W. Kintsch, editors, *Handbook of Latent Semantic Analysis*, pages 89–105. Erlbaum, Mahwah, NJ, 2007.

Walter Kintsch. Predication. *Cognitive Science*, 25(2):173 – 202, 2001. ISSN 0364-0213. doi: DOI:10.1016/S0364-0213(01)00034-9.

F.R. Kschischang, B.J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *Information Theory, IEEE Transactions on*, 47(2):498–519, feb. 2001. ISSN 0018-9448. doi: 10.1109/18.910572.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1-55860-778-1. URL http://portal.acm.org/citation.cfm?id=645530.655813.

Thomas K. Landauer and Susan T. Dumais. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104(2):211–240, 1997. ISSN 0033-295X.

Shari Landes, Claudia Leacock, and Randee I. Tengi. Building Semantic Concordances. In Christiane Fellbaum, editor, *WordNet: An Electronic Lexical Database*. The MIT Press, 1998.

Claudia Leacock, George A. Miller, and Martin Chodorow. Using corpus statistics and wordnet relations for sense identification. *Comput. Linguist.*, 24:147–165, March 1998. ISSN 0891-2017. URL `http://portal.acm.org/citation.cfm?id=972719.972726`.

Michael Lesk. Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th annual international conference on Systems documentation*, SIGDOC '86, pages 24–26, New York, NY, USA, 1986. ACM. ISBN 0-89791-224-1. doi: http://doi.acm.org/10.1145/318723.318728. URL `http://doi.acm.org/10.1145/318723.318728`.

Dekang Lin and Patrick Pantel. DIRT - Discovery of Inference Rules from Text. In *Proceedings of ACM Conference on Knowledge Discovery and Data Mining (KDD-01)*, pages 323–328, San Francisco, CA, 2001.

Ken Litkowski. Senseval-3 task: Automatic labeling of semantic roles. In Rada Mihalcea and Phil Edmonds, editors, *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 9–12, Barcelona, Spain, July 2004. Association for Computational Linguistics.

David J. C. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003. ISBN 0521642981. URL http://www.inference.phy.cam.ac.uk/mackay/Book.html.

Margaret Masterman. In *International Conference on Machine Translation of Languages and Applied Language Analysis*, pages 437–475, 1961. URL www.mt-archive.info/NPL-1961-Masterman.pdf.

A. K. McCallum. MALLET: A Machine Learning for Language Toolkit., 2002. http://mallet.cs.umass.edu.

D. McCarthy and R. Navigli. The English lexical substitution task. *Language Resources and Evaluation*, 43(2):139–159, 2009. Special Issue on Computational Semantic Analysis of Language: SemEval-2007 and Beyond.

Diana McCarthy and Roberto Navigli. Semeval-2007 task 10: English lexical substitution task. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 48–53, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/S/S07/S07-1009.

Susan W. McRoy. Using multiple knowledge sources for word sense discrimination. *Comput. Linguist.*, 18:1–30, March 1992. ISSN 0891-2017. URL http://portal.acm.org/citation.cfm?id=146680.146683.

Rada Mihalcea and Phil Edmonds, editors. *Proceedings of the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text.*

Association for Computational Linguistics, Barcelona, Spain, July 2004. URL `http://www.senseval.org/senseval3`.

Rada Mihalcea, Timothy Chklovski, and Adam Kilgarriff. The senseval-3 english lexical sample task. In Rada Mihalcea and Phil Edmonds, editors, *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 25–28, Barcelona, Spain, July 2004. Association for Computational Linguistics.

George A. Miller. Wordnet: a lexical database for english. *Commun. ACM*, 38:39–41, November 1995. ISSN 0001-0782. doi: http://doi.acm.org/10.1145/219717.219748. URL `http://doi.acm.org/10.1145/219717.219748`.

Jeff Mitchell and Mirella Lapata. Vector-based models of semantic composition. In *Proceedings of ACL-08: HLT*, pages 236–244, Columbus, Ohio, June 2008. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P/P08/P08-1028`.

