

A Relation-Centric View of Semantic Representation Learning

Sujay Kumar Jauhar

August 24th, 2017

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA – 15213

Thesis Committee:

Eduard Hovy, Chair

Chris Dyer

Lori Levin

Peter Turney

*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in Language and Information Technology.*

Keywords: Semantics, Representation Learning, Relations, Structure

*To the women in my life.
My mother and my wife.*

Abstract

Much of NLP can be described as mapping of a message from one sequence of symbols to another. Examples include word surface forms, POS-tags, parse trees, vocabularies of different languages, etc. Machine learning has been applied successfully to many NLP problems by adopting the symbol mapping view. No knowledge of the sets of symbols is required: only a rich feature representation to map between them.

The task of coming up with expressive features is typically a knowledge-intensive and time-consuming endeavor of human creativity. However the representation learning paradigm (Bengio et al., 2013) offers an alternative solution, where an optimal set of features are automatically learned from data for a target application. It has been successfully applied to many applied fields of machine learning including speech recognition (Dahl et al., 2010), vision and object recognition (Krizhevsky et al., 2012), and NLP (Socher et al., 2010; Schwenk et al., 2012).

One particular area of NLP that has received a lot of interest, from the perspective of representation learning, is lexical semantics. Many efforts have focussed on attempting to learn high-dimensional numerical vector representations for units of meaning (most often words) in a vocabulary. While the current slew of research uses neural network techniques (Collobert and Weston, 2008; Mikolov et al., 2013a), word-embeddings have been around much longer in the form of distributional vector space models (Turney and Pantel, 2010). We use the term semantic representation learning to encompass all these views and techniques.

Semantic representations learned automatically from data have proven to be useful for many downstream applications such as question answering (Weston et al., 2015), word-sense discrimination and disambiguation (McCarthy et al., 2004; Schütze, 1998), and selectional preference modeling (Erk, 2007). The successes of representation learning, along with its unsupervised and data-driven nature make it a compelling modelling solution for semantics.

However, unlike manual feature engineering, it remains difficult to inject linguistic or world knowledge into the process of learning semantic representations automatically. This has led many models to still fail to account for many basic properties of human languages, such as antonymy, polysemy, semantic composition, and negation, among others. We attempt to overcome these shortcomings in this thesis.

When considering semantic representation learning models, two questions may be asked: firstly, *how* is the model learned; and secondly; *what* does the model represent? The research community has largely focussed on the first of the two questions – the *how* – proposing many different machine learning solutions that yield semantic vector representations. In this thesis, we instead focus on the second of the two questions – the *what* – namely gaining insight into the underlying semantic information that is actually captured by models.

A crucial component of every representation learning model, is the notion of a *context*. This is the basis upon which models extract count statistics to process and

produce representations. We propose that understanding the hypothesis that defines contexts is the key to making sense of *what* a model represents (and even what it will be unable to capture no matter *how* it is learned). In other words, being able to specify what is being counted leads to insights into what is being represented.

Perhaps the most common example of such a hypothesis is the so-called distributional hypothesis, which intuits that words that occur in similar contexts tend to have similar meanings. This hypothesis is operationalized to define a *context* over word windows. Turney and Pantel (2010) describe other such hypotheses that define different contexts such as entire documents or patterns. Models that use these hypotheses capture different information because they define *contexts* differently. The question we ask ourselves: is there an over-arching view to understand any hypothesis and the consequent contexts that they engender?

We believe, in fact, there is. This thesis holds the view that relations and structure play a primary role in creating meaning in language, and that each linguistic phenomenon has its own set of relations and characteristic structure. We propose that each hypothesis operationalized by semantic representation learning models in fact defines a different relation, and the consequent structure is the *context* upon which counts are extracted and processed.

From this point of view the distributional hypothesis is first and foremost a relational hypothesis, only one among many possible relational hypotheses. It becomes clear why models that use this hypothesis to extract counts are incapable of capturing all of semantics. The *distributional* relation is simply not the right or most expressive relation for many linguistic phenomena or semantic problems – composition, negation, antonymy etc. It would thus seem imperative to integrate relations into modelling semantics with representation learning.

This thesis proposes a relation-centric view of semantics. Our central conjecture is that semantic relations are fundamentally important in giving meaning to words (and generally all of language). We hypothesize that by understanding and articulating the principal relations relevant to a particular problem, one can leverage them to build more expressive and empirically superior models of semantic representations.

We begin by introducing the problem of semantic representation learning, defining relations and arguing for their centrality in meaning. We then outline a framework within which to think about and implement solutions for integrating relations in representation learning. The framework is example-based, in the sense that we draw from concrete instances in this thesis to illustrate five sequential themes that need to be considered.

The rest of the thesis provides concrete instantiations of these sequential themes. The problems we tackle in the four subsequent chapters (Chapters 3– 6) are broad, both from the perspective of the target learning problem as well as the nature of the relations and structure that we use. We thus show that by focussing on relations, we can learn models of word meaning that are semantically and linguistically more expressive while also being empirically superior.

In the first of these chapters, we outline a general methodology to learn sense-specific representations for polysemous words by using the relational links in an on-

tological graph to tease apart the different senses of words. In the next chapter, we show how the relational structure of tables can be used to produce high-quality annotated data for question answering, as well as build models that leverage this data and structure to perform question answering. The third of the four chapters deals with the problem of inducing an embedded semantic lexicon using minimal annotation. In this chapter we jointly leverage local token-level and global type-level relations between predicate-argument pairs to yield representations for words in the vocabulary, as well as a codebook for a set of latent relational argument slots. Finally, in the last content chapter we show how tensor-based structured distributional semantic models can be learned for tackling the problems of contextual variability and faceted comparison; we give examples of both syntactic and latent semantic relational integration into the model.

This thesis provides a step towards better understanding computational semantic modelling with representation learning. It allows for a unified view of word meaning through semantic relations, while supporting the representation of richer, more comprehensive models of semantics. At the same time, it results in representations that can potentially benefit many downstream NLP applications that rely on features of semantic units.

Acknowledgments

I have been immensely fortunate to have had the opportunity of working with some of the most brilliant, generous, humble and down-to-earth people during my time as a graduate student at CMU. I have learned so much, and from so many people that I am afraid I will undoubtedly forget to mention a few in this note of acknowledgement. If you are one of these people, forgive my limited memory and know that I am grateful for your impact on my life as a PhD student.

First and foremost, I would like to thank my advisor Ed Hovy. This thesis would literally not have been possible without his constant guidance and support. He has given me vast latitude to grow and develop as a researcher over the last five years, while his deep intuition and understanding of problems that were truly worth tackling have helped me steer back whenever I drifted off course. His has been a voice of sanity and encouragement throughout, and I am very grateful to have had such a fantastic advisor.

I would also like to give my sincere thanks to the other members of my thesis committee. To Chris Dyer, who is a fount of creativity and brilliance, and whose excitement for research is infectious. To Lori Levin, who taught me to appreciate the richness, nuance, complexity and diversity of language, and that NLP is all the more exciting and interesting for it. And to Peter Turney, who remains a role model scientist for me: someone who is deeply knowledgeable in many different areas, and who is thoughtful, meticulous and impactful in all the work that he does.

This thesis has only been made better by my committee's insight and comments. Any errors therein are solely a reflection of my own foibles.

My friends and collaborators through grad school have probably had the biggest impact on my ability to actually do research. They have taught me the tools of the trade and given me the wherewithal to deal with the daily grind of being a researcher. I have had the honor of collaborating with some very brilliant people: Dirk Hovy, Kartik Goyal, Huiying Li, Mrinmaya Sachan, Shashank Srivastava, Yun-Nung Chen, Florian Metze, Manaal Faruqui, Jesse Dodge, Noah Smith, Raul Guerra, Edgar González, Marta Recasens and of course members of my committee Ed, Chris and Peter.

I'd also like to give a shout-out to folks with whom I had the pleasure of sharing an office. Kartik, Mrinmaya, Shashank, Huiying, Whitney Sanders and I were crammed into GHC 5510 (aka the storeroom) like sardines. Those were early days for me at CMU as a Masters student and you guys made working difficult, at times. But I wouldn't swap it for the world. As a PhD student I often worked from home, but when I did come in to the office I had the pleasure of sharing space with Subhdeep Moitra, Keerthiram Murugesan and Zhilin Yang. I hope you guys enjoyed the freedom of an empty desk my absence often afforded you.

I have had the fortune of being taught by some stellar faculty at CMU. Thank you to Ed Hovy, Roni Rosenfeld, Noah Smith, Florian Metze, Chris Dyer and Lori Levin for making classes so much fun.

The LTI would not function, if it weren't for the often thankless contributions of the wonderful staff. A special thank you to Stacey Young for being such a great help to the student body in so many different capacities.

A major formative influence on my years as a PhD student were the internships I did at Google and AI2. I would like to thank my mentors, Marta Recasens, Edgar González and Peter Turney, who made the two summers I spent away from CMU seem like a well-deserved break – even though I was still doing research. I'd also like to thank Oren Etzioni, whose blessing permitted AI2 to fund part of the research in Chapter 4 of this thesis. I had the opportunity to learn from, collaborate, interact and generally hang out with some other seriously cool people at Google and AI2, both fellow interns and full-timers. Raul Guerra, Dani Yogatama, Daniel Khashabi, Ashish Sabharwal, Chen-Tse Tsai, Rachel Rudinger, Kavya Srinet, Chandra Bhagavatula, Jayant Krishnamurthy and Peter Clark – thank you!

I am deeply grateful to the people who made me feel a part of their family when I was half a world away from home. Arpita and Manish, you took me into your home when I arrived in the US with nothing more than a couple of suitcases, and a head full of dreams and uncertainty. You treated me like one of your own during the first few difficult months of my life in Pittsburgh. You will always be family and I do not know how to thank you enough. Kalpana aunty, Mohana aunty, Ramesh uncle and Sruthi: your presence in California made my time at Google all the more memorable and special. Thank you for your warmth, hospitality and thoughtfulness. And of course Sanjeev chacha and Nishtha bhua in Seattle: I did not know that we were related until two years ago. But now that I do I am very grateful for knowing such wonderful relatives in the US.

My friends in Pittsburgh have been my support system, without whom I would have almost certainly gone insane, or worse – quit my PhD. Whether it was pot luck, board games, so-bad-they're-actually-good movies, pizza night or just hanging out, I almost always had something to look forward to on Saturday nights thanks to you guys. KD, Marimaya, Avimama, Shashank dood, Snigdha dood – you were my oldest and continue to be some of the best friends I have made in Pittsburgh. Masterji, Akanksha, Sudhi, Pradeep, Modhurima, Keerthi – it has been great getting to know and hang out with all of you. Pallavi, Subhodeep, Prasanna, Manaal, Swabha, Jesse, Volkan, Nico, Lin, Kavya, Sai – thank you all for your friendship. If I've forgotten anybody, I sincerely apologize.

Last, but certainly not the least, I would like to thank my large, loud, Indian family. They have put up with my moodiness, my unique brand of crazy and the many months spent away from home with unconditional love and support. A special thanks to my wife, whom I finally married after knowing for ten years – six of which were spent doing long-distance across continents. Her steadfast love and constancy have helped me through troubled times more than I can count and I am proud to say that she is my best friend with benefits and my partner in crime. Finally, I owe my identity, my values, half my genes and literally my life to my mother. She raised me almost single-handedly to be the person I am today and nothing I do in this life will repay the debt of love I owe her.

Contents

1	Introduction	1
1.1	Contributions of this Thesis	10
2	Relational Representation Learning	13
3	Ontologies and Polysemy	21
3.1	Related Work	22
3.2	Unified Symbolic and Distributional Semantics	23
3.2.1	Retrofitting Vectors to an Ontology	24
3.2.2	Adapting Predictive Models with Latent Variables and Structured Regularizers	26
3.3	Resources, Data and Training	28
3.4	Evaluation	30
3.4.1	Experimental Results	30
3.4.2	Generalization to Another Language	32
3.4.3	Antonym Selection	33
3.5	Discussion	35
3.5.1	Qualitative Analysis	35
3.6	Conclusion	36
4	Tables for Question Answering	37
4.1	Related Work	39
4.2	Tables as Semi-structured Knowledge Representation	40
4.2.1	Table Semantics and Relations	40
4.2.2	Table Data	41
4.3	Answering Multiple-choice Questions using Tables	42
4.4	Crowd-sourcing Multiple-choice Questions and Table Alignments	43
4.4.1	MCQ Annotation Task	43
4.5	Feature Rich Table Embedding Solver	45
4.5.1	Model and Training Objective	45
4.5.2	Features	47
4.5.3	Experimental Results	50
4.6	TabNN: QA over Tables with Neural Networks	53
4.6.1	The Model	53

4.6.2	Training	58
4.6.3	Experiments and Evaluation	59
4.7	Conclusion	63
5	Embedded Frame Lexicons	65
5.1	Related Work	67
5.2	Joint Local and Global Frame Lexicon Induction	68
5.2.1	Local Predicate-Argument Likelihood	69
5.2.2	Global Latent Slot Regularization	69
5.2.3	Optimization	71
5.2.4	Relational Variant	73
5.3	Experiments and Evaluation	73
5.3.1	Implementational Details	73
5.3.2	Pseudo Disambiguation of Selection Preference	74
5.3.3	Frame Lexicon Overlap	76
5.3.4	Qualitative Analysis of Latent Slots	79
5.4	Conclusion and Future Work	80
6	Structured Distributional Semantics	81
6.1	Distributional Versus Structured Distributional Semantics	83
6.2	Syntactic Structured Distributional Semantics	84
6.2.1	Related Work	85
6.2.2	The Model	85
6.2.3	Mimicking Compositionality	87
6.2.4	Single Word Evaluation	90
6.2.5	Event Coreference Judgment	91
6.3	Latent Structured Distributional Semantics	94
6.3.1	Related Work	96
6.3.2	Latent Semantic Relation Induction	96
6.3.3	Evaluation	99
6.3.4	Semantic Relation Classification and Analysis of the Latent Structure of Dimensions	102
6.4	Conclusion	105
7	Conclusion	107
7.1	Summary of Contributions	107
7.2	Future Directions	109
	Bibliography	113

List of Figures

1.1	Examples illustrating various sources of information exhibiting the dual nature of structure and relations. Note how Definitions 1 and 2, as well as Corollary 1 apply to each of these examples. All these sources of relations and structure are used in this thesis.	9
3.1	A factor graph depicting the retrofitting model in the neighborhood of the word “bank”. Observed variables corresponding to word types are shaded in grey, while latent variables for word senses are in white.	24
3.2	The generative process associated with the skip-gram model, modified to account for latent senses. Here, the context of the ambiguous word “bank” is generated from the selection of a specific latent sense.	27
3.3	Performance of sense specific vectors (<i>wnsense</i>) compared with single sense vectors (original) on similarity scoring in French. Higher bars are better. <i>wnsense</i> outperforms <i>original</i> across 2 different VSMs on this task.	33
3.4	Performance of sense specific vectors (<i>wnsense</i>) compared with single sense vectors (original) on closest to opposite selection of target verbs. Higher bars are better. <i>wnsense</i> outperforms <i>original</i> across 3 different VSMs on this task.	34
4.1	Example table from MTurk annotation task illustrating constraints. We ask Turk-ers to construct questions from blue cells, such that the red cell is the correct answer, and distracters must be selected from yellow cells.	43
4.2	The component of the TabNN model that embeds question and row in a joint space and predicts an alignment score between the two. Embedding layers are omitted for visual compactness.	56
4.3	The component of the TabNN model that embeds choices and column in a joint space and predicts an alignment score between the two. Embedding layers are omitted for visual compactness.	57
4.4	The component of the TabNN model that embeds a single choice and table cell in a joint space and predicts an alignment score between the two. Embedding layers are omitted for visual compactness.	58
5.1	The generative story depicting the realization of an argument from a predicate. Argument words are generated from latent argument slots. Observed variables are shaded in grey, while latent variables are in white.	70

5.2	Illustration of a potential binary feature matrix (1s colored, 0s blank) learned by IBP. Rows represent verbs from a fixed vocabulary, while columns are of a fixed but arbitrary number and represent latent arguments.	71
6.1	A sample parse fragment and some examples of triples that we extract from it. Triples are generated from each dependency arc by taking the cross-product of the annotations that the nodes connected to the arc contain.	86
6.2	A depiction of the composition of two words. After composition the joint unit is treated as a single node, and the expectations of that combined node are then computed from the PropStore.	87
6.3	Visual representation of the intuition behind the Latent Relational SDSM. The idea revolves around inducing a relation space from which a row vector is transformed to a column vector in SDSM space.	97
6.4	Evaluation results on WS-353 and ESL with varying number of latent dimensions. Generally high scores are obtained in the range of 100-150 latent dimensions, with optimal results on both datasets at 130 latent dimensions.	101
6.5	Correlation distances between semantic relations' classifier weights. The plot shows how our latent relations seem to perceive humanly interpretable semantic relations. Most points are fairly well spaced out, with opposites such as "Attribute" and "Non-Attribute" as well as "Similar" and "Contrast" being relatively further apart.	104

List of Tables

3.1	Similarity scoring and synonym selection in English across several datasets involving different VSMs. Higher scores are better; best scores within each category are in bold. In most cases our models consistently and significantly outperform the other VSMs.	30
3.2	Contextual word similarity in English. Higher scores are better.	32
3.3	Training time associated with different methods of generating sense-specific VSMs.	35
3.4	The top 3 most similar words for two polysemous types. Single sense VSMs capture the most frequent sense. Our techniques effectively separates out the different senses of words, and are grounded in WordNet.	36
4.1	Example part of table concerning phase state changes.	40
4.2	Comparison of different ways of generating MCQs with MTurk.	44
4.3	Examples of MCQs generated by MTurk. Correct answer choices are in bold.	44
4.4	Summary of features. For a question (q) and answer (a) we compute scores for elements of tables: whole tables (t), rows (r), header rows (h), columns (c), column headers (c_h) and cells (s). Answer-focus (a_f) and question-focus (q_f) terms added where appropriate. Features marked \diamond denote soft-matching variants, marked with while those marked with a \dagger are described in further detail in Section 4.5.2. Finally, \bullet features denote those that received high weights during training with all features, and were subsequently selected to form a compact FRETTS model.	48
4.5	Evaluation results on three benchmark datasets using different sets of tables as knowledge bases. Best results on a dataset are highlighted in bold.	50
4.6	Ablation study on FRETTS, removing groups of features based on level of granularity. \diamond refers to the soft matching features from Table 4.4. Best results on a dataset are highlighted in bold.	52
4.7	A summary of the different TabNN model variants we train along with the number of parameters learned with each model.	60
4.8	Evaluation of the TabNN model. Training results are provided to gain insight into the models and should not be taken to represent goodness of the models in any way. The best FRETTS model’s test results are given as a comparison.	61
4.9	Evaluation of the TabNN model, trained on only 90% of the TabMCQ dataset. Test results on the 10% held out portion of TabMCQ show that the model itself is capable of learning, and that it’s architecture is sound. There is, however, a severe domain mismatch between TabMCQ and the ESSQ test set.	61

5.1	Results on pseudo disambiguation of selectional preference. Numbers are in % accuracy of distinguishing true arguments from false ones. Our models all outperform the skip-gram baseline.	75
5.2	Results on the lexicon overlap task. Our models outperform the syntactic baseline on all the metrics.	76
5.3	Examples for several predicates with mappings of latent slots to the majority class of the closest argument vector in the shared embedded space.	79
6.1	Single word evaluations on similarity scoring and synonym selections. SDSM does not clearly outperform DSM, likely due to issues of sparsity.	91
6.2	Effects of SVD methods on SDSM representations. With higher order tensor SVD, SDSM outperforms DSM baseline.	91
6.3	Cross-validation performance of Event Coreference Judgement on IC and ECB datasets. SDSM outperforms the other models on F-1 and accuracy on both datasets.	93
6.4	Results on the WS-353 similarity scoring task and the ESL synonym selection task. LRA-SDSM significantly outperforms other structured and non-structured distributional semantic models.	100
6.5	Results on Relation Classification Task. LR-SDSM scores competitively, outperforming all but the SENNA-AVC model.	102

Chapter 1

Introduction

Most of NLP can be viewed as a mapping from one sequence of symbols to another. For example, translation is mapping between sequences of symbols in two different languages. Similarly, part-of-speech tagging is mapping between words' surface forms and their hypothesized underlying part-of-speech tags. Even complex structured prediction tasks such as parsing or simple tasks like text categorization labelling are mappings: only between sequences of symbols of different granularities. These sequences may or may not interact with one another in complex ways.

From a computational perspective, this is a powerful and appealing framework. There is no need to know anything about linguistics or language: only that it can be represented as sequences of symbols. Using this symbol mapping perspective, machine learning has been applied with great success to practically every NLP problem.

Driven by better learning techniques, more capable machines and vastly larger amounts of data, useable natural language applications are finally in the hands of millions of users. People can now translate text between many languages via services like Google Translate¹, or interact with virtual personal assistants such as Siri² from Apple, Cortana³ from Microsoft or Alexa⁴ from Amazon.

However, while these applications work well for certain common use cases, such as translating between linguistically and structurally close language pairs or asking about the weather, they continue to be embarrassingly poor for processing and understanding the vast majority of human language.

For example, asking Siri “I think I have alcohol poisoning. What do I do?” led one user to receive the darkly humorous failed response “I found 7 liquor stores fairly close to you.” with a list of map locations to the liquor stores. Many instances of NLP systems failing occur because language is complex, nuanced and very often highly ambiguous (although the example above is not). And despite recent efforts, such as Google's Knowledge Graph project⁵ to build repositories of world and linguistic knowledge – models for semantics continue to be incomplete

¹<https://translate.google.com/>

²<https://en.wikipedia.org/wiki/Siri>

³[https://en.wikipedia.org/wiki/Cortana_\(software\)](https://en.wikipedia.org/wiki/Cortana_(software))

⁴https://en.wikipedia.org/wiki/Amazon_Alexa

⁵https://en.wikipedia.org/wiki/Knowledge_Graph

and rudimentary, and extremely knowledge-intensive to build.

From the symbol transformation perspective, a typical NLP pipeline involves taking an input A in symbol vocabulary \mathcal{A} , featurizing this input by some function F to express the important dimensions of distinction, learning or applying a model M to the featurized input and finally performing a transformation to a sequence B in symbol vocabulary \mathcal{B} .

To concretize this notional pipeline, consider the example of part-of-speech-tagging:

- Sequence A is typically text in the source language – say English, which contains a vocabulary of English tokens \mathcal{A} .
- The function F may consist of features like lexical-to-tag mapping probabilities that are estimated from some training data, or induced in an unsupervised fashion. In machine learning terms this process is known as featurization.
- The model M concerns the definition and learning of a mathematical function that is able to map featurized input to output. In the case of part-of-speech tagging this could be, for example, a Hidden Markov Model (HMM) (Rabiner and Juang, 1986) or a Conditional Random Field (CRF) (Lafferty et al., 2001).
- Finally the transformation via model M to output is known as decoding or inference. The output symbols obtained in the case of part-of-speech tagging would be a sequence of tags B from a vocabulary consisting of some tagset \mathcal{B} , such as the Penn Treebank tagset⁶.

A similar sort of abstract pipeline can be envisaged for almost any NLP problem.