Jeff Mitchell and Mirella Lapata. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429, 2010.

Taesun Moon and Katrin Erk. An inference-based model of word meaning in context as a paraphrase distribution. submitted.

Taesun Moon, Katrin Erk, and Jason Baldridge. Crouching dirichlet, hidden markov model: Unsupervised POS tagging with context local tag generation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Process-*

ing, pages 196–206, Cambridge, MA, October 2010. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D10-1020.

Kevin Murphy, Yair Weiss, and Michael Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of the Proceedings of the Fifteenth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 467–47, San Francisco, CA, 1999. Morgan Kaufmann.

V. Nastase. Unsupervised all-words word sense disambiguation with grammatical dependencies. In *Proceedings of IJCNLP*, 2008.

Sebastian Padó and Mirella Lapata. Dependency-based construction of semantic space models. *Comput. Linguist.*, 33:161–199, June 2007. ISSN 0891-2017. doi: http://dx.doi.org/10.1162/coli.2007.33.2.161. URL http://dx.doi.org/10.1162/coli.2007.33.2.161.

M. Palmer, H. Trang Dang, and C. Fellbaum. Making fine-grained and coarse-grained sense distinctions, both manually and automatically. *Natural Language Engineering*, 13:137–163, 2007.

Patrick Pantel, Rahul Bhagat, Bonaventura Coppola, Timothy Chklovski, and Eduard Hovy. ISP: Learning inferential selectional preferences. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 564–571, Rochester, New York, April 2007. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/N/N07/N07-1071.

Hoifung Poon and Pedro Domingos. Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1–10, Singapore, August 2009. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/D/D09/D09-1001`.

Sameer Pradhan, Edward Loper, Dmitriy Dligach, and Martha Palmer. Semeval-2007 task-17: English lexical sample, srl and all words. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 87–92, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/S/S07/S07-1016`.

Sebastian Riedel and Ivan Meza-Ruiz. Collective semantic role labelling with markov logic. In *CoNLL 2008: Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pages 193–197, Manchester, England, August 2008. Coling 2008 Organizing Committee. URL `http://www.aclweb.org/anthology/W08-2125`.

Alan Ritter, Mausam, and Oren Etzioni. A latent dirichlet allocation method for selectional preferences. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 424–434, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P10-1044`.

Eleanor Rosch. Cognitive representations of semantic categories. *Journal of Experimental Psychology: General*, 104(3):192 – 233, 1975.

ISSN 0096-3445. doi: DOI:10.1037/0096-3445.104.3.192. URL http://www.sciencedirect.com/science/article/pii/S0096344507600861.

Mark Sanderson. Retrieving with good sense. *Information Retrieval*, 2:49–69, 2000. ISSN 1386-4564. URL http://dx.doi.org/10.1023/A:1009933700147. 10.1023/A:1009933700147.

Serge Sharoff. Open-source corpora: Using the net to fish for linguistic data. *International Journal of Corpus Linguistics*, 11(4):435–462, 2006.

David Smith and Jason Eisner. Dependency parsing by belief propagation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 145–156, Honolulu, Hawaii, October 2008. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/D08-1016.

P. Smolensky. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial Intelligence*, 46:159–216, 1990.

Benjamin Snyder and Martha Palmer. The english all-words task. In Rada Mihalcea and Phil Edmonds, editors, *Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 41–43, Barcelona, Spain, July 2004. Association for Computational Linguistics.

Christopher Stokoe. Differentiating homonymy and polysemy in information retrieval. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 403–410, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/1220575.1220626. URL `http://dx.doi.org/10.3115/1220575.1220626`.

Charles Sutton, Andrew McCallum, and Khashayar Rohanimanesh. Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data. *J. Mach. Learn. Res.*, 8:693–723, May 2007. ISSN 1532-4435. URL `http://portal.acm.org/citation.cfm?id=1248659.1248684`.

Idan Szpektor and Ido Dagan. Learning entailment rules for unary templates. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 849–856, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. ISBN 978-1-905593-44-6. URL `http://portal.acm.org/citation.cfm?id=1599081.1599188`.