A symbol mapping pipeline of this kind systematizes and separates the different steps involved in the transformation. Thus, the machine learning model M does not really need to know anything beyond the denotation of the sets of symbols \mathcal{A} and \mathcal{B} . Rather, a much more important requirement is an expressive featurization F that characterizes the two symbol vocabularies, the example sequences A and B and their relationships to one another. These features provide the basis upon which a symbol mapping model can be learned to generalize from known to unseen examples.

This thesis is centrally concerned with the improvement of the *featurization* component of the pipeline F .

Until recently, the work of thinking up and engineering expressive features for machine learning models was still, for the most part, the responsibility of the researcher or engineer. It was up to the researcher or engineer to think up clever ways of featurizing the input, and in the case of NLP often applying her insight into linguistics and knowledge about the world to do this effectively.

However, the **representation learning** paradigm (Bengio et al., 2013) was proposed as an alternative that eliminated the need for manual feature engineering. In general, representation learning allows a system to automatically learn a set of optimal features given a target application and training data. In the context of the pipeline described above, the featurization step F is no longer a separate stage in the pipeline but rather a sub-component of the model M .

In the running example of part-of-speech tagging above, the model M might now be a sequential neural network such as a Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber,

⁶https://www.ling.upenn.edu/courses/Fall_2003/ling001/penn_treebank_pos.html

1997) that only requires examples of aligned inputs A and B . The features or *representations* of the input A are no longer required to be coded manually via F . Rather, F might now be an embedding matrix that maps words in the input vocabulary to some high dimensional vector representation. This embedding matrix is induced automatically through the process of training the network M .

Representation learning has been successfully used in several applied research areas of machine learning including speech recognition (Dahl et al., 2010; Deng et al., 2010), vision and object recognition (Ciresan et al., 2012; Krizhevsky et al., 2012), and NLP Socher et al. (2010); Schwenk et al. (2012); Glorot et al. (2011). In particular for NLP, Collobert et al. (2011) showed how a single architecture could be applied in an unsupervised fashion to many NLP problems, while achieving state-of-the-art results (at the time) for many of them.

There have since been a number of efforts to extend and improve the use of representation learning for NLP. One particular area that has received a lot of recent interest in the research community is semantics.

In semantics, representation learning is commonly referred to in the literature as the so-called word-embedding problem. Typically, the aim is to learn features (or *embeddings*) for words in a vocabulary – which are often the units of lexical semantics – automatically from a corpus of text. However in this thesis, we adopt a broader view of lexical semantics where the *lexicon* can consist of other semantic units for which we may wish to learn representations. Therefore, where appropriate we will talk about word embeddings, but otherwise deal with the general problem of representation learning for semantics.

There are two principal reasons that representation learning for semantics is an attractive proposition: it is unsupervised (thus requiring no human effort to build), and data-driven (thus containing little bias from human intervention). However, in contrast to manual feature engineering it is significantly more difficult to introduce coded linguistic or world knowledge.

The goal of this thesis is centered around reconciling this difference: to enable representation learning models of lexical semantics to also leverage and encode linguistic or world knowledge. In what follows we shall motivate and outline a possible solution.

To understand the variety of different approaches to representation learning for semantics, one can examine the following two questions:

1. How is a model learned?
2. What does the model represent?

The first question attempts to classify the general class of methods or models used to learn the representations in question. The currently dominant approach is the **neural embedding** class of models which use neural networks to maximize some objective function over the data and learn word vectors in the process. The LSTM mentioned in the context of the part-of-speech tagging example above might be one such model. Some notable papers that learn semantic representations on the basis of neural networks are Collobert et al. (2011), Mikolov et al. (2010) Mikolov et al. (2013a) and Pennington et al. (2014).

Earlier work on **count statistics** over windows and transformations thereof are other ways to automatically learn representations of some semantic units from data. These include heuristic methods such as Term Frequency Inverse Document Frequency (TF-IDF) and Point-wise Mutual Information (PMI) weighting, and mathematically motivated **matrix factorization** techniques

such as Latent Semantic Analysis (Deerwester et al., 1990). Turney and Pantel (2010) provides an overview of these diverse approaches. More recently, Levy and Goldberg (2014) show an inherent mathematical link between older matrix factorization work and current neural embedding techniques.

Addressing the second – and, for this thesis, the more interesting and relevant – question specifies the underlying semantic hypothesis that is captured by the model. In other words, how are raw statistics over text data converted by the model into a semantic representations, and what semantic properties are maintained by this transformation.

Every semantic representation learning model operates on the notion of a *context*. The context specifically defines what information is available during training, and consequently what statistics can be computed and converted into representations. Different kinds of contexts, naturally lead to models that capture different information. This is a crucial phenomenon, and is central to the rest of this thesis.

Consider the example of perhaps the most commonly used semantic hypothesis: the **distributional hypothesis** (Harris, 1954). Articulated by Firth (1957) in the popular dictum “You shall know a word by the company it keeps”, the hypothesis claims that words that have similar meaning tend to occur in similar contexts.

The example below illustrates how the distributional hypothesis works. For example, the words “stadium” and “arena” may have been found to occur in a corpus with the following contexts:

- People crowded the *stadium* to watch the game.
- The *arena* was crowded with people eager to see the match.
- The *stadium* was host to the biggest match of the year.
- The most important game of season was held at the *arena*.

From these sentences it may be reasonably inferred (without knowing anything about them a priori) that the words “stadium” and “arena” in fact have similar meanings.

Notice that the *context*, in this case, is simply the neighboring words – perhaps parametrized by a window size. Extending this reasoning to extract *neighborhood* statistics from a corpus of millions of words, which contain many hundreds of occurrences of the words “stadium” and “arena” one can build a reasonably good and robust representation of the two words.

This is a very simple yet extremely powerful insight and the contributions of the distributional hypothesis on representation learning in semantics remain profound. Operationalized in practise, most word embedding models use the insight of the distributional hypothesis to essentially count statistics over a word’s neighborhood contexts across a corpus and represent all those statistics in a compact form as a vector representation for that word. The result is a vector space model (VSM), where every word is represented by a point in some high-dimensional space. We refer to any model that automatically produces such a VSM as a semantic representation learning model, regardless of what underlying machine learning technique is used to learn it.

While much effort in the research community has been devoted towards improving *how* models learn word vectors, most work gives little consideration to *what* the model represents. They simply uses the standard distributional hypothesis, because of it’s simplicity and wide empirical success to learning representations that are useful for a host of downstream NLP tasks. They

essentially use the same kind of *context* to extract and process count statistics. So while word-embedding models may be learned differently, most of them continue to represent the same information.

It should be noted that the distributional hypothesis is not the only way of converting raw text statistics into semantic representations. Turney and Pantel (2010) lists several other examples of hypotheses. These include the **bag-of-words hypothesis** (Salton et al., 1975), which conjectures that documents can be characterized by the frequency of words in it; the **latent relation hypothesis** (Turney, 2008a), which proposes that pairs of words that occur in similar patterns tend to have similar relations between them; and the **extended distributional hypothesis** which specifies that patterns that occur in similar contexts have similar meaning.

In the terminology of Turney and Pantel (2010), each of these hypotheses yields a different representation matrix. These are, for example, a Term-Document matrix in the case of the bag-of-words hypothesis, or a Pair-Pattern matrix in the case of the latent relation hypothesis. The columns of these matrices are tied to our idea of *contexts*: they define the kind of data that can be leveraged to extract and process count statistics.

An important and interesting question, in relation to this thesis, is whether there is an overarching view that encompasses all these different hypotheses and their related contexts in a single framework. As we shall argue in what follows, there is.

Semantic vectors in continuous space representations can be used for meaningful semantic operations such as computing word similarity (Turney, 2006), performing analogical reasoning (Turney, 2013) and discovering lexical relationships (Mikolov et al., 2013b).

These models have also been used for a wide variety of NLP applications including question answering (Tellex et al., 2003), semantic similarity computation (Wong and Raghavan, 1984; McCarthy and Carroll, 2003), automated dictionary building (Curran, 2004), automated essay grading (Landauer and Dumais, 1997), word-sense discrimination and disambiguation (McCarthy et al., 2004; Schütze, 1998), selectional preference modeling (Erk, 2007) and identification of translation equivalents (Hjelm, 2007).

Nevertheless, while they have been widely successful, representation learning models for semantics often fail at capturing many common linguistic phenomena. Semantic *compositionality* and *antonymy* are just two examples of phenomena that are stumbling blocks to many common vector learning approaches.

And this failure stems, not from *how* the representations are learned, but rather *what* semantic information is captured by them; in other words, the semantic hypothesis that is used to relate similar representations to one another. Most often this hypothesis is the distributional hypothesis. And while it continues to be very influential for its simplicity and effectiveness, it remains an incomplete explanation for the diversity of possible expressions and linguistic phenomena in all of natural language.

For example, compositionality is a basic property of language where unit segments of meaning combine to form more complex meaning. The words “big”, “brown” and “cow” all have unique meanings, and when put together produce a precise semantic meaning – a “big brown cow” – that is more than just big-ness, brown-ness or cow-ness. But in a VSM, where the unit segments are words, it remains unclear – and according to previous work non-trivial (Mitchell and Lapata, 2008) – how the vectors for words should be combined to form their compositional embedding. From the perspective of the distributional hypothesis, there is no signal in

the *neighborhood* or “big” or “brown” that says precisely how these words should modify the representation of “cow”.

Another example of a language phenomenon that is difficult to capture is antonymy. Words such as “good” and “bad” are semantic inverses, but are often contextually interchangeable. You can have a “good day” or a “bad day”, eat a “good meal” or a “bad meal” and watch a “good TV show” or a “bad TV show”. Thus, based on the distributional hypothesis, which relies on the contextual neighborhood of words, the vectors assigned to “good” and “bad” would be very close in the resulting semantic space. There exist solutions in the literature that require making modifications to the distributional hypothesis template (Turney, 2008b, 2011) in order to help models distinguish between synonymy and antonymy.

Yet another property of language that standard representation learning techniques are incapable of handling is polysemy, where an identical surface form has more than one meaning. For example, the word “bank” may refer to a financial institution or the shore of a river. Because representation learning techniques are unsupervised and operate on large unannotated corpora, the string “bank” is treated as a single semantic unit, despite the fact that it has more than one sense. From the perspective of semantic hypotheses, the distributional context of the different senses are conflated and only a single representation is learned. Again, modifications to the distributional template (Pantel and Lin, 2002; Reisinger and Mooney, 2010; Neelakantan et al., 2014) must be made to allow them to recognize different senses of words.

These examples of linguistic phenomena may seem disparate without a common thread tying them together. However, we’d like to propose that there is, in fact, an over-arching view that encompasses all these phenomena. Not only this, the same view also allows us to regard the distributional hypothesis as just another example phenomenon.

This view is that the meaning of language emerges as much from its constituent parts as from the structure that binds these parts together or the relations that link them to one another. In fact, the hypothesis that is captured by a representation learning technique – the *what* of the model – is essentially a semantic relation. That semantic relation defines the nature of the *context* that is given to a model, and consequently what linguistic properties are transferred to the semantic representations and how effective they will be at solving different problems.

This is a central contribution of this thesis.

The distributional hypothesis, for example, is encoded by the *word-neighbor-word* relation. The other example hypotheses from Turney and Pantel (2010) can be similarly encoded as relations: the bag-of-words hypothesis is in fact a *belong-to-same-document* relation, and the latent relational hypothesis is in fact a *word-neighbor-pattern*. Note that these are just informal relation names that we have assigned to illustrate as examples. We will formalize these in what follows.

This relational view of semantic representation learning also allows us to explain the failures of models. A model that is learned with the distributional hypothesis does not accurately capture antonymy simply because antonymy require a different relation than the *word-neighbor-word* relation. In other words, some structural or relational information has been ignored or misrepresented by using the *word-neighbor-word* relation, and therefore the resulting model performs poorly at recognizing antonymy.

Before proceeding we need to define, though, what we mean by relation and structure. We adopt a mathematical view of these terms.

We begin by defining the word “object” in the broadest sense to mean any cognitive atom,

such as a word, a phrase, a sentence, or any textual or even non-textual phenomenon such a visual thing, etc.

For purposes of clarity we will distinguish between object

We define a relation in the following way.

Definition 1. Consider one set of objects o_1^A, \dots, o_N^A belonging to a vocabulary set Ω^A and another set of objects o_1^B, \dots, o_M^B belonging to a vocabulary set Ω^B . A relation r satisfying a logical predicate p is a subset of the ordered cross-product of the two sets $\Omega^A \times \Omega^B$, such that any $(o_n^A, o_m^B) \in \Omega^A \times \Omega^B$ if and only if p is true for (o_n^A, o_m^B) .

For example, the distributional hypothesis is easily accommodated into this definition. Consider Ω^A and Ω^B to be one and the same: the set of words in a corpus. Then the distributional hypothesis is encoded by the relation that defines the subset of word pairs (o_n^A, o_m^B) that satisfy the predicate $|i(o_n^A) - i(o_m^B)| \leq k$, where $i(\cdot)$ is a function that returns the index of a word token, and k is the *neighborhood* parameter. The relation essentially captures pairs of words that occur within a fixed window k of one another in a corpus.

The bag-of-words hypothesis encodes a very similar relation. Only the neighborhood parameter k is eliminated in favor of an entire document.

Turney and Pantel (2010) explicitly tie *frequency* to all the different hypotheses and the corresponding representation matrices they describe. And this is certainly true: frequency is inalienable from meaning in representation learning. Every semantic representation learning model is *always* learned by performing some counting over contexts. However, the idea of a relation precedes the actual implementation of a model. In other words, a relation defines and gives rise to a context: it specifies *what* the model is going to learn. *How* the model is in fact learned, – what mechanism is used to count and process things in a context, or what types of counts are given more or less importance is a different issue entirely. This is an important distinction that we make, in order to separate rather than conflate the notions of relations and frequency.

For example, the relation between a word and a specific neighbor (for the distributional hypothesis) *exists* regardless of whether it occurs once or many thousands of times. It is a different matter that a relation that occurs once is likely less informative than one that occurs many thousands of times. That decision concerns *how* the model is learned, not *what* it is designed to capture.

Expanding on the definition of a relation we define structure as the following.

Definition 2. Consider the set of relations r_1, \dots, r_K that satisfy corresponding logical predicates p_1, \dots, p_k defined over object set pairs $(\Omega_{A,1}, \Omega_{B,1}), \dots, (\Omega_{A,K}, \Omega_{B,K})$. A structure s is a union of the cross-product pairs obtained by applying all the relations r_1, \dots, r_k with predicates p_1, \dots, p_k to the pairs of object sets $(\Omega_{A,1}, \Omega_{B,1}), \dots, (\Omega_{A,K}, \Omega_{B,K})$.

For example, a complex structure like a semantic ontology such as WordNet can be easily accommodated into this definition. Let us set every $\Omega_{A,1}, \dots, \Omega_{A,K}$ and every $\Omega_{B,1}, \dots, \Omega_{B,K}$ to be one and the same: the vocabulary of words in the semantic ontology. Moreover, let us define the relations r_1, \dots, r_K and their corresponding logical predicates p_1, \dots, p_k to be true if and only if two words are connected by a typed link in the ontology, such as synonymy, hypernymy, hyponymy etc.

Then the structure s is simply the union of all word pairs that are linked by at least one relation r_k

Again, note that structure precedes the notion of frequency. Structure only defines some set of relations that exist between objects and that can be operationalized to define a context for a representation learning model. It is only when that context is used to perform counting and processing that frequency becomes tied to the model.

By considering the two definitions of relation and structure together, we obtain the following corollary.

Corollary 1. *Structure is decomposable into a set of objects and relations.*

With these definitions and corollary, we note that relations and structure are in fact two inherently interconnected phenomena.

In addition to accommodating the hypotheses (such as the distributional and bag-of-words hypotheses) in the examples above, these definitions also encompass the different linguistic phenomena discussed in the context of semantic representation learning.

For example, composition is a recursive relation that operates on vocabulary sets of ever greater granularity. If we consider Ω^A and Ω^B to both be vocabularies of individual words, then composition of two words is the a subset of the cross-product $\Omega^A \times \Omega^B$. Similarly if Ω^A is a vocabulary of words but now Ω^B is the set of two word compositions, three word composition is a relation that defines a subset of the new cross-product $\Omega^A \times \Omega^B$. It is also clear that the application of every recursive composition relation produces a structure of ever greater granularity.

In fact it might seem like a relation could be applied to interpret practically any linguistic phenomenon. And this we believe is, in fact, true. Figure 1.1 illustrates just some of the sources of information where we discover and use structure and relations to learn semantic representations in this thesis.

Relations and structure in language are ubiquitous. Turney (2007) makes the point that relations are primary to producing meaning in language. He argues that even an attributive adjective like the color “red” is in fact a relation, a property that is “redder” than something else. Relations can even be defined over other relations. Such meta-relations can capture complex linguistic phenomena like describing the inherently linked acts of buying and selling, and the repercussions on ownership of goods and money.

Putting relations at the center of our investigation allows us to come up with a common overarching theme to the second (and more important, for this thesis) of our questions regarding semantic representation learning, “What does the model represent?”. Answers to this question are simply instances of different relations that are captured and learned by models. Each distinct semantic hypothesis is, in fact, specifying a different relational phenomenon.

At the same time, different problems require different linguistic information to be encoded. And thus understanding, defining and leveraging the correct (or at least more expressive) relation and structure in learning semantic representations for these problem is, centrally, the contribution of this thesis.

SOURCE	STRUCTURE	OBJECTS	RELATIONS				
DEPENDENCY RELATIONS		i, eat	doobj				
ONTOLOGIES		collie, dog, animal	hyponym, hypernym				
TABLES	<table border="1"> <tr> <td>mason</td> <td>stone</td> </tr> <tr> <td>carpenter</td> <td>wood</td> </tr> </table>	mason	stone	carpenter	wood	<div style="display: flex; flex-direction: column; align-items: center;"> <div style="display: flex; gap: 20px;"> <div style="border: 1px solid black; padding: 2px;">stone</div> <div style="border: 1px solid black; padding: 2px;">mason</div> </div> <div style="border: 1px solid black; padding: 2px; margin: 5px 0;">carpenter</div> <div style="border: 1px solid black; padding: 2px;">wood</div> </div>	Latent but enumerable, e.g. (belong to same row) (belong to same col)
mason	stone						
carpenter	wood						
PATTERNS	X causes Y Y caused by X	X, Y	Latent and un-enumerable				
PREDICATE ARGUMENT PAIRS	eat → pasta sing → ballad watch → TV look → telescope	PREDS, ARGS	Latent and un-enumerable				

Figure 1.1: Examples illustrating various sources of information exhibiting the dual nature of structure and relations. Note how Definitions 1 and 2, as well as Corollary 1 apply to each of these examples. All these sources of relations and structure are used in this thesis.

1.1 Contributions of this Thesis

Formally, we propose the central conjectures of this thesis:

1. Relations are central to producing and understanding meaning in language.
2. Semantic representation learning models can benefit from an over-arching view of *what* they encode, by understanding the key relations that they capture.
3. Different linguistic phenomena require different relations and structure to be optimally understood and represented.
4. Integrating the right relations and structure for a particular problem often leads to more expressive and empirically superior models of semantic representation learning.

This thesis substantiates these conjectures with evidence for several different semantic problems. The problems are selected to represent a broad range of linguistic phenomena and sources of relational and structural knowledge. They demonstrate the general applicability of our hypotheses. The problems also provide an example-based narrative which we study to suggest a framework within which to think about and implement solutions for integrating relations and structure in semantic representation learning.

In Chapter 2 we introduce our framework as a series of five research questions, each tackling a sequential set of themes related to relational modelling in representation learning. We suggest possible answers to these research questions by citing instances from this thesis as well as analyzing the different aspects related to each theme. In summary the five themes deal with: 1) formulating a representation learning problem centered around the question “What is being represented?”, 2) finding a source of information that encodes knowledge about the learning problem, 3) decomposing the information into a set of minimal relations and noting any structural patterns, 4) formulating an objective or model for the learning problem that integrates the relations and structure of the knowledge resource, and 5) characterizing the nature of the learned model and evaluating it appropriately to justify the central claims of this thesis – namely that using relations and structure in representation learning leads to more expressive and empirically superior models of semantics.

The rest of the thesis is organized as instantiations of this high level framework for several different semantic problems. While all these chapters fall into the broader theme of this thesis, we can divide them organizationally into two distinct parts.

The first part, consisting of Chapters 3 and 4 deal with leveraging existing structural and relational data to improve representation learning models for different problems. In the second part, which contains Chapters 5 and 6, we will investigate problems where there is no existing structure or relational data. Rather we will attempt to induce structure and relations and embed representation learning models with them. The contributions of each of these chapters are summarized in what follows.

Chapter 3 introduces work published in Jauhar et al. (2015). It deals with the problem of polysemy in semantic representation learning. In particular it introduces two models that learn vectors for polysemous words grounded in an ontology. The structure of the ontology is the source of relational information that we use to guide our techniques. The first model acts as a simple post-processing step that teases apart the different sense vectors from a single sense

vector learned by any existing technique on unannotated corpora. The second builds on the first and introduces a way of modifying maximum likelihood representation learning models to also account for senses grounded in an ontology. We show results on a battery of semantic tasks, including two additional sets of results not published in Jauhar et al. (2015) – one of which shows that our model can be applied to deal with antonymy as well.

In Chapter 4 we deal with the semi-structured representation of knowledge tables for the task of question answering. Our paper, Jauhar et al. (2016a), places relations at the center of our investigation. First, we leverage the relations between cells in tables to guide non-expert annotators to create a large dataset of high quality Multiple-Choice Questions (MCQs) with minimal effort (Jauhar et al., 2016b). Because of the set-up of our task, we obtain fine-grained alignments between MCQs and table elements as a by-product. We also introduce two different models that solve multiple-choice questions by using tables as a source of background knowledge. Crucially, both solutions leverage the structure and relations present within tables to solve MCQs. The first solution is a manually engineered feature-rich model, whose features are based on the structural properties of tables. The second solution extends the intuition of the feature-rich model with a neural network that automatically learns representations over cells.

We look at the problem of relation induction in Chapter 5, within the context of embedded semantic lexicons. This work is published in Jauhar and Hovy (2017). It represents a novel integration between a predictive maximum likelihood model and an Indian Buffet Process to induce an embedded semantic lexicon that contain a set of latent, automatically generated relation slots. A key contribution of this model is that it requires no high-level annotations – such as dependency information – and can learn slots by only accessing a corpus of predicate-argument pairs. Our proposed solutions jointly leverages two types of relations: local token level affinities as well as global type level affinities between predicates and arguments to arrive at an understanding of the latent slot structure. Quantitatively as well as qualitatively, we show that the induced latent slots produce an embedded lexicon that is semantically expressive and that captures several interesting properties of known, manually defined semantic relations.

Chapter 6 deals with the problems of contextual variability of word meaning, and faceted comparison in semantics. We introduce two solutions for these problems, both bound by a common modelling paradigm: tensor based representation, where the semantic signature of a standard word vector is decomposed into distinct relation dimensions. We instantiate this modelling paradigm with two kinds of relations: syntactic and latent semantic relations, each leading to a solution for the two central problems of this chapter. The models we develop are published in Goyal et al. (2013a,b) and Jauhar and Hovy (2014).

Finally, in Chapter 7, we conclude with a look back at the contributions of this thesis and suggest avenues for future research that merit investigation.

Chapter 2

Relational Representation Learning

A Framework for Using Semantic Relations in Lexical Representation Learning

Given our definitions of relations and structure in Definition 1 and 2, a very diverse range of linguistic properties, knowledge resources and computational phenomena can be said to exhibit relational and structural attributes. Moreover, there are many different ways that these phenomena can be integrated into existing methods of representation learning.

Hence, within the scope of our goal of proving a relation-centric view of semantic representation learning, it is difficult to come up with a single mathematical framework through which all relationally enriched semantic representation learning techniques may be viewed. Even if we were to propose such a framework, it isn't clear whether it would simplify an understanding of the diverse range of phenomena that fall under the umbrella terms of relations and structure.

Instead, we propose an example-based scheme to illustrate our use of relations in learning semantic models. The scheme sequentially highlights a number of recurring themes that we encountered in our research, and we structure our illustration as a series of questions that address these themes. The question-theme structure allows us to focus on observationally important information while also proposing an outline for how to apply our example-based framework to a new domain. The examples we cite from this thesis for each theme will illustrate the diverse range and flexibility of integrating relations in representation learning.

In summary, the five research questions we address, along with the themes that they expose are:

1. What is the semantic representation learning problem we are trying to solve?

Identify a target representation learning problem that focusses on what (linguistic) property needs to be modelled.

2. What linguistic information or resource can we use to solve this problem?

Find a corpus-based or knowledge-based resource that contains information relevant to solving the problem in item 1.

3. What are the relations present in this linguistic information or resource?

Discover useful structure that is present in the linguistic annotation or knowledge resource, and determine the set of relations that comprise this structure. This sometimes may be

explicit or latent.

4. How do we use the relations in representation learning?

Think about an objective or model that intuitively captures the desiderata of using the relations or structure in the resource from item 2 to solve the problem in item 1.

5. What is the result of including the relations in learning the semantic model?

Evaluate the learned model to see if the inclusion of relational information has measurably influenced the problem in item 1. Possibly characterize the nature of the learned model to gain insight on what evaluation might be appropriate.

We should stress that the set of questions and themes we describe is by no means complete and absolute. While useful as guidelines to addressing a new problem, they should be complemented by in-domain knowledge of the problem in question.

This chapter should be viewed as a summary of the thesis as well as a framework for applying our ideas to future work. The framework is outlined by the research agenda listed above. In the rest of this chapter we delve further into each of the five questions of our research agenda.

In the section detailing each of the five research questions, the final paragraph – which is formatted as a quotation block, for visual clarity – summarizes the high-level findings of this thesis as they relate to the respective questions. If only interested in these high-level findings the reader should feel free to skip over the examples from the thesis that we list in order to flesh out the findings. Skipping over those parts of this chapter will not affect the comprehensibility of the rest of the thesis.

What is the semantic representation learning problem we are trying to solve?

As detailed in our introduction this is a thesis about representation learning that focusses more on the question of *what* is being learned, rather than *how* it is being learned. Hence, the first and most fundamental question needs to address the issue of what semantic representation learning problem is being tackled.

Many existing methods for semantic representation learning operate on the intuition of the distributional hypothesis (Firth, 1957). Namely, that words that occur in similar contexts tend to have similar meanings. But as we have argued in Chapter 1, this hypothesis – while extremely useful – does not cover the the entire spectrum of natural language semantics. In fact, every different semantic understanding problem requires a specific relation and corresponding *context* to properly express and represent it.

In this thesis, for example, we tackle several different problems that do not fit into the mould of the word-neighborhood context that is engendered by the distributional hypothesis. We address the problem of contextual variability (Blank, 2003) in Chapter 6.2, where the meaning of a word varies and stretches based on the neighborhood context in which it occurs¹. Another problem with semantic vector space models is the facetedness in semantic comparison (Jauhar

¹Not to be confused with polysemy, which Blank (2003) contrasts with contextual variability in terms of a lexicon. The former is characterized by different lexicon entries (e.g. “human arm” vs “arm of a sofa”), while the latter is the extension of a single lexicon entry (e.g. “human arm” vs “robot arm”). Naturally, the distinction is a continuum that depends on the granularity of the lexicon.

and Hovy, 2014), whereby concepts cannot be compared along different facets of meaning (e.g. color, material, shape, size, etc.). We tackle this problem in Chapter 6.3. Yet another issue with distributional semantic models is their incapacity to learn polysemy from un-annotated data, or to learn abstract types that are grounded in known word senses. We propose a joint solution to both problems in Chapter 3.

Not all problems are necessarily linguistic shortcomings. For example, our work in Chapter 5 is motivated as a shortcoming of data and linguistic resources in learning a semantic lexicon, not of the representation problems themselves.

Another example is our work in Chapter 4, which tackles a representational issue, not a linguistic one: namely the trade-off between the expressive power of a representation and the ease of its creation or annotation. Hence our research focusses on semi-structured representations (Soderland, 1999) and the ability to learn models over and perform structured reasoning with these representations. In similar fashion, the models we propose in Chapter 6 – while tackling specific linguistic issues – are also concerned with schemas: moving from vectors to tensors as the unit of representation Baroni and Lenci (2010).

In summary, the first step in our sequence of research questions is to address the central issue of what is being represented. Any problem is legitimate as long as it focusses on this central research question. In our experience, generally problems will stem from issues or shortcomings in existing models to represent complex linguistic phenomena, but they could also concern (sometimes by consequence) the form of the representation itself.

What linguistic information or resource can we use to solve this problem?

Once a representation learning problem has been identified, the next item on the research agenda is to find a source of knowledge that can be applied to the problem. Naturally this source of knowledge should introduce information that helps to solve the central problem.

In our experience, there seem to be two broad categories of such sources of knowledge. First there is corpus-based information that is easily and abundantly available as naturally occurring free text, or some level of linguistic annotation thereof. For example, this category would cover free text from Wikipedia, it's part-of-speech tags, parse trees, coreference chains, to name only a few. Second are knowledge-based resources that are compiled in some form to encapsulate some information of interest. For example, these could be ontologies, tables, gazetteer lists, lexicons, etc.

The two categories are, of course, not mutually exclusive: for example a treebank could be reasonably assumed to belong to both categories. Moreover there is no restriction on using a single source of knowledge. Any number of each category can be combined to bring richer information to the model.

Examples of the first category of sources of information in this thesis are dependency information (Nivre, 2005) (Chapter 6.2 and Chapter 5), textual patterns (Ravichandran and Hovy, 2002) (Chapter 6.3), and predicate-argument pairs (Marcus et al., 1994) (Chapter 5). Similarly, examples of the second category of sources of knowledge include ontologies (Miller, 1995) (Chapter 3) and tables (Chapter 4).

In each of these cases we selected these sources of knowledge to bring novel and useful information to the problem we were trying to resolve. For example, ontologies contain word sense inventories that are useful to solving the problem of polysemy in representation learning. Similarly dependency arcs define relational neighborhoods that are useful for creating structured distributional semantic models or yielding features that can be used to find the principal relational arguments of predicates. Predicate-argument structures provide weak, latent cues that can be aggregated to arrive at an understanding of semantic frames.

To sum up, the goal of the second research question is to identify a resource or resources that can be used to resolve the central representation learning problem proposed in response to the first research question. In our experience resources fall into two broad (possibly overlapping) categories: corpus-based and knowledge-based.

What are the relations present in this linguistic information or resource?

With a well defined set of resources (both corpus-based and knowledge-based) in place, the next question seeks to identify the principal relations present in each of the resources. At face value, this may seem like a simple exercise to list out and analyze the relations in a resource. And this is, in fact, the simplest use case for well-defined carefully crafted resources.

In this thesis, the most direct uses of structure and relations involves our work on ontologies (Chapter 3). Here the ontology is a carefully conceived graph structure with typed edges that relate word senses to one another. Our work on tables (Chapter 4) also largely falls into this category, where the structure is constant and well-defined. However, the structure itself is not typed or even explicit. Rather, it stems from the inherent relations between table cells, which interact with one another in a number of interesting ways (see Chapter 4.2.1). Analogies (Turney, 2013), for example, are one type of semantic property that is naturally embedded in tables, and comes directly from the relations between table cells.

However, in the absence of such well-defined structure, there are several challenges to identifying and using the relations in a resource.

When the relations are latent rather than explicit, the goal of the model may be to discover or characterize these relations in some fashion. This is not the case with our work on tables, where relational properties between table cells are only leveraged but are not required to be directly encoded in the model. In contrast our work on latent tensor models (Chapter 6.3) and proposed event argument induction (Chapter 5) both require the latent relations to be explicitly learned as parts of the resulting representations.

There is also the problem of resources that do not specify a closed class of definite relations at all. This is the case with our work on Structured Distributional Semantics (Chapter 6.3), for example, when we attempt to extract relational patterns from free text. There is simply no way of enumerating all possible patterns beforehand, and even the patterns that are harvested number in the many millions and are consequently very difficult to manage. Similarly, in our work on Lexicon Induction (Chapter 5), the set of relations between predicate-argument types cannot be enumerated beforehand.

However, identifying properties of the relations and structure can lead to insight that makes the space of un-enumerable relations more manageable. For example, the pattern relations or predicate-argument relations are likely to follow a Zipfian distribution. This insight can be used

to select and only use the most frequent relation that occur in the data. Similarly when one is considering relations in a dependency parse tree (Chapter 6.2), it may be useful – for example – to note that the tree is projective (Nivre, 2003). In general, insights into these facets can lead to computationally desirable models or algorithms, as we will note in the our next research question.

There is, of course, an inherent trade-off between the linguistic complexity or abstractness of a relation, and its ability to be harvested or leveraged for learning word vectors. The distributional hypothesis, for example, encodes a simple relation that can be abundantly harvested from unannotated text. More complex relations, such as those defined over ontologies (Chapter 3) or tables (Chapter 4) are fewer and harder to directly leverage. In this thesis we sometimes supplement the distributional hypothesis with other forms of relational information (such as ontological relations), while at others, completely replacing it with some other form of more structured data (such as tables). We show that the inherent tension between complexity and availability in the data is resolved because even small amounts of structured data provide a great amount of valuable information.

In conclusion, identifying the relations in resources and the inherent structure that they define are important to gain insight into how best to integrate them into the representation learning framework. Specific kinds of structure, such as graphs or trees for example, have well-defined relational structure. These can lead to computational models or algorithms that can leverage the structural properties through principled approaches specifically designed for them. More generally, the wealth of research in structured prediction (Smith, 2011) is applicable to the scope of our problems since it deals with learning and inference over complex structure that is often decomposable into more manageable constituents, – which is exactly how we define the link between relations and structure in Definitions 1 and 2, and Corollary 1.

Finally, when the relations and their resulting structure are latent or un-enumerable, it may be required to induce representations that encode and consolidate the latent or un-enumerable relations into the model. In these cases, too, insights into the properties of the relations and underlying structure can help in developing models or learning solutions that are more computationally attractive and practical.

How do we use the relations in representation learning?

The next research question on the agenda deals with the actual modelling problem of integrating the relations or structure of interest into a well-defined representation learning problem. Here there is considerable flexibility, since there is no single correct way of integrating intuitions and observations about the target domain into model – and no guarantee that once one does, that it will function as expected.

Nevertheless, we draw a distinction here between defining an objective and a model. The former attempts to encode some mathematical loss function or transformation (such as maximum likelihood, or singular value decomposition) that captures the desiderata of a problem, while the latter specifies a model that will numerically compute and optimize this loss function (for example a recursive neural network or a log-linear model). The two are sometimes connected, and the definition of one may lead to the characterization of the other.

We note that our purpose for drawing this distinction is merely for the sake of illustrating our example-based framework. Hence, we will discuss the objective or model, as appropriate, when either one or the other more intuitively captures the use of relations or structure in the representation learning problem.

In this thesis we work with several kinds of transformations and objectives. An example of transformation based representation learning is singular value decomposition (Golub and Reinsch, 1970), which we use to reduce sparsity and noise in learning latent relations for our tensor model (Chapter 6.3).

Many existing semantic representation learning models are formulated as maximum likelihood estimation problems over corpora (Collobert et al., 2011; Mikolov et al., 2010). In our work with ontologies we propose a way to modify these objectives minimally with structured regularizers, thereby integrating knowledge from the ontological graph (Chapter 3). A very similar formulation is also applied to our work on semantic lexicon induction (Chapter 5), where we maximize the likelihood of a corpus of predicate-argument pairs, while accounting for a structured regularizer that is defined over predicate embedding representations.

Also in our work with ontologies, post-processing is another method that we use to modify word vectors and derive sense representations. In this case, a maximum a-posteriori estimation over smoothness properties of the graph (Corduneanu and Jaakkola, 2002; Zhu et al., 2003; Subramanya and Bilmes, 2009) is more appropriate. Specifically we have a prior (the existing word vectors) which we'd like to modify with data (the ontological graph) to produce a posterior (the final sense vectors).

In the case of tables, we are dealing with a countable set of predictable relations over table cells. Featurizing these relations is the most direct way of encoding them into learning our model. Consequently we use a log-linear model (Chapter 4.5) which conveniently captures features (Smith, 2004). We can also leverage these relations to automatically induce features, which we do with a neural network model over table cells (Chapter 4.6). For both kinds of models, we use a cross-entropy loss.

In characterizing the argument structure of predicates in a semantic lexicon (Chapter 5), we wish to allow for a potentially infinite set of relations, while letting the data decide what a really meaningful set of relations should be. Hence we use the Indian Buffet process (Griffiths and Ghahramani, 2005), which is a Bayesian non-parametric model (Hjort et al., 2010) to precisely model feature learning in such a setting.

Summarizing the theme of this part of the research agenda, we seek to concretize a specific model that efficiently and usefully integrates knowledge from our source of relations and the structure that it entails. The goal remains to solve the representation learning problem we originally defined. While there is a lot of flexibility in how this may be done, it can be useful to think about two aspects of the problem: the objective function and the actual model. Often one or the other presents an intuitive way of characterizing how relations or structure should be integrated, and the other follows directly or is left to empirical validation.

What is the result of including the relations in learning the semantic model?

Our final research item deals with the model after it has been learned. Specifically we discuss some differences in the models we learn in this thesis in an attempt to characterize them, and we also list how we evaluate them.

With regards to the learned model, there are several distinction. The most salient distinction, however, is between the case where relations have simply been leveraged as an external influencer on learning models, versus cases where relations become part of the resulting model. In this thesis examples of the first case can be found in Chapters 3 and 4, while examples of the second case are Chapter 5 and Chapter 6.

In terms of evaluation, when specifically dealing with the latter of the two cases it is useful to evaluate the relations in the model in addition to the representations for vocabulary elements (which should be evaluated anyway). We do this in this thesis with a relation classification task (Zhang, 2004) in Chapter 6.3 and two different evaluations on local token-level and global type-level relations between predicates and arguments in Chapter 5. Additionally, we qualitatively evaluate the relations in both relevant chapters by mapping them to know, manually-defined semantic relations to better understand how they represent information.

For the sake of completeness of our summary of the work in this thesis, we also list some less important distinctions about the models we learn. These need not be considered, in general when characterizing a learned model or thinking about how it will be evaluated. Most often semantic representation learning yields vectors, which is the case for our work on ontologies, tables and semantic lexicons (in Chapters 3, 4, and 5 respectively). But we also introduce tensor based semantic models (Chapter 6). In both kinds of objects, the numerical values that comprise them are most often abstract real numbers, as is the case with our models from Chapters 6.3, 3 and 4. But we also have models that are probabilistically normalized (Chapter 6.2) and binary (Chapter 5).

On the evaluation side, depending on the nature of the model and it's connection to the problem that it is attempting to solve, it is also useful to consider an intrinsic or an extrinsic evaluation. For general purpose semantic word vector (or tensor) models, such as the ones learned in Chapters 3 and 6 popular intrinsic evaluation tasks such as *word similarity judgement* (Finkelstein et al., 2002) and *most appropriate synonym selection* (Rubenstein and Goodenough, 1965) are adequate. For targeted models such as the one we learn over tables specifically created for question answering (Chapter 4), an extrinsic task evaluation is more appropriate. This is also the case for the evaluations that we conduct for our work on predicate-argument structure (Chapter 5).

In summary, most importantly, – as with any good empirical task – the evaluation should directly evaluate whether the inclusion of relations in the model measurably influences the problem we set out to solve. Sometimes, characterizing the nature of the learned model can lead to insights of what might be an appropriate evaluation.

In the rest of this thesis we detail instances of the application of this research agenda to several research problems and proposed future work.

Chapter 3

Ontologies and Polysemy

Using Ontological Relation Links to Learn Sense-specific Word Representations of Polysemy

Most approaches to representation learning for lexical semantics assign a single vector to every surface word type. This is a problem we encounter in Chapter 6.3 with respect to the issue of the multiple facets or relational arguments of concepts.

In this chapter we tackle a different problem, which however has its roots in the same representational issue with the vector space model. Namely, that words – across all human languages – are polysemous.

Models that assign a single representation to a unique surface form (such as vector space models) ignore the possibility that words may have more than one meaning. For example, the word “bank” can either denote a financial institution or the shore of a river.

The ability to model multiple meanings is an important component of any NLP system, given how common polysemy is in language. For example, knowing which of the senses of a word needs to be used in a Machine Translation system (Carpuat and Wu, 2007) to translate into another language can make the difference between a correct translation and a nonsensical one.

Much of the attractiveness of vector space models of semantics are their ability to be learned in an unsupervised fashion from vast amounts of unannotated data. Thus it is difficult for them to capture the different senses of words, when these senses are not annotated to begin with.

The lack of sense annotated corpora large enough to robustly train VSMs, and the absence of fast, high quality word sense disambiguation (WSD) systems makes handling polysemy difficult in the representation learning of semantics.

Meanwhile, lexical ontologies, such as WordNet (Miller, 1995) specifically catalog sense inventories and provide typologies that link these senses to one another. These ontologies typically encode structure and connectedness between concepts through relations, such as synonymy, hypernymy and hyponymy. Data in this form provides a complementary source of information to distributional statistics. Its relation-induced semantic structure also falls into the over-arching theme of this thesis as a potential source of structured knowledge that can be leveraged to learn more expressive models of semantics.

Recent research tries to leverage ontological information to train better VSMs (Yu and Dredze, 2014; Faruqui et al., 2014), but does not tackle the problem of polysemy. Parallely, work on polysemy for VSMs revolves primarily around techniques that cluster contexts to distinguish between different word senses (Reisinger and Mooney, 2010; Huang et al., 2012), but does not integrate ontologies in any way.

In this chapter we present two novel approaches to integrating ontological and distributional sources of information. Our focus is on allowing already existing, proven techniques to be adapted to produce ontologically grounded word sense embeddings.

Our first technique is applicable to any sense-agnostic VSM as a post-processing step that performs graph propagation on the structure of the ontology. The second is applicable to the wide range of current techniques that learn word embeddings from predictive models that maximize the likelihood of a corpus (Collobert and Weston, 2008; Mnih and Teh, 2012; Mikolov et al., 2013a). Our technique adds a latent variable representing the word sense to each token in the corpus, and uses EM to find parameters. Using a structured regularizer based on the ontological graph, we learn grounded sense-specific vectors.

Both techniques crucially rely on the structure of the underlying ontology, and the relations that bind related word senses to one another.

There are several reasons to prefer ontologies as distant sources of supervision for learning sense-aware VSMs over previously proposed unsupervised context clustering techniques. Clustering approaches must often parametrize the number of clusters (senses), which is neither known a priori nor constant across words (Kilgarriff, 1997). Also the resulting vectors remain abstract and un-interpretable. With ontologies, interpretable sense vectors can be used in downstream applications such as WSD, or for better human error analysis. Moreover, clustering techniques operate on distributional similarity only whereas ontologies support other kinds of relationships between senses. Finally, the existence of cross-lingual ontologies would permit learning multi-lingual vectors, without compounded errors from word alignment and context clustering.

We evaluate our methods on 4 lexical semantic tasks across 9 datasets and 2 languages, and show that our sense-specific VSMs effectively integrate knowledge from the ontology with distributional statistics. Empirically, this results in consistently and significantly better performance over baselines in most cases. We discuss and compare our two different approaches to learning sense representations from the perspectives of performance, generalizability, flexibility and computational efficiency. Finally, we qualitatively analyze the vectors and show that they indeed capture sense-specific semantics.

3.1 Related Work

Since Reisinger and Mooney (2010) first proposed a simple context clustering technique to generate multi-prototype VSMs, a number of related efforts have worked on adaptations and improvements relying on the same clustering principle. Huang et al. (2012) train their vectors with a neural network and additionally take global context into account. Neelakantan et al. (2014) extend the popular skip-gram model (Mikolov et al., 2013a) in a non-parametric fashion to allow for different number of senses for words. Guo et al. (2014) exploit bilingual alignments to perform better context clustering during training. Tian et al. (2014) propose a probabilistic ex-

tension to skip-gram that treats the different prototypes as latent variables. This is similar to our second EM training framework, and turns out to be a special case of our general model. In all these papers, however, the multiple senses remain abstract and are not grounded in an ontology.

Conceptually, our work is also similar to Yu and Dredze (2014) and Faruqui et al. (2014), who treat lexicons such as the Paraphrase Database (PPDB) (Ganitkevitch et al., 2013) or WordNet (Miller, 1995) as an auxiliary thesaurus to improve VSMs. However, they do not model senses in any way. Pilehvar et al. (2013) do model senses from an ontology by performing random-walks on the WordNet graph, however their approach does not take distributional information from VSMs into account.

Thus, to the best of our knowledge, our work presents the first attempt at producing sense grounded VSMs that are symbolically tied to lexical ontologies. From a modelling point of view, it is also the first to outline a unified, principled and extensible framework that effectively combines the symbolic and distributional paradigms of semantics by specifically paying attention to relational structure.

Both our models leverage the graph structure of and relational links in ontologies to effectively ground the senses of a VSM. This ties into previous research (Das and Smith, 2011; Das and Petrov, 2011) that propagates information through a factor graph to perform tasks such as frame-semantic parsing and POS-tagging across languages. More generally, this approach can be viewed from the perspective of semi-supervised learning, with an optimization over a graph loss function defined on smoothness properties (Corduneanu and Jaakkola, 2002; Zhu et al., 2003; Subramanya and Bilmes, 2009).

Related to the problem of polysemy is the issue of different shades of meaning a word assumes based on context. The space of research on this topic can be divided into three broad categories: models for computing contextual lexical semantics based on composition (Mitchell and Lapata, 2008; Erk and Padó, 2008; Thater et al., 2011), models that use fuzzy exemplar-based contexts without composing them (Erk and Padó, 2010; Reddy et al., 2011), and models that propose latent variable techniques (Dinu and Lapata, 2010; Séaghdha and Korhonen, 2011; Van de Cruys et al., 2011). Chapter 6 addresses the first of the three categories with a Structured Distributional Semantic Model that takes relational context into account. The current chapter, however, tackles the stronger form of lexical ambiguity in polysemy and falls into the latter two of three categories.