Idan Szpektor, Ido Dagan, Roy Bar-Haim, and Jacob Goldberger. Contextual preferences. In *Proceedings of ACL-08: HLT*, pages 683–691, Columbus, Ohio, June 2008. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P/P08/P08-1078`.

S. Thater, G. Dinu, and M. Pinkal. Ranking paraphrases in context. In *Proceedings of the ACL Workshop on Applied Textual Inference*, Singapore, 2009. URL `http://www.aclweb.org/anthology/W/W09/W09-2506`.

Stefan Thater, Hagen Fürstenau, and Manfred Pinkal. Contextualizing semantic representations using syntactically enriched vector models. In *Proceedings of the*

48th Annual Meeting of the Association for Computational Linguistics, pages 948–957, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P10-1097`.

D. H. Tuggy. Ambiguity, polysemy and vagueness. *Cognitive linguistics*, 4(2):273–290, 1993.

Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.*, 1(1-2):1–305, 2008. ISSN 1935-8237. doi: http://dx.doi.org/10.1561/2200000001.

W. Weaver. Translation. In W.N. Locke and A.D. Booth, editors, *Machine Translation of Languages: Fourteen Essays*. MIT Press, Cambridge, MA, 1949.

Stephen F. Weiss. Learning to disambiguate. *Information Storage and Retrieval*, 9 (1):33–41, 1973. ISSN 0020-0271. doi: DOI:10.1016/0020-0271(73)90005-3. URL `http://www.sciencedirect.com/science/article/pii/0020027173900053`.

Yair Weiss. Correctness of local probability propagation in graphical models with loops. *Neural Computation*, 12(1): 1–41, 2000. doi: 10.1162/089976600300015880. URL `http://www.mitpressjournals.org/doi/abs/10.1162/089976600300015880`.

Yorick Wilks. Is word sense disambiguation just one more nlp task? *Computers and the Humanities*, 34:235–243, 2000. ISSN 0010-4817. URL `http://dx.doi.org/10.1023/A:1002656922270`. 10.1023/A:1002656922270.

David Yarowsky. One sense per collocation. In *Proceedings of the workshop on Human Language Technology*, HLT '93, pages 266–271, Stroudsburg, PA, USA, 1993. Association for Computational Linguistics. ISBN 1-55860-324-7. doi: http://dx.doi.org/10.3115/1075671.1075731. URL `http://dx.doi.org/10.3115/1075671.1075731`.

David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, ACL '95, pages 189–196, Stroudsburg, PA, USA, 1995. Association for Computational Linguistics. doi: http://dx.doi.org/10.3115/981658.981684. URL `http://dx.doi.org/10.3115/981658.981684`.

J.S. Yedidia, W.T. Freeman, and Y. Weiss. Generalized belief propagation. *Advances in neural information processing systems*, pages 689–695, 2001.

Katsumasa Yoshikawa, Sebastian Riedel, Masayuki Asahara, and Yuji Matsumoto. Jointly identifying temporal relations with markov logic. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 405–413, Suntec, Singapore, August 2009. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P/P09/P09-1046`.

Deniz Yuret. Ku: Word sense disambiguation by substitution. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 207–214, Prague, Czech Republic, June 2007. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/S/S07/S07-1044`.

Deniz Yuret, Aydin Han, and Zehra Turgut. Semeval-2010 task 12: Parser evaluation using textual entailments. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 51–56, Uppsala, Sweden, July 2010. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/S10-1009.

Vita

Taesun Moon was born in Seoul, South Korea. He received his B.A. in English Language and Literature from Seoul National University in 2002. He received his M.A. in English Language and Literature from Seoul National University in 2006. He received an M.A. in Linguistics from the University of Texas at Austin in 2008.

Permanent address: 111-403 Banpo-bon-dong
Seocho-gu, Seoul,
South Korea, 137-770

This dissertation was typeset with LaTeX[†] by the author.

[†] LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.