3.2 Unified Symbolic and Distributional Semantics

In this section, we present our two techniques for inferring sense-specific vectors grounded in an ontology. We begin with notation. Let $W = \{w_1, \dots, w_n\}$ be a set of word types, and $W_s = \{s_{ij} \mid \forall w_i \in W, 1 \leq j \leq k_i\}$ a set of senses, with k_i the number of senses of w_i . Moreover, let $\Omega = (T_\Omega, E_\Omega)$ be an ontology represented by an undirected graph. The vertices $T_\Omega = \{t_{ij} \mid \forall s_{ij} \in W_s\}$ correspond to the word senses in the set W_s , while the edges $E_\Omega = \{e_{ij-i'j'}^r\}$ connect some subset of word sense pairs $(s_{ij}, s_{i'j'})$ by semantic relation r .

Here’s an example of the kind of data we will be dealing with. Given words such as “cat” and “feline” in our vocabulary, we might have associated senses such as “cat(1)” (the animal), “cat(2)” (the Linux command), and “feline(1)”. Our ontology would then encode an edge de-

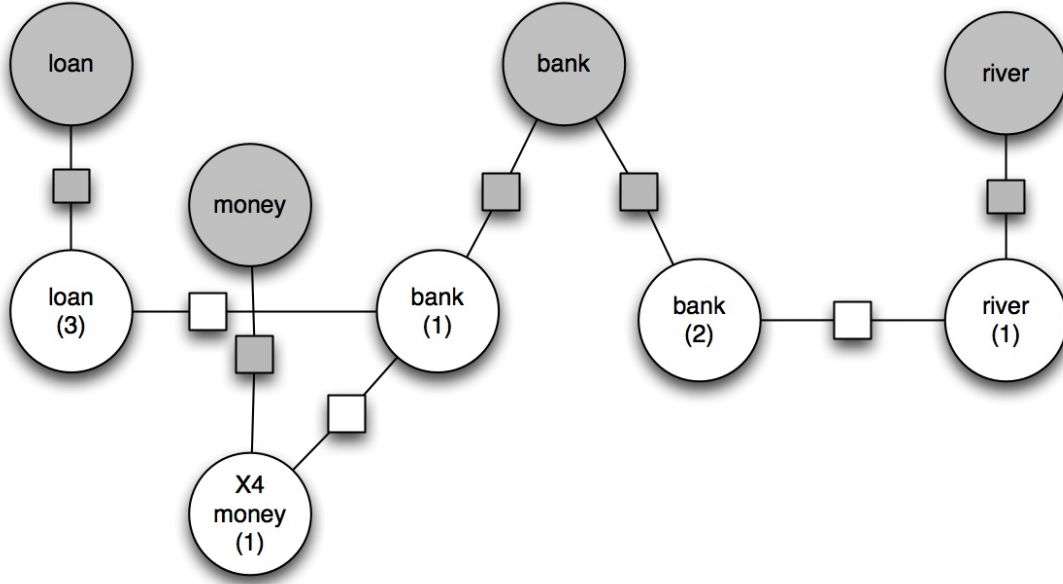


Figure 3.1: A factor graph depicting the retrofitting model in the neighborhood of the word “bank”. Observed variables corresponding to word types are shaded in grey, while latent variables for word senses are in white.

noted “synonym” between “cat(1)” and “feline(1)”.

3.2.1 Retrofitting Vectors to an Ontology

Our first technique assumes that we already have a vector space embedding of a vocabulary of *surface forms* $\hat{U} = \{\hat{u}_i | \forall w_i \in W\}$. We wish to infer vectors $V = \{v_{ij} | \forall s_{ij} \in W_s\}$ for *word senses* that are maximally consistent with both \hat{U} and Ω , by some notion of consistency. We formalize this notion as MAP inference in a Markov network (MN).

The MN we propose contains variables for every vector in \hat{U} and V . These variables are connected to one another by dependencies as follows. Variables for vectors v_{ij} and $v_{i'j'}$ are connected iff there exists an edge $e_{ij-i'j'}^r \in E_\Omega$ connecting their respective word senses in the ontology. Furthermore, vectors \hat{u}_i for the word types w_i are each connected to all the vectors v_{ij} of the different senses s_{ij} of w_i . If w_i is not contained in the ontology, we assume it has a single unconnected sense and set its only sense vector v_{i1} to its empirical estimate \hat{u}_i .

The structure of this MN is illustrated in Figure 3.1, where the neighborhood of the ambiguous word “bank” is presented as a factor graph.

We set each pairwise clique potential to be of the form $\exp(a\|u - v\|^2)$ between neighboring nodes. Here u and v are the vectors corresponding to these nodes, and a is a weight controlling the strength of the relation between them. We use the Euclidean norm instead of a distance based on cosine similarity because it is more convenient from an optimization perspective.¹

¹It can be shown, in fact, that using the Euclidean norm turns out to be a special case of the constrained problem with cosine log factors. Moreover, vectors that are spatially close – in the Euclidean sense – also have small cosine

Our inference problem is to find the MAP estimate of the vectors V , given \hat{U} , which may be stated as follows:

$$C(V) = \arg \min_V \sum_{i-ij} \alpha \|\hat{u}_i - v_{ij}\|^2 + \sum_{ij-i'j'} \beta_r \|v_{ij} - v_{i'j'}\|^2 \quad (3.1)$$

Here α is the sense-agnostic weight and β_r are relation-specific weights for different semantic relations. This objective encourages vectors of neighboring nodes in the MN to pull closer together, leveraging the tension between sense-agnostic neighbors (the first summation term) and ontological neighbors (the second summation term). This allows the different neighborhoods of each sense-specific vector to tease it apart from its sense-agnostic vector.

Taking the partial derivative of the objective in equation 3.1 with respect to vector v_{ij} and setting to zero gives the following solution:

$$v_{ij} = \frac{\alpha \hat{u}_i + \sum_{i'j' \in \mathcal{N}_{ij}} \beta_r v_{i'j'}}{\alpha + \sum_{i'j' \in \mathcal{N}_{ij}} \beta_r} \quad (3.2)$$

where \mathcal{N}_{ij} denotes the set of neighbors of ij . Thus, the MAP sense-specific vector is an α -weighted combination of its sense-agnostic vector and the β_r -weighted sense-specific vectors in its ontological neighborhood.

Algorithm 1 Outputs a sense-specific VSM, given a sense-agnostic VSM and ontology

```

1: function RETROFIT( $\hat{U}, \Omega$ )
2:    $V^{(0)} \leftarrow \{v_{ij}^{(0)} = \hat{u}_i \mid \forall s_{ij} \in W_s\}$ 
3:   while  $\|v_{ij}^{(t)} - v_{ij}^{(t-1)}\| \geq \epsilon \forall i, j$  do
4:     for  $t_{ij} \in T_\Omega$  do
5:        $v_{ij}^{(t+1)} \leftarrow$  update using equation 3.2
6:     end for
7:   end while
8:   return  $V^{(t)}$ 
9: end function

```

We use coordinate descent to iteratively update the variables V using equation 3.2. The optimization problem in equation 3.1 is convex, and we normally converge to a numerically satisfactory stationary point within 10 to 15 iterations. This procedure is summarized in Algorithm 1. The generality of this algorithm allows it to be applicable to any VSM as a computationally attractive post-processing step.

An implementation of this technique is available at <https://github.com/sjauhar/SenseRetrofit>.

distances (and high cosine similarity) between them, anyway.

3.2.2 Adapting Predictive Models with Latent Variables and Structured Regularizers

Many successful techniques for semantic representation learning are formulated as models where the desired embeddings are parameters that are learned to maximize the likelihood of a corpus (Collobert and Weston, 2008; Mnih and Teh, 2012; Mikolov et al., 2013a). In our second approach we extend an existing probability model by adding latent variables representing the senses, and we use a structured prior based on the topology of the ontology to ground the sense embeddings. Formally, we assume a corpus $D = \{(w_1, c_1), \dots, (w_N, c_N)\}$ of pairs of target and context words, and the ontology Ω , and we wish to infer sense-specific vectors $V = \{v_{ij} \mid \forall s_{ij} \in W_s\}$.

Consider a model with parameters $\theta (V \in \theta)$ that factorizes the probability over the corpus as $\prod_{(w_i, c_i) \in D} p(w_i, c_i; \theta)$. We propose to extend such a model to learn ontologically grounded sense vectors by presenting a general class of objectives of the following form:

$$C(\theta) = \arg \max_{\theta} \sum_{(w_i, c_i) \in D} \log\left(\sum_{s_{ij}} p(w_i, c_i, s_{ij}; \theta)\right) + \log p_{\Omega}(\theta) \quad (3.3)$$

This objective introduces latent variables s_{ij} for senses and adds a structured regularizer $p_{\Omega}(\theta)$ that grounds the vectors V in an ontology. This form permits flexibility in the definition of both $p(w_i, c_i, s_{ij}; \theta)$ and $p_{\Omega}(\theta)$ allowing for a general yet powerful framework for adapting MLE models.

In what follows we show that the popular skip-gram model (Mikolov et al., 2013a) can be adapted to generate ontologically grounded sense vectors. The classic skip-gram model uses a set of parameters $\theta = (U, V)$, with $U = \{u_i \mid \forall c_i \in W\}$ and $V = \{v_i \mid \forall w_i \in W\}$ being sets of vectors for context and target words respectively. The generative story of the skip-gram model involves generating the context word c_i conditioned on an observed word w_i . The conditional probability is defined to be $p(c_i \mid w_i; \theta) = \frac{\exp(u_i \cdot v_i)}{\sum_{c_{i'} \in W} \exp(u_{i'} \cdot v_i)}$.

We modify the generative story of the skip-gram model to account for latent sense variables by first selecting a latent word sense s_{ij} conditional on the observed word w_i , then generating the context word c_i from the sense distinguished word s_{ij} . This process is illustrated in Figure 3.2. The factorization $p(c_i \mid w_i; \theta) = \sum_{s_{ij}} p(c_i \mid s_{ij}; \theta) \times p(s_{ij} \mid w_i; \theta)$ follows from the chain rule since senses are word-specific. To parameterize this distribution, we define a new set of model parameters $\theta = (U, V, \Pi)$, where U remains identical to the original skip-gram, $V = \{v_{ij} \mid \forall s_{ij} \in W_s\}$ are a set of vectors for word senses (rather than word surface forms), and Π are the context-independent sense proportions $\pi_{ij} = p(s_{ij} \mid w_i)$. We use a Dirichlet prior over the multinomial distributions π_i for every w_i , with a shared concentration parameter λ .

Although this model can be learned using EM, there is nothing to identify it to the senses in the ontology Ω . We address this by formulating priors. We define the ontological prior on vectors as $p_{\Omega}(\theta) \propto \exp(-\gamma \sum_{ij-i'j'} \beta_r \|v_{ij} - v_{i'j'}\|^2)$, where γ controls the strength of the prior. We

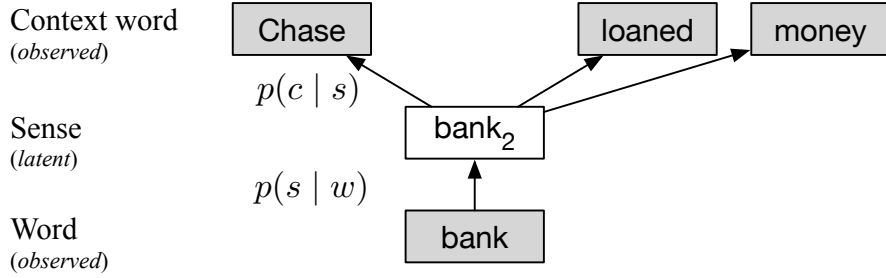


Figure 3.2: The generative process associated with the skip-gram model, modified to account for latent senses. Here, the context of the ambiguous word “bank” is generated from the selection of a specific latent sense.

note the similarity to the retrofitting objective in equation 3.1, except with $\alpha = 0$. This leads to the following realization of the objective in equation 3.3:

$$C(\theta) = \arg \max_{\theta} \sum_{(w_i, c_i) \in D} \log \left(\sum_{s_{ij}} p(c_i | s_{ij}; \theta) \times p(s_{ij} | w_i; \theta) \right) - \gamma \sum_{ij-i'j'} \beta_r \|v_{ij} - v_{i'j'}\|^2 \quad (3.4)$$

This objective can be optimized using EM, for the latent variables, and with lazy updates (Carpenter, 2008) every k words to account for the prior regularizer. However, since we are primarily interested in learning good vector representations, and we want to learn efficiently from large datasets, we make the following simplifications.

First, we perform “hard” EM, selecting the most likely sense at each position rather than using the full posterior over senses. Also, given that the structured regularizer $p_{\Omega}(\theta)$ is essentially the retrofitting objective in equation 3.1, we run retrofitting periodically every k words (with $\alpha = 0$ in equation 3.2) instead of lazy updates.²

The following decision rule is used in the “hard” E-step:

$$s_{ij} = \arg \max_{s_{ij}} p(c_i | s_{ij}; \theta^{(t)}) \pi_{ij}^{(t)} \quad (3.5)$$

In the M-step we use Variational Bayes to update Π with:

$$\pi_{ij}^{(t+1)} \propto \frac{\exp \left(\psi \left(\tilde{c}(w_i, s_{ij}) + \lambda \pi_{ij}^{(0)} \right) \right)}{\exp \left(\psi \left(\tilde{c}(w_i) + \lambda \right) \right)} \quad (3.6)$$

where $\tilde{c}(\cdot)$ is the online expected count and $\psi(\cdot)$ is the digamma function. This approach is motivated by Johnson (2007) who found that naive EM leads to poor results, while Variational Bayes is consistently better and promotes faster convergence of the likelihood function.

²We find this gives slightly better performance.

To update the parameters U and V , we use negative sampling (Mikolov et al., 2013a) which is an efficient approximation to the original skip-gram objective. Negative sampling attempts to distinguish between true word pairs in the data, relative to noise. Stochastic gradient descent on the following equation is used to update the model parameters U and V :

$$\begin{aligned} \mathcal{L} = & \log \sigma(u_i \cdot v_{ij}) + \sum_{\substack{j' \\ j' \neq j}} \log \sigma(-u_i \cdot v_{ij'}) \\ & + \sum_m \mathbb{E}_{c_{i'} \sim P_n(c)} [\log \sigma(-u_{i'} \cdot v_{ij})] \end{aligned} \quad (3.7)$$

Here $\sigma(\cdot)$ is the sigmoid function, $P_n(c)$ is a noise distribution computed over unigrams and m is the negative sampling parameter. This is almost exactly the same as negative sampling proposed for the original skip-gram model. The only change is that we additionally take a negative gradient step with respect to all the senses that were *not* selected in the hard E-step. We summarize the training procedure for the adapted skip-gram model in Algorithm 2.

Algorithm 2 Outputs a sense-specific VSM, given a corpus and an ontology

```

1: function SENSEEM( $D, \Omega$ )
2:    $\theta^{(0)} \leftarrow$  initialize
3:   for  $(w_i, c_i) \in D$  do
4:     if period  $i, k$  then
5:       RETROFIT( $\theta^{(t)}, \Omega$ )
6:     end if
7:     (Hard) E-step:
8:        $s_{ij} \leftarrow$  find argmax using equation 3.5
9:     M-step:
10:     $\Pi^{(t+1)} \leftarrow$  update using equation 3.6
11:     $U^{(t+1)}, V^{(t+1)} \leftarrow$  update using equation 3.7
12:   end for
13:   return  $\theta^{(t)}$ 
14: end function

```

3.3 Resources, Data and Training

We detail the training and setup for our experiments in this section.

We use WordNet (Miller, 1995) as the sense repository and ontology in all our experiments. WordNet is a large, hand-annotated ontology of English composed of 117,000 clusters of senses, or “synsets” that are related to one another through semantic relations such as hypernymy and hyponymy. Each synset additionally comprises a list of sense specific lemmas which we use to form the nodes in our graph. There are 206,949 such sense specific lemmas, which we connect

with synonym, hypernym and hyponym³ relations for a total of 488,432 edges.

To show the applicability of our techniques to different VSMs we experiment with two different kinds of base vectors.

Global Context Vectors (GC) (Huang et al., 2012): These word vectors were trained using a neural network which not only uses local context but also defines global features at the document level to further enhance the VSM. We distinguish three variants: the original single-sense vectors (SINGLE), a multi-prototype variant (MULTI), – both are available as pre-trained vectors for download⁴ – and a sense-based version obtained by running retrofitting on the original vectors (RETRO).

Skip-gram Vectors (SG) (Mikolov et al., 2013a): We use the word vector tool `Word2Vec`⁵ to train skip-gram vectors. We define 6 variants: a single-sense version (SINGLE), two multi-sense variants that were trained by first sense disambiguating the entire corpus using WSD tools, – one unsupervised (Pedersen and Kolhatkar, 2009) (WSD) and the other supervised (Zhong and Ng, 2010) (IMS) – a retrofitted version obtained from the single-sense vectors (RETRO), an EM implementation of the skip-gram model with the structured regularizer as described in section 3.2.2 (EM+RETRO), and the same EM technique but ignoring the ontology (EM). All models were trained on publicly available WMT-2011⁶ English monolingual data. This corpus of 355 million words, although adequate in size, is smaller than typically used billion word corpora. We use this corpus because the WSD baseline involves preprocessing the corpus with sense disambiguation, which is slow enough that running it on corpora orders of magnitude larger was infeasible.

Retrofitted variants of vectors (RETRO) are trained using the procedure described in algorithm 1. We set the convergence criteria to $\epsilon = 0.01$ with a maximum number of iterations of 10. The weights in the update equation 3.2 are set heuristically: the sense agnostic weight α is 1.0, and relations-specific weights β_r are 1.0 for synonyms and 0.5 for hypernyms and hyponyms. EM+RETRO vectors are the exception where we use a weight of $\alpha = 0.0$ instead, as required by the derivation in section 3.2.2.

For skip-gram vectors (SG) we use the following standard settings, and do not tune any of the values. We filter all words with frequency < 5 , and pre-normalize the corpus to replace all numeric tokens with a placeholder. We set the dimensionality of the vectors to 80, and the window size to 10 (5 context words to either side of a target). The learning rate is set to an initial value of 0.025 and diminished linearly throughout training. The negative sampling parameter is set to 5.

Additionally for the EM variants (section 3.2.2) we set the Dirichlet concentration parameter λ to 1000. We use 5 abstract senses for the EM vectors, and initialize the priors uniformly. For EM+RETRO, WordNet dictates the number of senses; also when available WordNet lemma counts are used to initialize the priors. Finally, we set the retrofitting period k to 50 million words.

³We treat edges as undirected, so hypernymy and hyponymy are collapsed and unified in our representation schema.

⁴http://nlp.stanford.edu/~socherr/ACL2012_wordVectorsTextFile.zip

⁵<https://code.google.com/p/word2vec/>

⁶<http://www.statmt.org/wmt11/>

Model	Variant	Word Similarity (ρ)				Synonym Selection (%)		
		WS-353	RG-65	MC-30	MEN-3K	ESL-50	RD-300	TOEFL-80
GC	Single	0.623	0.629	0.657	0.314	47.73	45.07	60.87
	Multi	0.535	0.510	0.309	0.359	27.27	47.89	52.17
	Retro	0.543	0.661	0.714	0.528	63.64	66.20	71.01
SG	Single	0.639	0.546	0.627	0.646	52.08	55.66	66.67
	EM	0.194	0.278	0.167	0.228	27.08	33.96	40.00
	WSD	0.481	0.298	0.396	0.175	16.67	49.06	42.67
	IMS	0.549	0.579	0.606	0.591	41.67	53.77	66.67
	Retro	0.552	0.673	0.705	0.560	56.25	65.09	73.33
	Retro+EM	0.321	0.734	0.758	0.428	62.22	66.67	68.63

Table 3.1: Similarity scoring and synonym selection in English across several datasets involving different VSMs. Higher scores are better; best scores within each category are in bold. In most cases our models consistently and significantly outperform the other VSMs.

3.4 Evaluation

In this section we detail experimental results on 4 lexical semantics tasks across 9 different datasets and 2 languages.

3.4.1 Experimental Results

We evaluate our models on 3 kinds of lexical semantic tasks: similarity scoring, synonym selection, and similarity scoring in context.

Similarity Scoring: This task involves using a semantic model to assign a score to pairs of words. We use the following 4 standard datasets in this evaluation: WS-353 (Finkelstein et al., 2002), RG-65 (Rubenstein and Goodenough, 1965), MC-30 (Miller and Charles, 1991) and MEN-3k (Bruni et al., 2014). Each dataset consists of pairs of words along with an averaged similarity score obtained from several human annotators. For example an item in the WS-353 dataset is “book, paper \rightarrow 7.46”. We use standard cosine similarity to assign a score to word pairs in single-sense VSMs, and the following average similarity score to multi-sense variants, as proposed by Reisinger and Mooney (2010):

$$avgSim(w_i, w_{i'}) = \frac{1}{k_i k_{i'}} \sum_{j, j'} \cos(v_{ij}, v_{i'j'}) \quad (3.8)$$

The output of systems is evaluated against the gold standard using Spearman’s rank correlation coefficient.

Synonym Selection: In this task, VSMs are used to select the semantically closest word to a target from a list of candidates. We use the following 3 standard datasets in this evaluation: ESL-50 (Turney, 2002), RD-300 (Jarmasz and Szpakowicz, 2004) and TOEFL-80 (Landauer and Dumais, 1997). These datasets consist of a list of target words that appear with several candidate lexical items. An example from the TOEFL dataset is “rug \rightarrow sofa, ottoman, carpet, hallway”,

with “carpet” being the most synonym-like candidate to the target. We begin by scoring all pairs composed of the target and one of the candidates. We use cosine similarity for single-sense VSMs, and max similarity for multi-sense models⁷:

$$\text{maxSim}(w_i, w_{i'}) = \max_{j, j'} \cos(v_{ij}, v_{i'j'}) \quad (3.9)$$

These scores are then sorted in descending order, with the top-ranking score yielding the semantically closest candidate to the target. Systems are evaluated on the basis of their accuracy at discriminating the top-ranked candidate.

The results for similarity scoring and synonym selection are presented in table 3.1. On both tasks and on all datasets, with the partial exception of WS-353 and MEN-3k, our vectors (RETRO & EM+RETRO) consistently yield better results than other VSMs. Notably, both our techniques perform better than preprocessing a corpus with WSD information in unsupervised or supervised fashion (SG-WSD & SG-IMS). Simple EM without an ontological prior to ground the vectors (SG-EM) also performs poorly.

We investigated the observed drop in performance on WS-353 and found that this dataset consists of two parts: a set of *similar* word pairs (e.g. “tiger” and “cat”) and another set of *related* word pairs (e.g. “weather” and “forecast”). The synonym, hypernym and hyponym relations we use tend to encourage *similarity* to the detriment of *relatedness*.

We ran an auxiliary experiment to show this. SG-EM+RETRO training also learns vectors for context words – which can be thought of as a proxy for relatedness. Using this VSM we scored a word pair by the average similarity of all the sense vectors of one word to the context vector of the other word, averaged over both words. For example, given the pair “bank” and “money” (which are related, not similar), we would take the different sense vectors of “bank” from the sense vector space and compare it to the single vector of “money” from the context space.

Using this scoring strategy the correlation ρ jumped from 0.321 to 0.493. While still not as good as some of the other VSMs, it should be noted that this scoring function, while improving *related* words, negatively influences the *similar* word pairs in the dataset.

The MEN-3k dataset is crowd-sourced and contains much diversity, with word pairs evidencing similarity as well as relatedness. However, we aren’t sure why the performance for GC-RETRO improves greatly over GC-SINGLE for this dataset, while that of SG-RETRO and SG-RETRO+EM drops in relation to SG-SINGLE.

Similarity Scoring in Context: As outlined by Reisinger and Mooney (2010), multi-sense VSMs can be used to consider context when computing similarity between words. We use the SCWS dataset (Huang et al., 2012) in these experiments. This dataset is similar to the similarity scoring datasets, except that they additionally are presented in context. For example an item involving the words “bank” and “money”, gives the words in their respective contexts, “along the east **bank** of the Des Moines River” and “the basis of all **money** laundering” with a low averaged similarity score of 2.5 (on a scale of 1.0 to 10.0). Following Reisinger and Mooney (2010) we use the following function to assign a score to word pairs in their respective contexts, given a multi-sense VSM:

⁷Here we are specifically looking for synonyms, so the max makes more sense than taking an average.

Model	Variant	SCWS (ρ)
GC	Multi	0.657
	Retro	0.420
SG	EM	0.613
	WSD	0.343
	IMS	0.528
	Retro	0.417
	Retro+EM	0.587

Table 3.2: Contextual word similarity in English. Higher scores are better.

$$\begin{aligned}
 & avgSimC(w_i, c_i, w_{i'}, c_{i'}) = \\
 & \sum_{j,j'} p(s_{ij}|c_i, w_i) p(s_{i'j'}|c_{i'}, w_{i'}) \cos(v_{ij}, v_{i'j'})
 \end{aligned} \tag{3.10}$$

As with similarity scoring, the output of systems is evaluated against gold standard using Spearman’s rank correlation coefficient.

The results are presented in table 3.2. Pre-processing a corpus with WSD information in an unsupervised fashion (SG-WSD) yields poor results. In comparison, the retrofitted vectors (SG-RETRO & GC-RETRO) already perform better, even though they do not have access to context vectors, and thus do not take contextual information into account. Supervised sense vectors (SG-IMS) are also competent, scoring better than both retrofitting techniques. Our EM vectors (SG-EM & SG-EM+RETRO) yield even better results and are able to capitalize on contextual information, however they still fall short of the pre-trained GC-MULTI vectors. We were surprised that SG-EM+RETRO actually performed worse than SG-EM, given how poorly SG-EM performed in the other evaluations. However, an analysis again revealed that this was due to the kind of similarity encouraged by WordNet rather than an inability of the model to learn useful vectors. The SCWS dataset, in addition to containing related words – which we showed, hurt our performance on WS-353 – also contains word pairs with different POS tags. WordNet synonymy, hypernymy and hyponymy relations are exclusively defined between lemmas of the same POS tag, which adversely affects performance further.

3.4.2 Generalization to Another Language

Our method only depends on a sense inventory to improve and disambiguate VSMs. We thus experiment with extending the technique to another language. In particular, we run the retrofitting procedure in algorithm 1 on VSMs trained on French monolingual data from the WMT-2011 corpus. The ontological base for our algorithm is obtained from a semi-automatically created version of WordNet in French (Pradet et al., 2013).

We compare the resulting VSMs against the “original” vectors on a French translation of the RG-65 word similarity dataset (Joubarne and Inkpen, 2011). GC vectors cannot be retrained and are thus not used in this evaluation. We do not include WSD or IMS results for this task since we

Word Similarity in French

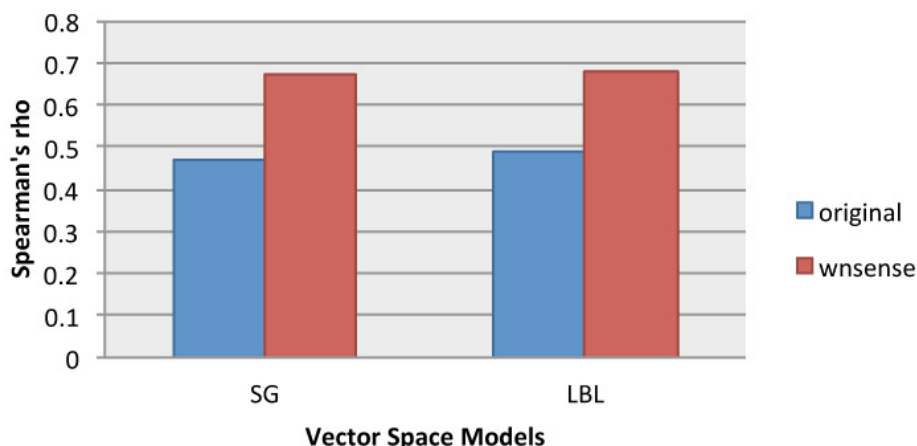


Figure 3.3: Performance of sense specific vectors (*wnsense*) compared with single sense vectors (*original*) on similarity scoring in French. Higher bars are better. *wnsense* outperforms *original* across 2 different VSMs on this task.

were unable to find a reasonable WSD system in French to sense tag the training corpus. As with previous experiments involving similarity scoring we use the *avgSim* function to assign similarity scores to word pairs, and system outputs are evaluated against gold standard on Spearman's rank correlation coefficient.

Results of the experiment are presented in figure 3.3, and again show that our sense disambiguated VSMs outperform their single-sense counterparts on the evaluation task. We conclude that the technique we propose not only generalizes to any VSM but to any language, as long as a reasonable sense inventory for that language exists.

3.4.3 Antonym Selection

The general formulation of our model allows us to incorporate knowledge from more than just the similarity neighborhood (i.e. the synonyms, the hypernyms and the hyponyms) of word senses in an ontology. The power of the model lies in leveraging the relational structure of the ontology, and the flexibility to use different kinds of relations in different ways.

For example, it may be useful to also know the antonyms of a word sense to tease its vector apart from the empirical surface form embedding more effectively.

Corpus-based methods struggle to learn good representations for antonyms, because antonym pairs often occur in very similar contexts. For example the words “hot” and “cold” are equally appropriate to describe a number of things – food, the weather, or someone's mood. Thus, based on the distributional hypothesis, their associated vectors tend to be very similar.

However, antonym pairs should ideally have low similarity (i.e. large distance) between them. In our model, it is thus appropriate to set a negative relation weight to these terms in the model parameters in equation 3.2 to force the respective vectors further apart.

Mathematically, in certain cases this causes the objective to diverge, and thus produces un-

Closest To Opposite Verb Selection

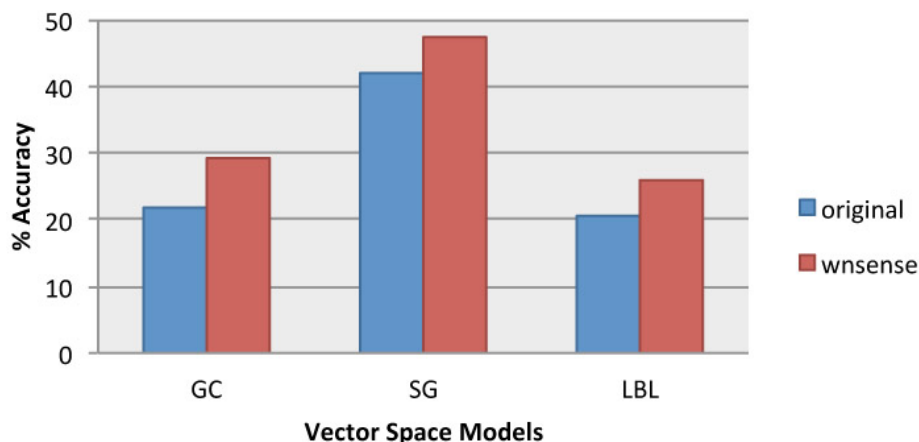


Figure 3.4: Performance of sense specific vectors (*wnsense*) compared with single sense vectors (*original*) on closest to opposite selection of target verbs. Higher bars are better. *wnsense* outperforms *original* across 3 different VSMs on this task.

predictable behavior. This happens, in particular when some variables in the model form an “antonym clique”. That is, when the nodes associated with each of the concepts in this “clique” are all connected to one another (and *only* to one another) via an antonym relation.

Thus we test our model on the subset of WordNet consisting only of verb lemmas, which contain very few antonym cliques⁸. We run exactly the same procedure of algorithm 1 to create sense disambiguated vectors as before, except that we additionally expand word sense neighborhoods with antonyms, and set the antonym specific relation weight γ_r to be -1.0 in equation 3.2 for the model parameters. In the few cases that do evidence divergent behavior, we heuristically overcome the problem by ceasing to update the problematic vectors if their norm increases beyond some threshold (in our experiments we set this to be the norm of the longest vector before any update iterations).

We evaluate the resulting VSMs on the task of closest-to-opposite verb selection. This is very similar in format to the synonym selection task, except that the candidate with the most antonym-like meaning with respect to a target needs to be selected. The dataset we use in our experiments is from Mohammad et al. (2008). The data contains 226 closest-to-opposite verb selection questions such as the following: “inflate \rightarrow overstating, understating, deflate, misrepresenting, exaggerating” where “deflate” is the correct answer.

We use the *minSim* function to score similarity between words since we are interested in discovering the least synonymous word sense pairs. These scores are then sorted in descending order, with the lowest-ranking score yielding the closest-to-opposite candidate to the target. Systems are evaluated on the basis of their accuracy at discriminating the correct candidate.

Results on this task are shown in figure 3.4. The baseline vectors show typical behavior of models that are based on the distributional hypothesis. Antonyms are treated much like syn-

⁸For comparison, 45% of all adjective lemmas appear in such cliques, while only 0.1% of verbs do.

Method	CPU Time
Retro	~20 secs
Retro+EM	~4 hours
IMS	~3 days
WSD	~1 year

Table 3.3: Training time associated with different methods of generating sense-specific VSMs.

onyms, and thus the baselines perform poorly on this task. In contrast, our sense specific vectors are forced to be further apart if they are antonyms and thus perform better on the evaluation task consistently across all three VSMs.

3.5 Discussion

While both our approaches are capable of integrating ontological information into VSMs, an important question is which one should be preferred? From an empirical point of view, the EM+RETRO framework yields better performance than RETRO across most of our semantic evaluations. Additionally EM+RETRO is more powerful, allowing to adapt more expressive models that can jointly learn other useful parameters – such as context vectors in the case of skip-gram. However, RETRO is far more generalizable, allowing it to be used for *any* VSM, not just predictive MLE models, and is also empirically competitive. Another consideration is computational efficiency, which is summarized in table 3.3.

Not only is RETRO much faster, but it scales linearly with respect to the vocabulary size, unlike EM+RETRO, WSD, and IMS which are dependent on the input training corpus. Nevertheless, both our techniques are empirically superior as well as computationally more efficient than both unsupervised and supervised word-sense disambiguation paradigms.

Both our approaches are sensitive to the structure of the ontology. Therefore, an important consideration is the relations we use and the weights we associate with them. In our experiments we selected the simplest set of relations and assigned weights heuristically, showing that our methods can effectively integrate ontological information into VSMs. A more exhaustive selection procedure with weight tuning on held-out data would almost certainly lead to better performance on our evaluation suite.

3.5.1 Qualitative Analysis

We qualitatively attempt to address the question of whether the vectors are truly sense specific. In table 3.4 we present the three most similar words of an ambiguous lexical item in a standard VSM (SG-SINGLE) in comparison with the three most similar words of different lemma senses of the same lexical item in grounded sense VSMs (SG-RETRO & SG-EM+RETRO).

The sense-agnostic VSMs tend to capture only the most frequent sense of a lexical item. On the other hand, the disambiguated vectors capture sense specificity of even less frequent senses successfully. This is probably due to the nature of WordNet where the nearest neighbors of the

Word or Sense	Top 3 Most Similar
hanging	hung, dangled, hangs
hanging (suspending)	shoring, support, suspension
hanging (decoration)	tapestry, braid, smock
climber	climbers, skier, Loretan
climber (sportsman)	lifter, swinger, sharpshooter
climber (vine)	woodbine, brier, kiwi

Table 3.4: The top 3 most similar words for two polysemous types. Single sense VSMs capture the most frequent sense. Our techniques effectively separates out the different senses of words, and are grounded in WordNet.

words in question are in fact these rare words. A careful tuning of weights will likely optimize the trade-off between ontologically rare neighbors and distributionally common words.

In our analyses, we noticed that lemma senses that had many neighbors (i.e. synonyms, hypernyms and hyponyms), tended to have more clearly sense specific vectors. This is expected, since it is these neighborhoods that disambiguate and help to distinguish the vectors from their single sense embeddings.

3.6 Conclusion

We have presented two general and flexible approaches to producing sense-specific VSMs grounded in an ontology. The first technique is applicable to any VSM as an efficient post-processing step, which we call retrofitting. The second provides a framework to integrate ontological information by adapting existing MLE-based predictive models. We evaluated our models on a battery of tasks and the results show that our proposed methods are effectively able to capture the different senses in an ontology. In most cases this results in significant improvements over baselines. We thus re-affirm the central claim of our thesis: namely that integrating relational structure into learning semantic models results in representations that are more expressive and empirically superior than their relation agnostic counterparts.

We also discussed the trade-offs between the two techniques from several different perspectives. In summary, while the adaptive MLE technique seems to be empirically superior, retrofitting is much more computationally friendly. Crucially, however, both models significantly surpass word-sense disambiguating a training corpus from both computational and empirical perspectives. Finally, we also presented a qualitative analysis investigating the nature of the sense-specific vectors, and showed that they do indeed capture the semantics of different senses.

Chapter 4

Tables for Question Answering

Leveraging Structural Relations in Tables for Question Answering

Question answering (QA) has emerged as a practical research problem for pushing the boundaries of artificial intelligence (AI). Dedicated projects and open challenges to the research community include examples such as Facebook AI Research’s challenge problems for AI-complete QA (Weston et al., 2015) and the Allen Institute for AI’s (AI2) Aristo project (Clark, 2015) along with its recently completed Kaggle competition¹. The reason for this emergence is the diversity of core language and reasoning problems that a complex, integrated task like QA exposes: information extraction (Srihari and Li, 1999), semantic modelling (Shen and Lapata, 2007; Narayanan and Harabagiu, 2004), logic and reasoning (Moldovan et al., 2003), and inference (Lin and Pantel, 2001).

Complex tasks such as QA require some form of knowledge base to store facts about the world and reason over them. Consider the example question: “In which country is Texas?”. A system truly capable of understanding and then answering this question must know – or infer – that Texas is the name of a place and that it is contained within a geo-political entity called the United States of America. But even the simplest system must be able to search for a relevant passage in a background corpus to answer this question.

Thus by knowledge base, we mean any form of knowledge: structured (e.g., tables, ontologies, rules) or unstructured (e.g., natural language text). For QA, knowledge has been harvested and used in a number of different modes and formalisms: large-scale extracted and curated knowledge bases (Fader et al., 2014), structured models such as Markov Logic Networks (Khot et al., 2015), and simple text corpora in information retrieval approaches (Tellex et al., 2003).

When considered in the context of this thesis, an important question concerns the semantic relations that structure the source of background knowledge for a problem like QA.

For example, the relations structuring a large-scale ontology such as Yago (Suchanek et al., 2007) or DBpedia (Lehmann et al., 2015) are very different from the relations in a triple store such as the PropStore (see Chapter 6.2). These in turn are very different from the set of relations in a collection of tables, which are for example, implicit membership relations such as “in_same_row” or “in_same_column”.

¹<https://www.kaggle.com/c/the-allen-ai-science-challenge>

The schema of the relations in the underlying knowledge base have direct implications on the QA system that is built on top of it. What relations are usable? Which ones capture important properties relating questions to answers?

There is a fundamental trade-off in the structure and regularity of the knowledge schema and its ability to be curated, modelled or reasoned with easily. The degree of structure can be thought of as falling on a spectrum.

At one end of this spectrum is, for example, a simple text corpus. It contains no structure, and is therefore hard to reason with in a principled manner. Nevertheless, vast amounts of unstructured text data are easily and abundantly available. At the other end of the spectrum is, for example, a Markov Logic Networks. This sort of knowledge formalism comes with a wealth of theoretical knowledge connected with its usage in principled inference. However, it is difficult to induce automatically from text or to build manually.

In this chapter we explore a knowledge formalism that falls somewhere in the middle of this spectrum of structured knowledge. Specifically we investigate tables as semi-structured knowledge for multiple-choice question (MCQ) answering.

In particular, we focus on tables that represent general knowledge facts, with cells that contain free-form text (Section 4.2.1 details the nature and semantics of these tables). The structural properties of tables, along with their free-form text content represents a semi-structured balanced compromise in the trade-off between degree of structure and ubiquity.

Crucially, in the context of this thesis, the role of semantic relations in these tables is three-fold. First, the way the tables are structured leads to a formalism that is easy enough to build by non-expert annotators, while being flexible enough to be used for the complex reasoning required in QA. Second, the relations governing tables define a greater connection with QA; specifically, they can be used as constraints to generate high quality QA data via crowd-sourcing for our target domain. Thirdly, the same structure in tables and its connection to the QA data we generate can be leveraged to build QA solvers that are capable of performing reasoning over tables.

The two main contributions of this chapter are centered around the role that relations play in the structure of tables. First, we crowd-source a collection of over 9000 MCQs with alignment annotations to table elements, using the structural properties of tables as guidelines in efficient data harvesting. Second, we develop feature-driven models that use these MCQs to perform QA, while fact-checking and reasoning over tables.

Additionally, in the latter contribution we explore two model paradigms. In the first, the structure and relations of tables are used to inspire manual feature-engineering. While in the second, those same relations are used in a neural network to automatically induce features for the task.

The use of tables in the context of QA is not new; others have explored this avenue of research as well. Question bank creation for tables has been investigated (Pasupat and Liang, 2015), but without structural guidelines or the alignment information that we propose. Similarly, tables have been used in QA reasoning (Yin et al., 2015b; Neelakantan et al., 2015; Sun et al., 2016) but have not explicitly attempted to encode all the semantics of table structure (see Section 4.2.1). To the best of our knowledge, no previous work uses tables for both creation and reasoning in a connected framework.

We evaluate our manually feature-engineered model on MCQ answering for three benchmark datasets. Our results consistently and significantly outperform a strong retrieval baseline as well

as a Markov Logic network model (Khot et al., 2015). We thus show the benefits of semi-structured data and models over unstructured or highly-structured counterparts. We also validate our curated MCQ dataset and its annotations as an effective tool for training QA models. Finally, we find that our model learns generalizations that permit inference when exact answers may not even be contained in the knowledge base.

Our evaluations of the neural network model show that the architecture we define is able to capture the relational and structural properties of tables. It shows promising results when we evaluate on a held-out portion of our training set. The model is however, much more sensitive to the content in training than the manually feature-engineered model. The representations over MCQ and table elements we learn do not port to the largest of our benchmark test sets, which has a very different profile to our crowd-sourced training set. Our analyses indicate the need for a larger and more diverse training set to accommodate the power of the neural network model.

But both sets of results re-affirm the central claims of this thesis. That structure and relations are key to producing meaning in language. Moreover, leveraging the correct set of relations to yield feature representations (in this chapter, both manually and automatically) lead to richer and more capable models.

4.1 Related Work

Our work with tables, semi-structured knowledge bases and QA relates to several parallel lines of research. In terms of dataset creation via crowd-sourcing, Aydin et al. (2014) harvest MCQs via a gamified app, although their work does not involve tables. Pasupat and Liang (2015) use tables from Wikipedia to construct a set of QA pairs. However their annotation setup does not impose structural constraints from tables, and does not collect fine-grained alignment to table elements.

The construction of knowledge bases in structured format relates more broadly to work on open information extraction (Etzioni et al., 2008; Wu and Weld, 2010; Fader et al., 2011) and knowledge base population (Ji and Grishman, 2011). While the set of tables we currently work with are hand-built, there are efforts to at least partially automate this process (Dalvi et al., 2016).

On the inference side Pasupat and Liang (2015) also reason over tables to answer questions. Unlike our approach, they do not require alignments to table cells. However, they assume knowledge of the table that contains the answer, a priori – which we do not. Yin et al. (2015b) and Neelakantan et al. (2015) also use tables in the context of QA, but deal with synthetically generated query data. Sun et al. (2016) perform cell search over web tables via relational chains, but are more generally interested in web queries. Clark et al. (2016) combine different levels of knowledge for QA, including an integer-linear program for searching over table cells. None of these other efforts leverage tables for generation of data.

Our research more generally pertains to natural language interfaces for databases. Cafarella et al. (2008) parse the web to create a large-scale collection of HTML tables. They compute statistics about co-occurring attributes, inferring schemas and reasoning about likely table relational structure. Venetis et al. (2011) are more specifically concerned with recovering table schema, and their insight into column semantics is related to our own structural decomposition and analysis of tables. Other research connected with relational schema induction for database

Phase Change		Initial State		Final State		Form of Energy Transfer
Melting	causes a	solid	to change into a	liquid	by	adding heat
Vaporization	causes a	liquid	to change into a	gas	by	adding heat
Condensation	causes a	gas	to change into a	liquid	by	removing heat
Sublimation	causes a	solid	to change into a	gas	by	adding heat

Table 4.1: Example part of table concerning phase state changes.

tables is by Syed et al. (2010) and Limaye et al. (2010).

A closely related line of study is connected with executing queries over tables represented as relational databases. This applies more directly to the problem of QA with tables. Cafarella et al. (2008), besides building tables and inferring schemas, also perform queries against these tables. But unlike the natural language questions normally associated with QA, these queries are more IR style lists of keywords. The work of Pimplikar and Sarawagi (2012) is another effort that relies on column semantics to reason better about search requests. Yin et al. (2015a) consider databases where information is stored in n-tuples, which are essentially tables.

4.2 Tables as Semi-structured Knowledge Representation

Tables can be found on the web containing a wide range of heterogenous data. To focus and facilitate our work on QA we select a collection of tables that were specifically designed for the task. Specifically we use AI2’s Aristo Tablestore². However, it should be noted that the contributions of this work are not tied to specific tables, as we provide a general methodology that could equally be applied to a different set of tables.

In this section we present the set of tables of curated natural language facts. In the context of this thesis, we discuss their semi-structured form and the semantic relations that make up their structural components. We also discuss the utility of these relations in knowledge representation for the domain. Finally we give details of the data we deal with.

4.2.1 Table Semantics and Relations

Part of a table from the Aristo Tablestore is given as an example in Table 4.1. The format is semi-structured: the rows of the table (with the exception of the header) are a list of sentences, but with well-defined recurring filler patterns. Together with the header, these patterns divide the rows into meaningful columns. This semi-structured data format is flexible. Since facts are presented as sentences, the tables can act as a text corpus for information retrieval. At the same time the structure can be used – as we do – to focus on specific nuggets of information. The flexibility of these tables allows us to compare our table-based system to an information retrieval baseline.

Such tables have some interesting structural semantics, which we will leverage throughout the paper. A row in a table corresponds to a *fact*³. For example, the sentence “Melting causes a

²<http://allenai.org/content/data/AristoTablestore-Nov2015Snapshot.zip>

³Also predicates, or more generally frames with typed arguments.

solid to change into a liquid by adding heat” is ostensibly a fact.

The cells in a row correspond to concepts, entities, or processes that participate in this fact. In the example sentence above, “melting” “solid”, “liquid” etc. are all related processes or concepts that participate in the corresponding fact. They are bound to one another by the prototype relation defined by *types and effects of energy addition*.

A content column – which we differentiate from filler columns that only contain a recurring pattern, and no information in their header cells – corresponds to a group of concepts, entities, or processes that are the same *type*. For example, the column containing the processes “melting”, “vaporization”, “condensation” etc. is a content column, while the column containing the identical pattern “to change into” in every row is a filler column.

The header cell of the column is an abstract description of the *type*. We may view the head as a hypernym and the cells in the column below as co-hyponyms of the head. The header row defines a generalization of which the rows in the table are specific instances. For example, “phase change” is an abstract descriptor, or hypernym of the processes “melting”, “vaporization”, “condensation” etc.

These tables also encode implicit analogical relations. Let row_1 and row_2 be any two rows in a table. Let col_1 and col_2 be any two *content* columns in a table. The four cells where the rows and columns intersect form an analogy: $cell_{1,1}$ is to $cell_{1,2}$ as $cell_{2,1}$ is to $cell_{2,2}$. That is, the relation between $cell_{1,1}$ and $cell_{1,2}$ is highly similar to the relation between $cell_{2,1}$ and $cell_{2,2}$. As an example, one might be able to derive the following analogy from the intersection of rows and columns in the tables: “melting is to solid” as “freezing is to liquid”. The table is thus essentially a compact representation of a large number of scientific analogies.

The semi-structured nature of this data is flexible. Since facts are presented as sentences, the tables can simply act as a text corpus for information retrieval. Depending on the end application, more complex models can rely on the inherent structure in tables. They can be used as is for information extraction tasks that seek to zero in on specific nuggets of information, or they can be converted to logical expressions using rules defined over neighboring cells. Regarding construction, the recurring filler patterns can be used as templates to extend the tables semi-automatically by searching over large corpora for similar facts (Ravichandran and Hovy, 2002; Dalvi et al., 2016).

Finally, this structure is directly relevant to multiple-choice QA. Facts (rows) form the basis for creating or answering questions, while instances of a type (columns) act as the choices of an MCQ. We use these observations both for crowd-sourcing MCQ creation as well as for designing models to answer MCQs with tables.

4.2.2 Table Data

The Aristo Tablestore consists of 65 hand-crafted tables organized by topic. Some of the topics are bounded, containing only a fixed number of facts, such as the possible phase changes of matter (see Table 4.1). Other topics are unbounded, containing a very large or even infinite number of facts, such as the kind of energy used in performing an action (the corresponding tables can only contain a sample subset of these facts). A total of 3851 facts (one fact per row) are present in the manually constructed tables. An individual table has between 2 and 5 *content* columns.

The target domain for these tables is two 4th grade science exam datasets. The majority of the tables were constructed to contain topics and facts from the publicly available Regents dataset⁴. The rest were targeted at an unreleased dataset called Monarch. In both cases only the training partition of each dataset was used to formulate and hand-craft tables. However, for unbounded topics, additional facts were added to each table, using science education text books and web searches.

4.3 Answering Multiple-choice Questions using Tables

In this section we detail how tables can be used to answer MCQs. Importantly, the method we describe relies on fine grained alignments between MCQs and the relational components of tables.

Consider the table given in Table 4.1, and the MCQ “What is the process by which water is changed from a liquid to a gas?” with choices “melting, sublimation, vaporization, condensation”. The problem of finding the correct answer amounts to finding a cell (or cells) in the table that is most relevant to a candidate Question-Answer pair. In other words, a relevant cell should confirm the assertion made by a particular Question-Answer pair.

It turns out that relevant cells can be defined (as shown in the example below) as a function of row and column intersections. Fortunately, there are clues in the table and the MCQ that help in locating such intersections and relevant cells.

If one can parse the question correctly one might learn that the question is seeking a “process”. Parallely one might learn that the choices “melting, sublimation, vaporization, condensation” are all phase changes, and therefore processes. Then, by the hypernym-hyponym relationship between a header-cell and its corresponding column, one might be able to say – with some degree of certainty – that the relevant cell is contained in the column headed by the cell “Phase Change”.

Solving the MCQ also requires locating a correct row. This can be done by matching the words “liquid” and “gas” to entries of a single row, and hypothetically using the “from” and “to” indicators to further distinguish between initial and final states of a phase change (thus potentially avoiding ambiguity between evaporation and condensation).

If we follow this line of reasoning, we find that the intersection of the correct row and column is the cell “vaporization”, which correctly answers the MCQ.

Figure 4.1 illustrates this with color for row and column alignments to the example MCQ presented above. The intersection of the blue row and the yellow column is the red cell, which contains the correct answer.

Of course, the ability to find these fine grained alignments is hypothetical. Lexical variation, word ordering, syntactic diversity all contribute to making the discovery of these alignments a non-trivial problem. We require training data in order to train a model to align table elements to parts of MCQs properly. And in the next section we will see exactly how these alignments can be obtained cheaply by non-expert annotators through a Mechanical Turk task that leverages the structural and relational properties of tables.

⁴<http://allenai.org/content/data/Regents.zip>

PHASE CHANGE		INITIAL PHASE		FINAL PHASE		HEAT TRANSFER
Melting	causes a	solid	to change into a	liquid	by	adding heat
Vaporization	causes a	liquid	to change into a	gas	by	adding heat
Sublimation	causes a	solid	to change into a	gas	by	adding heat
Freezing	causes a	liquid	to change into a	solid	by	removing heat
Deposition	causes a	gas	to change into a	solid	by	removing heat
Condensation	causes a	gas	to change into a	liquid	by	removing heat

Figure 4.1: Example table from MTurk annotation task illustrating constraints. We ask Turkers to construct questions from blue cells, such that the red cell is the correct answer, and distracters must be selected from yellow cells.

4.4 Crowd-sourcing Multiple-choice Questions and Table Alignments

Since this thesis is focussed on feature representation learning, we are interested in developing models that leverage features that rely on the structural and relational properties of tables, MCQs and the alignments between them.

In this section we outline the MCQs generated from tables. We first describe the annotation task and argue for constraints imposed by table structure and relations. We then describe the data generated from this annotation task.

4.4.1 MCQ Annotation Task

We use Amazon’s Mechanical Turk (MTurk) service to generate MCQs by imposing constraints derived from the structure of the tables. These constraints help annotators create questions with scaffolding information, and lead to consistent quality in the generated output. An additional benefit of this format is the alignment information, linking cells in the tables to the MCQs generated by the Turkers. The alignment information is generated as a by-product of making the MCQs and is crucial in developing our models that answer MCQs (see Section 4.3).

We present Turkers with a table such as the one in Figure 4.1. Given this table, we choose a target cell to be the correct answer for a new MCQ; for example, the red cell in Figure 4.1. First, Turkers create a question by using information from the rest of the row containing the target (i.e., the blue cells in Figure 4.1), such that the target is its correct answer. Then they select the cells in the row that they used to construct the question. Following this, they construct four succinct choices for the question, one of which is the correct answer and the other three are distractors. Distractors are formed from other cells in the column containing the target (i.e. yellow cells in Figure 4.1). If there are insufficient unique cells in the column Turkers create their own. Annotators can rephrase and shuffle the contents of cells as required to create the question, answer or any of the distractors.

In addition to an MCQ, we obtain alignment information with no extra effort from annotators.

Task	Avg. Time (s)	\$/hour	% Reject
Rewrite	345	2.61	48
Paraphrase	662	1.36	49
Add choice	291	2.47	24
Write new	187	5.78	38
TabMCQ	72	5.00	2

Table 4.2: Comparison of different ways of generating MCQs with MTurk.

What is the orbital event with the longest day and the shortest night?
A) Summer solstice
B) Winter solstice
C) Spring equinox
D) Fall equinox
Steel is a/an _____ of electricity
A) Separator
B) Isolator
C) Insulator
D) Conductor

Table 4.3: Examples of MCQs generated by MTurk. Correct answer choices are in bold.

We know which table, row, and column contains the answer, and thus we know which header cells might be relevant to the question. We also know the cells of a row that were used to construct a question. This is precisely the alignment information that we need to be able to train a model to replicate these alignments and thus answer MCQs by finding intersections between maximally aligned rows and columns.

It is worth noting, in the context of this thesis, that both the constraints of the annotation task and the alignment information we obtain as a by-product are centered around the relational properties of tables. The row and column structure play an important role in framing the annotation task for Turkers, while alignment information links into these same structural components.

4.4.1.1 The TabMCQ Dataset

We created a HIT (the MTurk acronym for Human Intelligence Task) for every *content* cell (i.e. non-filler cell; see Section 4.2) from each one of the 65 manually constructed tables of the Aristo Tablestore. We paid annotators 10 cents per MCQ, and asked for 1 annotation per HIT for most tables. For an initial set of four tables which we used in a pilot study, we asked for three annotations per HIT⁵. We required Turkers to have a HIT approval rating of 95% or higher, with a minimum of at least 500 HITs approved. We restricted the demographics of our workers to the US.

Table 4.2 compares our method with other studies conducted at AI2 to generate MCQs. These

⁵The goal was to obtain diversity in the MCQs created for a target cell. The results were not sufficiently conclusive to warrant a threefold increase in the cost of creation.

methods attempt to generate new MCQs from existing ones, or write them from scratch, but do not involve tables in any way. Our annotation procedure leads to faster data creation, with consistent output quality that resulted in the lowest percentage of rejected HITs. Manual inspection of the generated output also revealed that questions are of consistently good quality. They are good enough for training machine learning models and many are good enough as evaluation data for QA. A sample of generated MCQs is presented in Table 4.3.

We implemented some simple checks to evaluate the data before approving HITs. These included things like checking whether an MCQ has at least three choices and whether choices are repeated. We had to further prune our data to discard some MCQs due to corrupted data or badly constructed MCQs. A total of 159 MCQs were lost through the cleanup. In the end our complete data consists of 9092 MCQs, which is – to the best of our knowledge – orders of magnitude larger than any existing collection of science exam style MCQs available for research. These MCQs also come with alignment information to tables, rows, columns and cells. The dataset, bundled together with the Aristo Tablestore, can be freely downloaded at http://ai2-website.s3.amazonaws.com/data/TabMCQ_v_1.0.zip.

4.5 Feature Rich Table Embedding Solver

In this section we detail the first of our two system that leverages MCQs, tables and alignment information between them. Specifically we develop a feature rich solver of multiple-choice questions, where the features are manually engineered. Crucially, these features rely on the relational information in tables for inspiration.

We begin by outlining the model and the objective against which it is trained. We then describe our feature set, and show evaluation results in a number of different settings on three different benchmark datasets. Our results consistently and significantly outperform two baselines: a strong retrieval model and a highly structured Markov Logic Network model (Khot et al., 2015).

We demonstrate the advantages of semi-structured data and models over unstructured or highly-structured counterparts. We also find that our manually feature engineered model learns generalizations that permit inference when exact answers may not even be contained in the knowledge base, – thus effectively learning a model that abstracts over the content *and* structure of tables.

4.5.1 Model and Training Objective

Recall from Section 4.3 that a table may be used to answer a question by finding and focussing on a small set of relevant cells. Moreover, these cells are found by creating an intersection between rows and columns of interest, by a combination of leveraging the structure of a table as well as its contents.

By applying the reasoning used to create MCQs (see Section 4.4) in the inverse direction, finding these relevant cells becomes the task of finding an intersection between rows and columns of interest.

Consider the MCQ “What is the process by which water is changed from a liquid to a gas?” with choices “melting, sublimation, vaporization, condensation”, and the table given in Fig-

ure 4.1. Assuming we have some way of aligning a question to a row (blue cells) and choices to a column (yellow cells), then the relevant cell is at the intersection of the two (the red cell). This alignment is precisely what we get as a by-product of the annotation task we setup in Section 4.2.1 to harvest MCQs.

We can thus featurize connections between MCQs and elements of tables and use the alignment data to train a model over the features. This is exactly the intuition that our Feature Rich Table Embedding Solver (FRETs) seeks to model.

Let $\mathcal{Q} = \{q_1, \dots, q_N\}$ denote a set of MCQs, and $\mathcal{A}_n = \{a_n^1, \dots, a_n^k\}$ be the set of candidate answer choices for a given question q_n . Let the set of tables be defined as $\mathcal{T} = \{T_1, \dots, T_M\}$. Given a table T_m , let t_m^{ij} be the cell in that table corresponding to the i th row and j th column.

We define a log-linear model that scores every cell t_m^{ij} of every table in our collection according to a set of discrete weighted features, for a given QA pair. We have the following:

$$\log p(t_m^{ij} | q_n, a_n^k; \mathcal{A}_n, \mathcal{T}) = \sum_d \lambda_d f_d(q_n, a_n^k, t_m^{ij}; \mathcal{A}_n, \mathcal{T}) - \log Z \quad (4.1)$$

Here λ_d are weights and $f_d(q_n, a_n^k, t_m^{ij}; \mathcal{A}_n, \mathcal{T})$ are features. These features should ideally leverage both structure and content of tables to assign high scores to relevant cells, while assigning low scores to irrelevant cells. Z is the partition function, defined as follows:

$$Z = \sum_{m,i,j} \exp \left(\sum_d \lambda_d f_d(q_n, a_n^k, t_m^{ij}; \mathcal{A}_n, \mathcal{T}) \right) \quad (4.2)$$

Z normalizes the scores associated with every cell over all the cells in all the tables to yield a probability distribution. During inference the partition term $\log Z$ can be ignored, making scoring cells of every table for a given QA pair efficient.

These scores translate to a solution for an MCQ. Every QA pair produces a hypothetical fact, and as noted in Section 4.2.1, the row of a table is in essence a fact. Relevant cells (if they exist) should confirm the hypothetical fact asserted by a given QA pair. During inference, we assign the score of the highest scoring row (or the most likely fact) to a hypothetical QA pair. Then the correct solution to the MCQ is simply the answer choice associated with the QA pair that was assigned the highest score. Mathematically, this is expressed as follows:

$$a_n^* = \arg \max_{a_n^k} \max_{m,i} \sum_j \sum_d \lambda_d f_d(q_n, a_n^k, t_m^{ij}; \mathcal{A}_n, \mathcal{T}) \quad (4.3)$$

4.5.1.1 Training

Since FRETs is a log-linear model, training involves optimizing a set of weights λ_d . As training data, we use alignment information between MCQs and table elements (see Section 4.2.2). The predictor value that we try to maximize with our model is an alignment score that is closest to the true alignments in the training data. True alignments to table cells for a given QA pair are essentially indicator values but we convert them to numerical scores as follows⁶.

⁶On training data, we experimented with a few different scoring heuristics and found that these ones worked well.

- If the answer in a QA pair is the right answer, we assign:
 - a score of 1.0 to cells whose row and column values are both relevant (i.e. contains exactly the information needed to answer the question).
 - a score of 0.5 to cells whose row but not column values are relevant (i.e. is related to the fact asserted by the QA pair, but not exactly the relevant information).
 - a score of 0.0 otherwise (i.e. is not related to the QA pair)
- If the answer in a QA pair is not the right answer, we assign:
 - a score of 0.1 to random cells from irrelevant tables, with a probability of 1%.
 - a score of 0.0 otherwise.

The intuition behind this scoring scheme is to guide the model to pick relevant cells for correct answers, while encouraging it to pick faulty evidence with low scores for incorrect answers.

Given these scores assigned to all cells of all tables for all QA pairs in the training set, suitably normalized to a probability distribution over tables for a given QA pair, we can then proceed to train our model. We use cross-entropy, which minimizes the following loss:

$$L(\vec{\lambda}) = \sum_{\substack{q_n \\ a_n^k \in \mathcal{A}_n}} \sum_{m,i,j} p(t_m^{*ij} | q_n, a_n^k; \mathcal{T}) \cdot \log p(t_m^{ij} | q_n, a_n^k; \mathcal{A}_n, \mathcal{T}) \quad (4.4)$$

Here $p(t_m^{*ij} | q_n, a_n^k; \mathcal{T})$ is the normalized probability of the true alignment scores.

While this is an indirect way to train our model to pick the best answer, in our pilot experiments it worked better than direct maximum likelihood or ranking with hinge loss, achieving a training accuracy of almost 85%. Our experimental results on the test suite, presented in the next section, also support the empirical effectiveness of this approach.

4.5.2 Features

The features we implement are summarized in Table 4.4. In the context of this thesis, it should be noted that almost all of these features leverage some structural or relational property of the tables as described in Section 4.2. The features in our model compute statistics between question-answer pairs and different structural components of tables.

Even though they are manually engineered, thinking about the semantic relations as a basis for knowledge representation can facilitate the process of feature engineering. They can also lead to a model that replicates this process by automating the process of feature generation (see Section 4.6).

While the features in Table 4.4 are weighted and summed for each cell individually, they can capture more global properties such as scores associated with tables, rows or columns in which the specific cell is contained. Features are divided into four broad categories based on the level of granularity at which they operate. In what follows we give some details of Table 4.4 that require further elaboration.

Level	Feature	Description	Intuition	S-Var	Compct
Table	Table score	Ratio of words in \mathbf{t} to $\mathbf{q}+\mathbf{a}$	Topical consistency	◇	
	†TF-IDF table score	Same but TF-IDF weights	Topical consistency	◇	●
Row	Row-question score	Ratio of words in \mathbf{r} to \mathbf{q}	Question align	◇	●
	Row-question w/o focus score	Ratio of words in \mathbf{r} to $\mathbf{q}-(\mathbf{a}_r+\mathbf{q}_r)$	Question align	◇	
	Header-question score	Ratio of words in \mathbf{h} to \mathbf{q}	Prototype align	◇	
Column	Column overlap	Ratio of elements in \mathbf{c} and \mathbf{A}	Choices align	◇	●
	Header answer-type match	Ratio of words in \mathbf{c}_h to \mathbf{a}_r	Choices hypernym align	◇	●
	Header question-type match	Ratio of words in \mathbf{c}_h to \mathbf{q}_r	Question hypernym align	◇	
Cell	†Cell salience	Salience of \mathbf{s} to $\mathbf{q}+\mathbf{a}$	QA hypothesis assert	◇	●
	†Cell answer-type entailment	Entailment score between \mathbf{s} and \mathbf{a}_r	Hypernym-hyponym align		●
	Cell answer-type similarity	Avg. vector sim between \mathbf{s} and \mathbf{a}_r	Hypernym-hyponym sim.		

Table 4.4: Summary of features. For a question (\mathbf{q}) and answer (\mathbf{a}) we compute scores for elements of tables: whole tables (\mathbf{t}), rows (\mathbf{r}), header rows (\mathbf{h}), columns (\mathbf{c}), column headers (\mathbf{c}_h) and cells (\mathbf{s}). Answer-focus (\mathbf{a}_r) and question-focus (\mathbf{q}_r) terms added where appropriate. Features marked ◇ denote soft-matching variants, marked with white those marked with a † are described in further detail in Section 4.5.2. Finally, ● features denote those that received high weights during training with all features, and were subsequently selected to form a compact FRETTS model.

4.5.2.1 Soft matching

Many of the features that we implement are based on string overlap between bags of words. However, since the tables are defined statically in terms of a fixed vocabulary (which may not necessarily match words contained in an MCQ), these overlap features will often fail. We therefore soften the constraint imposed by hard word overlap by a more forgiving soft variant. More specifically we introduce a word-embedding based soft matching overlap variant for every feature in the table marked with \diamond . The soft variant targets high recall while the hard variant aims at providing high precision. We thus effectively have almost twice the number of features listed.

Mathematically, let a hard overlap feature define a score $|S_1 \cap S_2| / |S_1|$ between two bags of words S_1 and S_2 . We can define the denominator S_1 here, without loss of generality. Then, a corresponding word-embedding soft overlap feature is given by this formula:

$$\frac{1}{|S_1|} \sum_{w_i \in S_1} \max_{w_j \in S_2} \text{sim}(\vec{w}_i, \vec{w}_j) \quad (4.5)$$

Intuitively, rather than matching a word to its exact string match in another set, we instead match it to its most similar word, discounted by the score of that similarity.

4.5.2.2 Question parsing

We parse questions to find the desired answer-type and, in rarer cases, question-type words. For example, in the question “What form of energy is required to convert water from a liquid to a gas?”, the type of the answer we are expecting is a “form of energy”. Generally, this answer-type corresponds to a hypernym of the answer choices, and can help find relevant information in the table, specifically related to column headers.

By carefully studying the kinds of question patterns in our data, we implemented a rule-based parser that finds answer-types from queries. This parser uses a set of hand-coded regular expressions over phrasal chunks. The parser is designed to have high accuracy, so that we only produce an output for answer-types in high confidence situations. In addition to producing answer-types, in some rarer cases we also detect hypernyms for parts of the questions. We call this set of words question-type words. Together, the question-type and answer-type words are denoted as focus words in the question.

4.5.2.3 TF-IDF weighting

TF-IDF scores for weighting terms are pre-computed for all words in all the tables. We do this by treating every table as a unique document. At run-time we discount scores by table length as well as length of the QA pair under consideration to avoid disproportionately assigning high scores to large tables or long MCQs.

4.5.2.4 Saliency

The saliency of a string for a particular QA pair is an estimate of how relevant it is to the hypothesis formed from that QA pair. It is computed by taking words in the question, pairing them with words in an answer choice and then computing PMI statistics between these pairs from a large

Model	Data	Regents Test	Monarch Test	ESSQ
Lucene	Regents Tables	37.5	32.6	36.9
	Monarch Tables	28.4	27.3	27.7
	Regents+Monarch Tables	34.8	35.3	37.3
	Waterloo Corpus	55.4	51.8	54.4
MLN (Khot et al., 2015)	-	47.5	-	-
FRETS (Compact)	Regents Tables	60.7	47.2	51.0
	Monarch Tables	56.0	45.6	48.4
	Regents+Monarch Tables	59.9	47.6	50.7
FRETS	Regents Tables	59.1	52.8	54.4
	Monarch Tables	52.9	49.8	49.5
	Regents+Monarch Tables	59.1	52.4	54.9

Table 4.5: Evaluation results on three benchmark datasets using different sets of tables as knowledge bases. Best results on a dataset are highlighted in bold.

corpus. A high salience score indicates words that are particularly relevant for a given QA pair hypothesis.

4.5.2.5 Entailment

To calculate the entailment score between two strings, we use several features, such as overlap, paraphrase probability, lexical entailment likelihood, and ontological relatedness, computed with n-grams of varying lengths.

4.5.2.6 Normalization

All the features in Table 4.4 produce numerical scores, but the range of these scores vary to some extent. To make our final model more robust, we normalize all feature scores to have a range between 0.0 and 1.0. We do this by finding the maximum and minimum values for any given feature on a training set. Subsequently, instead of using the raw feature value of a feature f_d , we instead replace it with $(f_d - \min f_d) / (\max f_d - \min f_d)$.

4.5.3 Experimental Results

We train FRETS (Section 4.5.1.1) on the TabMCQ dataset (Section 4.4.1.1) using adaptive gradient descent with an L2 penalty of 1.0 and a mini-batch size of 500 training instances. We train two variants: one consisting of all the features from Table 4.4, the other – a compact model – consisting of the most important features (above a threshold) from the first model by feature-weight. These features are noted by ● in the final column of Table 4.4.

We run experiments on three 4th grade science exam MCQ datasets: the publicly available Regents dataset, the larger but unreleased dataset called Monarch, and a third even larger public

dataset of Elementary School Science Questions (ESSQ)⁷. For the first two datasets we use the test splits only, since the training sets were directly studied to construct the Aristo Tablestore, which was in turn used to generate our TabMCQ training data. On ESSQ we use all the questions since they are independent of the tables. The Regents test set consists of 129 MCQs, the Monarch test set of 250 MCQs, and ESSQ of 855 MCQs.

Since we are investigating semi-structured models, we compare against two baselines. The first is an unstructured information retrieval method, which uses the Lucene search engine. To apply Lucene to the tables, we ignore their structure and simply use rows as plain-text sentences. The score for top retrieved hits are used to rank the different choices of MCQs. The second baseline is the highly-structured Markov-logic Network (MLN) model from Khot et al. (2015) as reported in Clark et al. (2016), who use the model as a baseline⁸. Note that Clark et al. (2016) achieve a score of 71.3 on Regents Test, which is higher than FRETSS’ scores (see Table 4.5), but their results are not comparable to ours because they use an ensemble of algorithms. In contrast, we use a single algorithm with a much smaller collection of knowledge. FRETSS rivals the best individual algorithm from their work.

We primarily use the tables from the Aristo Tablestore as knowledge base data in three different settings: with only tables constructed for Regents (40 tables), with only supplementary tables constructed for Monarch (25 tables), and with all tables together (all 65 tables; see Section 4.2.2). For the Lucene baseline we also experiment with several orders of magnitude more data by indexing over the 5×10^{10} words Waterloo corpus compiled by Charles Clarke at the University of Waterloo. Data is not a variable for MLN, since we directly cite results from Clark et al. (2016).

The word vectors we used in soft matching feature variants (i.e., \diamond features from Table 4.4) for all our experiments were trained on 300 million words of Newswire English from the monolingual section of the WMT-2011 shared task data. These vectors were improved post-training by retrofitting (Faruqui et al., 2014) them to PPDB (Ganitkevitch et al., 2013).

The results of these experiments is presented in Table 4.5. All numbers are reported in percentage accuracy. We perform statistical significance testing on these results using Fisher’s exact test with a p-value of 0.05 and report them in our discussions.

First, FRETSS – in both full and compact form – consistently outperforms the baselines, often by large margins. For Lucene, the improvements over all but the Waterloo corpus baseline are statistically significant. Thus FRETSS is able to capitalize on data more effectively and rival an unstructured model with access to orders of magnitude more data. For MLN, the improvements are statistically significant in the case of *Regents* and *Regents+Monarch* tables. FRETSS is thus performing better than a highly structured model while making use of a much simpler data formalism.

Our models are able to effectively generalize. With Monarch tables, the Lucene baseline is little better than random (25%). But with the same knowledge base data, FRETSS is competitive and sometimes scores higher than the best Lucene or MLN models (although this difference is statistically insignificant). These results indicate that our models are able to effectively capture

⁷<http://aristo-public-data.s3.amazonaws.com/AI2-Elementary-NDMC-Feb2016.zip>

⁸We do not re-implement the MLN, and therefore only cite results from previous work on part of our test suite.

Model	REG	MON	ESSQ
FRETS	59.1	52.4	54.9
w/o tab features	59.1	47.6	52.8
w/o row features	49.0	40.4	44.3
w/o col features	59.9	47.2	53.1
w/o cell features	25.7	25.0	24.9
w/o \diamond features	62.2	47.5	53.3

Table 4.6: Ablation study on FRETS, removing groups of features based on level of granularity. \diamond refers to the soft matching features from Table 4.4. Best results on a dataset are highlighted in bold.

both content and structure, reasoning approximately (and effectively) when the knowledge base may not even contain the relevant information to answer a question. The Monarch tables themselves seem to add little value, since results for Regents tables by themselves are just as good or better than Regents+Monarch tables. This is not a problem with FRETS, since the same phenomenon is witnessed with the Lucene baseline. It is noteworthy, however, that our models do not suffer from the addition of more tables, showing that our search procedure over table cells is robust.

Finally, dropping some features in the compact model doesn't always hurt performance, in comparison with the full model. This indicates that potentially higher scores are possible by a principled and detailed feature selection process. In these experiments the difference between the two FRETS models on equivalent data is statistically insignificant.

4.5.3.1 Ablation Study

To evaluate the contribution of different features we perform an ablation study, by individually removing groups of features from the full FRETS model, and re-training. Evaluation of these partial models is given in Table 4.6. In this experiment we use all tables as knowledge base data.

Judging by relative score differential, cell features are by far the most important group, followed by row features. In both cases the drops in score are statistically significant. Intuitively, these results make sense, since row features are crucial in alignment to questions, while cell features capture the most fine-grained properties. It is less clear which among the other three feature groups is dominant, since the differences are not statistically significant. It is possible that cell features replicate information of other feature groups. For example, the cell answer-type entailment feature indirectly captures the same information as the header answer-type match feature (a column feature). Similarly, salience captures weighted statistics that are roughly equivalent to the coarse-grained table features. Interestingly, the success of these fine-grained features would explain our improvements over the Lucene baseline in Table 4.5, which is incapable of such fine-grained search.

4.6 TabNN: QA over Tables with Neural Networks

We have shown that tables can be effectively used as a source of semi-structured background knowledge in Question Answering. In particular, we demonstrated that the implicit relations in tables can be leveraged as insight into developing a feature rich model for QA.

However, the FRETTS MCQ solver that we implemented and evaluated in Section 4.5 uses a set of manually engineered features to find relevant cells in tables for answering questions. In keeping with the over-arching goal of this thesis, we'd like to also explore the replacement of manual feature engineering with automatic representation learning, while still leveraging the relational properties of tables.

Some important question must be asked, in this context. Does such a replacement in fact result in a model that is as effective or better at solving MCQs? What are the trade-offs between manually engineering features versus inducing them automatically via representation learning, – while continuing to leverage structural and relational information. What are the repercussions in terms of performance, generality and dependence on data?

To answer these questions we develop a neural network model that mimics some of the relational features we developed for FRETTS. This model requires no feature engineering. Instead it leverages the structural components of tables (i.e. rows and columns) and their relations to MCQ elements to learn representations that reproduce the alignment signal from our TabMCQ dataset.

4.6.1 The Model

Recall the discussion about the relational structure in tables and its semantics from Section 4.2.1, and the translation of these semantics into manually engineered features in Section 4.5.2. At a high level, one can think about a question as being *mapped* to a specific row, while its choices are *mapped* to a specific column and the correct answer is *mapped* to the cell at the intersection of these specific row and column. In fact this is the process of thinking through which we guide our annotators in crowd-sourcing MCQs, as described in Section 4.4.1.

Hence, from a representation learning point of view, it makes sense to develop a neural network model that learns each of these mappings in a joint framework. This is precisely what we do with TabNN. We compute affinity scores between question and rows, choices and columns and answer and cells with three distinct loss functions. Each of these affinities are much like the feature score computed by FRETTS for a particular cell given an MCQ. Jointly, they capture how likely the cell at the intersection of a row and column is to containing information relevant to asserting a QA pair.

In what follows, we describe the parametrization of each of these three mappings, but we begin by describing shared components of the model.

4.6.1.1 Shared Model Components

Shared components of the model describe elements that are reused across the three mappings that we attempt to learn. They represent smaller units of representation, across which we attempt to share statistical strength, reduce the parameter space of our model and increase overall robustness. The shared representations are also what tie the three different objectives together.

Cell Embedding

These embedding parameters take the input words of cell elements in tables and map them into dense vectorial representations. Assuming a sequence of words w_1, \dots, w_n in a specific cell, this layer converts it into a sequence of vectors v_1, \dots, v_n , each with dimension d_1 .

The vector space of embeddings V can be learnable, or may be pre-trained and fixed.

Cell LSTM

These parameters convert the sequence of vector representations v_1, \dots, v_n that we generated from the embeddings into a single vector V_c that captures the *meaning* of the entire sequence. The LSTM that we use to perform this encoding makes this conversion with the sequence of following gated operations. A hidden state and cell state (not to be confused with the cell of a table), h_t and C_t are maintained throughout the sequence at every time step t .

The first step is to compute a forget gate to decide how much of the information from the previous state must actually be carried forward. This is done by:

$$f_t = \sigma(W_f \cdot [h_{t-1}, v_t] + b_f) \quad (4.6)$$

Parallely we can also compute what new information must be stored in the current cell state. This involves two related parts. The first is the “input gate layer” which decides what values must be updated. The second is a new candidate for the current state. Equations for these two parts are given below:

$$i_t = \sigma(W_i \cdot [h_{t-1}, v_t] + b_i) \quad (4.7)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, v_t] + b_C) \quad (4.8)$$

The forget and input vectors are then used to weight the previous cell state and current candidate state to produce the new current cell state:

$$C_t = f_t \diamond C_{t-1} + i_t * \tilde{C}_t \quad (4.9)$$

where \diamond is the element-wise multiplication operation between vectors.

Finally, we compute the new hidden state. This is done by an intermediate “output gate layer” that is then combined with the new current cell state:

$$o_t = \sigma(W_o \cdot [h_{t-1}, v_t] + b_o) \quad (4.10)$$

$$h_t = o_t \diamond \tanh(C_t) \quad (4.11)$$

By repeatedly applying these series of gated operations with each input vector at every time step t we obtain a final hidden state h_n . This hidden state is the single vector V_c that we are seeking, and is a representation of the entire cell.

The set of parameters W_i, b_i, W_C, b_C, W_o and b_o must all be trained.

Choice Embedding

These embeddings are very similar to the cell embeddings, except that they convert words in the different choices of the MCQ into sequences of vectors. As with the Cell Embeddings, this vector space of embeddings can be learned or pre-trained and fixed.

Choice LSTM

Again, these parameters convert the sequence of choice embeddings into a single vector representing the entire choice. The architecture of the LSTM network used to do this is identical to the one described above. Also as before, the set of parameters of the LSTM must all be trained, and are different from the LSTM parameters of the Cell LSTM.

4.6.1.2 Question-Row Objective

The question-row objective attempts to learn a set of discriminative representations that can distinguish when a question is in fact aligned to a row, and when it is not. The architecture of this model component is depicted in Figure 4.2. It comprises several parts that we describe below.

Question Embedding & LSTM

This part of the model performs the same function for questions as the shared components in Section 4.6.1.1 do for table cells and answer choices. We essentially take a sequence of words in the question as input and produce a single vector that encodes the question as an output. This is done with an LSTM on top of an embedding layer.

Row Encoding

The input to this component of the model is different from others, because it is already a sequence of vectors and not words. We take the sequence of vectors produced by the *Cell LSTM* for each cell in the row and feed it into another LSTM that combines them. This is effectively a stacked LSTM, which combines a sequence of sequences to produce a single vector that is a representation of the entire row.

Question-Row Prediction

We concatenate the question and row vectors and feed the resulting vector through a sigmoid activation to produce a response variable. The signal we seek to imitate with this response is the row-alignment score as described in Section 4.5.1.1.

4.6.1.3 Choices-Column Objective

The choices-column objective computes the affinity between the column of a table and the set of choices of the MCQ under consideration. The architecture of this model component is depicted in Figure 4.3. It comprises several parts that we describe below.

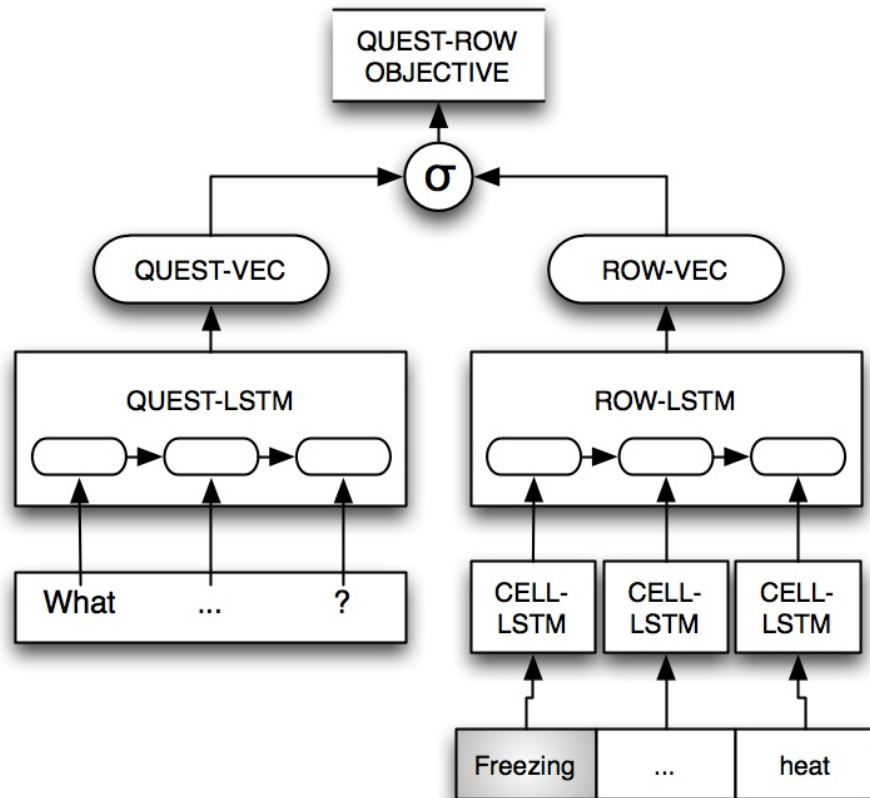


Figure 4.2: The component of the TabNN model that embeds question and row in a joint space and predicts an alignment score between the two. Embedding layers are omitted for visual compactness.

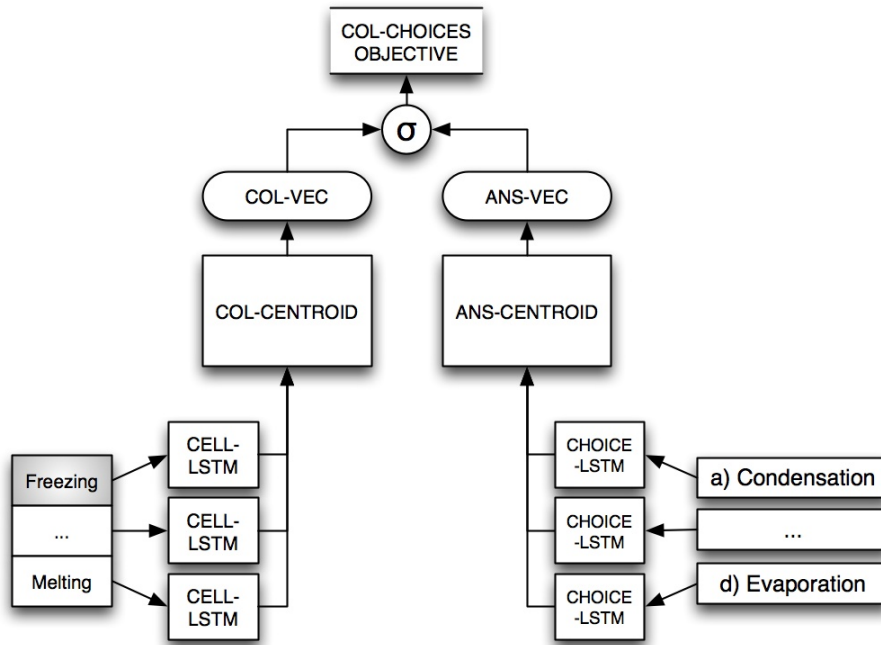


Figure 4.3: The component of the TabNN model that embeds choices and column in a joint space and predicts an alignment score between the two. Embedding layers are omitted for visual compactness.

Choices Encoding

This model component uses the shared Choice Embedding and LSTM parameters to convert the four different choices into four distinct vectors. It then computes the centroid of these four choices by averaging their four corresponding vectors.

Column Encoding

This model component performs a similar operation on columns. It uses the shared Cell Embedding and LSTM parameters to generate individual vectors for each cell in a column and then computes a centroid vector that is the average of all these cells.

Choices-Column Prediction

We concatenate the two centroid vectors of the choices and column and pass the resulting vector through a sigmoid activation. The response variable seeks to imitate the signal obtained from the alignment of columns to choices as described in Section 4.5.1.1.

4.6.1.4 Answer-Cell Objective

Finally, the answer-cell objective predicts whether a cell contains the answer to a question or not. This is done by concatenating cell and answer vectors obtained through the shared embedding

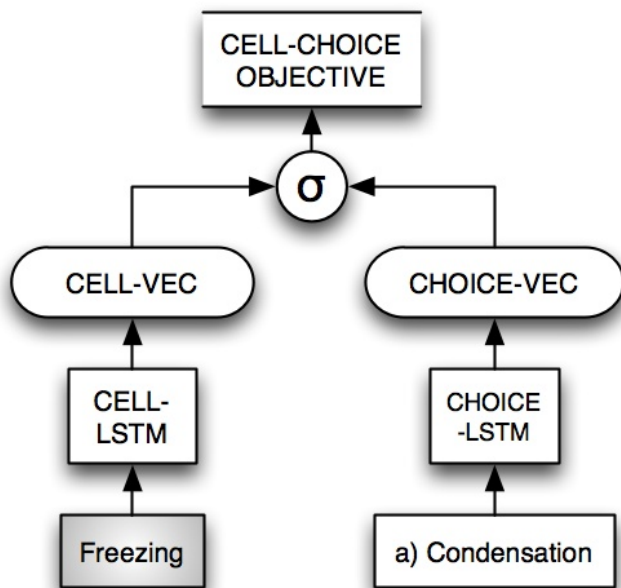


Figure 4.4: The component of the TabNN model that embeds a single choice and table cell in a joint space and predicts an alignment score between the two. Embedding layers are omitted for visual compactness.

and LSTM components of the model and passing the resulting vector through a sigmoid activation. The response variable here is imitating the signal obtained from the alignment of a cell to a particular answer as described in Section 4.5.1.1.

This model objective is pictorially represented in Figure 4.4

Intuitively our model is thus capturing three distinct relational affinities. They are, namely, between questions and rows, the multiple choices and a column, and a specific answer and a particular cell in the tables. These affinities are defined over the structural properties and are thus inspired from the same properties we use to engineer our features manually in the FRETTS model (Chapter 4.5).

At test time, it should be clear that for our model to predict a high score for a particular answer pair based on some evidence from the tables, the score associated with each of the three different objectives must also be high.

4.6.2 Training

Since our proposed TabNN model follows the same design template as FRETTS, namely by scoring individual rows, columns and cells for relevance to QA pairs, we can use the same training schema as before (see Section 4.5.1.1). In summary this objective is the cross-entropy loss between a hypothesized distribution and the true one in training.

The hypothesized distributions are obtained from the scores we assign to the three objectives of Section 4.6.1 with our model. The true distribution are obtained in the same way as with

FRETS: by converting alignments annotations between MCQs and table entries into numerical scores. By considering the discrepancy between the hypothesized and true distributions thus defined, we back-propagate the error to update all the model parameters.

An important distinction from the model in FRETS is that we do not consider representations over entire tables. Neither do we search over every row and column of all the tables. These choices are based on the computational complexities that are introduced by considering a model that either directly, or through some combination of smaller components, represents an entire table.

Instead, we use the following criteria to decide which tables we search over and featurize, for every QA pair:

- At training time
 - 1 correct table
 - 2 randomly chosen tables
- At test time
 - 3 top tables from TF-IDF ranking

The TF-IDF ranking we refer to is the same as the TF-IDF table feature defined in our FRETS model (see Table 4.4 in Section 4.5.2). On a section of the training set that we used to develop our FRETS features, this ranker selected the correct table for a particular MCQ with over 90% accuracy.

4.6.3 Experiments and Evaluation

We now turn to the task of evaluating the TabNN model. We experiment on the publicly available and largest of the three datasets that we used to evaluate FRETS (the ESSQ dataset) and compare the results to the manually feature engineered model. We also discuss differences between the two models and attempt to explain the discrepancy between the results in the context of these differences.

We begin by providing some implementational details.

4.6.3.1 Implementational Details

The training data is obtained by converting the TabMCQ dataset and its corresponding alignment data into response variables as described in Section 4.5.1.1. To normalize the data for input to our neural network architecture we use padding, truncation and masking.

Padding and truncation ensure that the size of each input is constant. So, for example, if a question is too short it is *padded* to be of the constant size and if it is too long it is *truncated* to be of the same size. We set a maximum size of 20 tokens for answer choices and table cells, a maximum of 8 cells in a column, a maximum of 5 cells in a row, and a maximum of 30 tokens in a question.

We also apply masking to the fixed length input sequences. This process indicates time-steps in a sequence that need to be skipped. This works in combination with padding, where the

Embeddings	LSTM Size	Number of Parameters
Learned	160	4,377,603
	320	7,147,843
Pre-trained	160	1,131,843
	320	3,902,083

Table 4.7: A summary of the different TabNN model variants we train along with the number of parameters learned with each model.

additional time-steps introduced for maintaining constant size are to be skipped when performing LSTM computations.

We implement a total of four distinct models, which are formed from the cross-product of two dimensions of distinction. The first dimension consists of two variants in the types of embeddings we use: learned and pre-trained.

The learned embeddings variant – as the name suggests – randomly initializes the embedding matrices and updates the parameters through training. In this variant the embedding layers used in different parts of the model are correspondingly different.

But the vocabulary sets in training and testing might be different. In this case the learned embeddings, which are obtained from training, will not cover the words encountered at test time. To account for this discrepancy we also experiment with a set of pre-trained embeddings that cover a large vocabulary.

The pre-trained embeddings are obtained by training regular Skip-gram word vectors (Mikolov et al., 2013a) over Wikipedia, then retrofitting (Faruqui et al., 2014) them for enhanced quality. We use this single set of embeddings for all components of the model that require an embedding layer, and we do not update the parameters during training. This significantly reduces the size of the model in terms of the number of parameters.

In both cases we maintain a constant embedding size of 80 for all embedding layers.

The second dimension of distinction, which deals with the size of the LSTM encoding, also yields two variants. We experiment with 160 and 320 for compact and complex LSTM models respectively.

These 4 models are summarized in Table 4.7 along with the number of parameters that they contain.

To avoid overfitting, we apply regularization throughout our model architecture. While training we use a dropout factor of 0.5, which is the recommended standard value (Srivastava et al., 2014), to regularize the different LSTMs. We also use L-2 regularization, with a parameter value of 0.5, applied to all the embedding matrices.

To maximize the joint objective of the model, we perform stochastic gradient descent on batch sizes of 512 and train for a total of 10 epochs over the entire training set.

4.6.3.2 Results

The results of the TabNN model are presented in Table 4.8. We present training values to gain insights into the model, but it should be stressed that high values are by no means an indication

Model	Embeddings	LSTM Size	Training Accuracy	ESSQ Test Accuracy
TabNN	Learned	160	66.5%	25.3%
		320	68.2%	24.3%
	Pre-trained	160	42.2%	26.2%
		320	42.6%	26.1%
FRETS	-	-	-	54.9%

Table 4.8: Evaluation of the TabNN model. Training results are provided to gain insight into the models and should not be taken to represent goodness of the models in any way. The best FRETS model’s test results are given as a comparison.

Model	Embeddings	LSTM Size	Training Acc.	10% TabMCQ Test Acc.	ESSQ Test Acc.
TabNN	Learned	160	68.0%	56.8%	24.4%
		320	71.1%	50.8%	24.8%
	Pre-trained	160	42.2%	31.9%	27.1%
		320	43.7%	30.9%	28.4%

Table 4.9: Evaluation of the TabNN model, trained on only 90% of the TabMCQ dataset. Test results on the 10% held out portion of TabMCQ show that the model itself is capable of learning, and that it’s architecture is sound. There is, however, a severe domain mismatch between TabMCQ and the ESSQ test set.

of a good model.

In fact we see this in the results, since evaluation on the ESSQ test set show that all variants of the TabNN model perform no better than random guessing (at 25%). The high training values, coupled with the poor test results seem to indicate that the TabNN architecture is massively overfitting to the training set, and is incapable of generalizing to test.

But is this true, considering that we use regularization throughout our model, both with dropout and with L-2 penalties? To test whether this is indeed the case, we run a second batch of experiments where we hold out a randomly selected 10% of training questions from TabMCQ as a secondary test set. We then train TabNN only on the remaining 90% of TabMCQ and then evaluate the results.

These results are presented in Table 4.9. While the test results on ESSQ are still no different than random, the test results on the 10% held out portion of TabMCQ are much better, with the best model achieving a score of 56.8%. This demonstrates that our model itself is capable of learning effectively, but there is a severe mismatch between the TabMCQ and ESSQ datasets that the TabNN model is incapable of overcoming – and the FRETS model is.

There are some other things to note, across both Tables 4.8 and 4.9. Firstly, as expected, having a more complex model, with an LSTM embed dimension of 320 leads to better training results. But this does not transfer to test, where the simpler models perform better (on 10% TabMCQ test) or at least no worse (on ESSQ test).

Secondly, using pre-trained embeddings does not seem to be justified. While the test results for corresponding models on ESSQ are *marginally* better than learning embeddings from scratch,

they are a *lot* worse when tested on 10% TabMCQ data. Thus not much power is gained by having access to pre-trained embeddings with a larger vocabulary.

Which leads to the third insight: that more training data, in this case, would not help. Instead, *different* and more *diverse* training data is required. This insight is also confirmed by the fact that TabNN, whether trained on the full TabMCQ dataset or only 90% has similar accuracies on training and ESSQ test.

4.6.3.3 Discussion on Differences with FRETS

The results from Section 4.6.3.2 have demonstrated a significant difference between the FRETS and TabNN model. In summary, while they are both capable of leveraging the relations and structure within tables to solve MCQs, FRETS does so by abstracting over both content *and* structure, while TabNN can only abstract over structure and is much more dependent on content.

This finding is in keeping with general expectations of neural network models. While they have potentially infinite capacity to approximate arbitrarily complex functions, they require vast amounts of diverse training data in order to be able to generalize from training to test.

In our case, the TabMCQ (training) dataset is not sufficiently diverse to accommodate the very different ESSQ (test) dataset. For example, the average length of questions in ESSQ is 24.4 words, with some questions containing lengthy explanatory paragraphs before the actual question. In contrast, the crowd-sourced TabMCQ dataset only has an average of only 11 words, with Turkers only creating direct MCQs.

The TabNN model must deal with these discrepancies by automatic means such as truncation and padding. There is no guarantee that truncation maintains all the information in the question relevant to finding an answer. In contrast, the feature-rich FRETS model relies on sentence splitting and manually crafted regular expression patterns over phrase chunks to extract relevant parts of the question.

Another issue is the content of tables. While FRETS demonstrated itself capable of answering questions that may not even be contained in tables, TabNN did not. In fact, TabNN only performed well on the 10% held out portion of the TabMCQ dataset, when the answers were guaranteed to be found in tables. Unfortunately, our proposed solution of using pre-trained embeddings to counter the mismatch in tables between training and test did not work because learning power is gained by learning the embeddings on in-domain data.

Again, this indicates the need for TabNN to have access to more diverse training data – this time in the form of tables.

In conclusion, the TabNN model is architecturally sound and capable of leveraging the relations within tables to learn representations over table elements for solving MCQs. We thus, re-iterate the central claim of this thesis: that representation learning can benefit from a relation-centric view of semantics. Finding and leveraging the correct relations for a particular domain, specifically, can lead to better models.

However, in comparison with FRETS, TabNN is much more reliant on content and less able to adapt to significantly different test sets. This is in keeping with the general understanding of neural network models which require very large and diverse training sets to properly generalize to unseen examples.

4.7 Conclusion

Tables represent a unique and interesting domain for the study of relational structure in representation learning, with interesting semantic properties emerging from relationships between row and column elements. These properties lead to two separate contributions in this chapter.

Firstly, we leveraged the semi-structured formalism of tables with text entries to create a dataset of MCQs. The semantics of this structure were used as scaffolding information to guide non-expert crowd-sourcing annotators to easily create a high-quality MCQ dataset. The relation-centric annotation setup also led us to harvest rich alignment information of these MCQs table elements, with no additional effort to annotators.

We have released our dataset of more than 9000 MCQs and their alignment information to the research community. To the best of our knowledge, it is the largest elementary school science MCQ dataset of its kind, containing a unique set of alignments to tables. We believe it offers interesting challenges that go beyond the scope of this chapter – such as question parsing, or textual entailment – and are exciting avenues for future research.

Secondly, we developed two kinds of feature-driven models that predicted the previously generated alignment information to answer new unseen MCQs. The first used manually engineered features, while the second automatically induced features within a neural network model. Crucially however, both kinds of models leveraged the relational and structural properties of tables and thus fit into the broader scope of this thesis.

We evaluated our models on the task of answering MCQs.

On three benchmark sets the manually engineered model, that relied on semi-structured tables, consistently and significantly outperformed an unstructured and a highly-structured baseline. We also showed that this model was able to generalize over content and structure, thus reasoning about questions whose answer was not even contained in the tables that we used as a knowledge repository. This demonstrated that the rich relational structure inherent in tables can in fact be usefully featurized to perform QA.

The neural network model also demonstrated the ability to leverage the relations within tables for learning representations jointly over table elements and MCQs. It showed, that with no feature engineering whatsoever, we were able to obtain promising results on a held-out portion of the training data that we used as a test set. However, it was much more dependent on content than the FRETTS model, performing poorly on a test set that had a distinctly different profile to the training data. These findings were in keeping with the general understanding of neural network models, that require large and diverse training sets to learn optimally. Our training set was simply not diverse enough.

The mixed results from the neural network model do not, however, counter the central claims of this thesis. This is because we still demonstrated that we benefitted from a relation-centric view of our domain – solving MCQs with tables. The only difference, in this case, was that in our test setup a manually feature-engineered model (which, it should be noted, centers around the relations and structure of tables) was better than a model that learned feature representations automatically. Our analyses indicate, though, that with a larger and (especially) more diverse training set, the neural network model is likely to perform much better.

Chapter 5

Embedded Frame Lexicons

Learning Event Structure through Unsupervised Lexicon Induction

This thesis is concerned with using semantic relations to learn more expressive models of semantics. Relations encompass a very broad range of phenomena in language and data formalisms, and in Chapter 2 we have attempted to provide a taxonomy of these occurrences of relations as it pertains to our work. This taxonomy serves, both as a technical summary of this thesis, as well as a high-level guideline on how to think about using relations in representation learning of a new domain or problem.

However, an important research issue when dealing with relations for representation learning is the relations themselves. In particular, for a given language phenomenon or data formalism, what happens when the relations are ill-defined, or no annotated data exists?

More generally, what happens when we must induce rather than use a pre-existing set of relations? This is an important question, in the context of this thesis because a comprehensive treatment of semantic relations requires dealing with such scenarios as well.

For example, a low-resource language may simply not have annotated data containing rich structure – such as an ontology – or even partial structure – such as tables. Or consider the example of semantic roles: it is a disputed matter what the exact set of relations to encode all possible semantic roles should be (Kingsbury and Palmer, 2002; Baker et al., 1998).

This and a major part of Chapter 6 will thus be concerned with developing representation learning models that do not leverage or embed a pre-defined annotated set of relations. Our models will rather work on inducing relations as part of the learning process.

Because we are inducing an unknown set of relations, we will also attempt to tackle some general questions about the nature of these relations. How many are there? What are they? How do they organize the elements in the domain to one another? Although, it may be noted that finding answers to these questions does not preclude the *usage* of the induced relations for practical and effective representation learning, since even if the relations are not interpretable they may still produce good results in downstream applications.

In this chapter we turn to the problem of providing a representation learning solution to the relations in a frame lexicon. This is an apt domain, since verbs undoubtedly have relational arguments, as supported by the body of linguistic literature devoted to studying argument re-

alization (Jackendoff, 1987; Moens and Steedman, 1988; Levin and Rappaport Hovav, 2005). However, the specific number and nature of these arguments remains contested (Fillmore, 1967; Sowa, 1991), with solutions ranging from a few prototypical PropBank style relations (Kingsbury and Palmer, 2002) to an effectively open-ended set of highly specific and fine-grained roles in FrameNet (Baker et al., 1998).

These semantic lexicons contain information about predicate-argument frame structure. These frames encode relations that capture knowledge about the affinity of predicates for certain types of arguments. They also specify their number and their semantic nature, regardless of syntactic realization.

For example, PropBank specifies frames in the following manner:

- eat \rightarrow [agent]₀, [patient]₁
- give \rightarrow [agent]₀, [theme]₁, [recipient]₂

These frames provide semantic information such as the fact that “eat” is transitive, while “give” is di-transitive, or that the beneficiary of one action is a “patient”, while the other is a “recipient”.

This structural knowledge is crucial for a number of NLP applications. Information about frames has been successfully used to drive and improve diverse tasks such as information extraction (Surdeanu et al., 2003), semantic parsing (Das et al., 2010) and question answering (Shen and Lapata, 2007), among others.

However, building these frame lexicons is very expensive and time consuming. Thus, it remains difficult to port applications from resource-rich languages or domains to data impoverished ones. The NLP community has tackled this issue along two different lines of unsupervised work.

At the local level, researchers have attempted to model frame structure by the selectional preference of predicates for certain arguments (Resnik, 1997; Séaghdha, 2010). This can be categorized as token level affinity of predicates for their relational arguments. For example, on this problem a good model might assign a high probability to the word “pasta” occurring as an argument of the word “eat”.

Contrastingly, at the global level, work has focussed on inducing frames by clustering predicates and arguments in a joint framework (Lang and Lapata, 2011a; Titov and Klementiev, 2012b). This can be categorized as the type level affinity between predicates and their semantically linked arguments. So for example, one might be interested in associating predicates such as “eat”, “consume”, “devour”, with a joint clustering of arguments such as “pasta”, “chicken”, “burger”.

While these two classes of solutions have been useful for several problems, they also have shortcomings. Selectional preference modelling only captures local predicate-argument affinities, but does not aggregate these associations to arrive at a structural understanding of frames.

Meanwhile, frame induction performs clustering at a global level. But most approaches tend to be algorithmic methods (or some extension thereof) that focus on semantic role labelling. Their lack of portable features or model parameters unfortunately means they cannot be used to solve other applications or problems that require lexicon-level information – such as information extraction or machine translation. Another limitation is that they always depend on high-level linguistic annotation, such as syntactic dependencies, which may not exist in resource-poor settings.

Thus, in this chapter we propose to combine the two approaches to induce a frame semantic lexicon in a minimally supervised fashion, using nothing more than unlabeled predicate-argument word pairs as input data. Additionally, we will learn an embedded lexicon that jointly produces embeddings for predicates, arguments and an automatically induced collection of latent slots. These slots are the induced relations that we learn and investigate in this chapter.

Because of the productivity of argument realization, the embedded lexicon we propose, provides flexibility for usage in downstream applications, where predicate-argument affinities can be computed at will.

To jointly capture the local and global streams of knowledge we propose a novel integration between a predictive embedding model and the posterior of an Indian Buffet Process. The embedding model maximizes the predictive accuracy of predicate-argument selectional preference at the local token level, while the posterior of the Indian Buffet process induces an optimal set of latent slots at the global type level that capture the regularities in the learned predicate embeddings. In the context of this thesis, it should be noted that both components of our models are centered on the relational properties between predicates and arguments and attempt to capture these properties in a latent fashion.

We evaluate our approach and show that our models are able to outperform baselines on both the local token and global type level of frame knowledge. At the local level we score higher than a standard predictive embedding model on selectional preference, while at the global level we outperform a syntactic baseline on lexicon overlap with PropBank. Finally, our analysis on the induced latent slots yields insight into some interesting generalities that we are able to capture from unlabeled predicate-argument pairs.

5.1 Related Work

The work in this chapter relates to research on identifying predicate-argument structure in both local and global contexts. These related areas of research correspond to the NLP community's work respectively on selectional preference modelling and semantic frame induction (which is also known variously as unsupervised semantic role labelling or role induction).

Selectional preference modelling seeks to capture the semantic preference of predicates for certain arguments in local contexts. These preferences are useful for many tasks, including unsupervised semantic role labelling (Gildea and Jurafsky, 2002) among others.

Previous work has sought to acquire these preferences using various means, including ontological resources such as WordNet (Resnik, 1997; Caramita and Johnson, 2000), latent variable models (Rooth et al., 1999; Séaghdha, 2010; Ritter et al., 2010) and distributional similarity metrics (Erk, 2007). Most closely related to our contribution is the work by Van de Cruys (2014) who use a predictive neural network to capture predicate-argument associations.

To the best of our knowledge, our research is the first to attempt using selectional preference as a basis for directly inducing semantic frames.

At the global level, frame induction subsumes selectional preference by attempting to group arguments of predicates into coherent and cohesive clusters. While work in this area has included diverse approaches, such as leveraging example-based representations (Kawahara et al., 2014) and cross-lingual resources (Fung and Chen, 2004; Titov and Klementiev, 2012b), most attempts

have focussed on two broad categories. These are latent variable driven models (Grenager and Manning, 2006; Cheung et al., 2013) and similarity driven clustering models (Lang and Lapata, 2011a,b).

Our work includes elements of both major categories, since we use latent slots to represent arguments, but an Indian Buffet process induces these latent slots in the first place. The work of Titov and Klementiev (2012a) and Woodsend and Lapata (2015) are particularly relevant to our research. The former use another non-parametric Bayesian model (a Chinese Restaurant process) in their work, while the latter embed predicate-argument structures before performing clustering.

Crucially, however all these previous efforts induce frames that are not easily portable to applications other than semantic role labelling (for which they are devised). Moreover, they rely on syntactic cues to featurize and help cluster argument instances. To the best of our knowledge, ours is the first attempt to go from unlabeled bag-of-arguments to induced frame embeddings without any reliance on annotated data.

5.2 Joint Local and Global Frame Lexicon Induction

In this section we present our approach to induce a frame lexicon with latent slots. The intuition is to jointly capture local token level and global type level associations between predicates and arguments to arrive at a representation that best explains the input data.

Following prior work on frame induction (Lang and Lapata, 2011a; Titov and Klementiev, 2012a), the procedural pipeline can be split into two distinct phases: argument identification and argument clustering.

As with previous work, we focus on the latter stage, and assume that we have unlabeled predicate-argument structure pairs. These could be obtained from gold standard annotation or through heuristic means (Lang and Lapata, 2014), although the model we describe is agnostic to this choice.

We begin with preliminary notation. Consider a vocabulary of predicate types $P = \{p_1, \dots, p_n\}$ and contextual argument types $A = \{a_1, \dots, a_m\}$. It should be noted that in this work, we assume argument chunks are broken down into individual words, – to increase training data size – but the model remains agnostic to this decision. Then let $C = \{(p_1, a_1), \dots, (p_N, a_N)\}$ be a corpus of predicate-argument word token pairs.

For example, consider sentence fragments such as “eat pasta with a fork” or “take bath in a tub”. Since we split argument chunks to increase training data size, our corpus would essentially consist of pairs of words such as “eat, pasta”, “eat, fork” “take, bath”, “take, tub” etc. Given a large corpus of this form, we will attempt to learn an optimal set of model parameters θ that maximizes a regularized likelihood over the corpus. That is, we will learn a model that best explains the data.

The model parameters include $V = \{v_i \mid \forall p_i \in P\}$ an $n \times d$ embedding matrix for the predicates and $U = \{u_i \mid \forall a_i \in A\}$ an $m \times d$ embedding matrix for the arguments. V and U are basically the word vectors, in two distinct vector spaces, of the predicates and arguments respectively.

Additionally, assuming K latent frame slots we define $Z = \{z_{ik}\}$ as an $n \times k$ binary matrix that represents the presence or absence of the slot k for the predicate i . Intuitively, K defines the

number of the relational arguments we are going to induce and Z is the template of the lexicon that says which predicates take on which relational arguments.

We also define a latent $K \times d$ weight matrix $S = \{s_k \mid 1 \leq k \leq K\}$ that associates a weight vector to each latent slot. This matrix S , intuitively, captures the semantic *signature* of each of the latent relational slots.

Given this set of parameters, the generalized form of the objective we optimize is given by:

$$\hat{\theta} = \arg \max_{\theta} \sum_{(p_i, a_i) \in C} \log \left(\sum_k Pr(a_i | p_i, z_{ik}, s_k) \right) + \log pr_{\theta}(Z|V) \quad (5.1)$$

This objective has two parts: a likelihood term, and a posterior regularizer. Notice, the similarity to the form of Equation 3.3 in Chapter 3.2.2. In that case, just as in the present one, we are attempting to reconcile two streams of information jointly.

In this chapter, we instantiate this objective in a way that accounts for both token and type level associations between predicates and arguments. The first part of the objective (the likelihood term) will be responsible for modelling the predictive accuracy of selectional-preference at a local level, while the second part of the objective (the posterior regularizer) will capture global consistencies for an optimal set of latent relational slots.

We detail the parametrization of each of these components separately in what follows.

5.2.1 Local Predicate-Argument Likelihood

The likelihood term of our model is based on the popular Skip-gram model from Mikolov et al. (2013a) but suitably extended to incorporate the latent frame slots and their associated weights. Specifically, we define the probability for a single predicate-argument pair (p_i, a_i) as:

$$Pr(a_i | p_i) = \sum_k Pr(a_i | p_i, z_{ik}, s_k) = \sum_k z_{ik} \frac{\exp((v_i \odot s_k) \cdot u_i)}{\sum_{a_{i'}} \exp((v_i \odot s_k) \cdot u_{i'})} \quad (5.2)$$

where \odot represents the element-wise multiplication operator. Intuitively, in the likelihood term we weight a general predicate embedding v_i to a slot-specific representations s_k , which then predicts a specific argument u_i . This is graphically represented in Figure 5.1.

5.2.2 Global Latent Slot Regularization

The posterior regularization term in equation 5.1 seeks to balance the likelihood term by yielding an optimal set of latent slots, given the embedding matrix of predicates.

We choose the posterior of an Indian Buffet process (IBP) (Griffiths and Ghahramani, 2005) in this step to induce an optimal latent binary matrix Z . Recall that, in Bayesian learning, the posterior is a product of a prior and likelihood function.

In the present case, the IBP itself places a prior on equivalence classes of infinite dimensional sparse binary matrices. It defines a culinary analogy on how matrices Z can be incrementally built, such that they are legitimate draws from the underlying distribution.

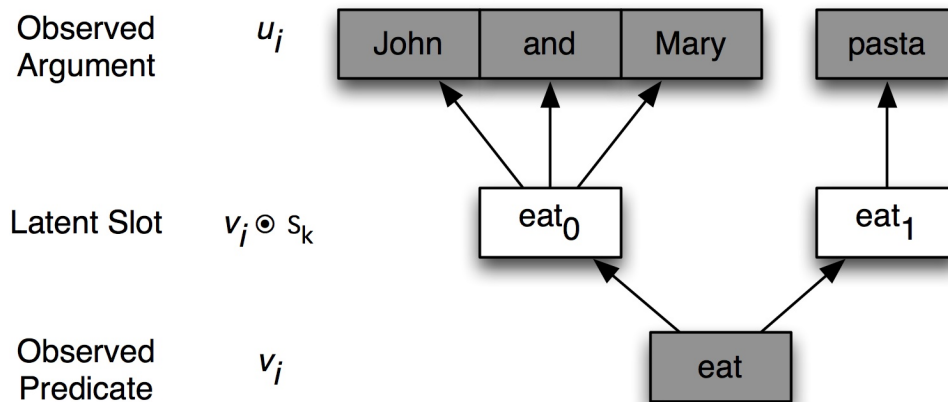


Figure 5.1: The generative story depicting the realization of an argument from a predicate. Argument words are generated from latent argument slots. Observed variables are shaded in grey, while latent variables are in white.

P customers (predicates) enter a restaurant one after another. The restaurant has a buffet consisting of an infinite number of dishes (relational arguments). When the first customer p_1 enters, she takes a serving from the first $\text{Poisson}(\alpha)$ dishes until her plate is full. Every subsequent customer p_i samples previously tasted dishes (previously selected relational arguments) with a probability $\frac{m_k}{i}$ that is proportional to its popularity (m_k is the number of predicates that already selected the i th relational argument). The new customers also serves themselves $\text{Poisson}(\frac{\alpha}{i})$ new dishes.

The elements z_{ik} of the matrix are 1 if customer v_i sampled dish k , and 0 otherwise – that is if predicate i selected argument k or not. In the limit of the number of features $K \lim \infty$, it can be shown that the probability of a matrix being produced by this process is:

$$P(Z) = \frac{\alpha^{K_+}}{\prod_{i=1}^N K^i!} \exp(-\alpha H_N) \prod_{k=1}^{K_+} \frac{(N - m_k!)(m_k - 1)}{N!} \quad (5.3)$$

Here K_+ denotes the non-zero columns of the infinite feature matrix, i.e. those whose $m_k > 0$. Also K^i is the number of new dishes sampled by the i th customer to enter the restaurant. Finally, H_N is the N th harmonic number – which can be computed as $\sum_{j=1}^N \frac{1}{j}$.

This is the form of the prior distribution that we will use to compute our actual feature matrices for verb argument structure induction. Given a suitable likelihood function and some data, inference in an IBP computes a posterior that yields an optimal *finite* binary matrix with respect to regularities in the data.

Setting the *data*, in our case, to be the embedding matrix of predicates V , this gives us precisely what we are seeking. It allows us to find regularities in the embeddings, while factorizing them according to these consistencies. The model also automatically optimizes the number of and relationship between latent slots, rather than setting these a priori.

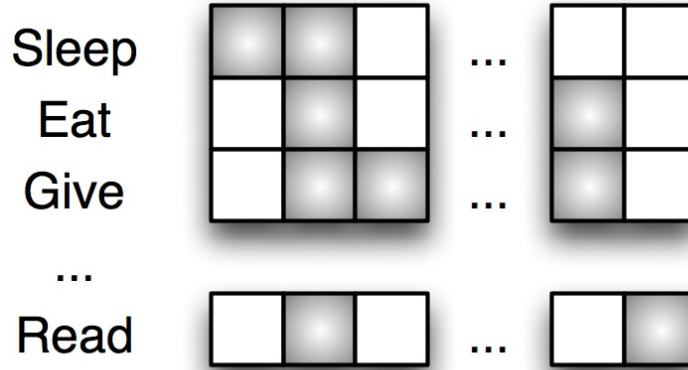


Figure 5.2: Illustration of a potential binary feature matrix (1s colored, 0s blank) learned by IBP. Rows represent verbs from a fixed vocabulary, while columns are of a fixed but arbitrary number and represent latent arguments.

For example, Figure 5.2 illustrates the kind of binary matrix Z that might be produced as a result of running IBP inference over our data.

The IBP encodes other desiderata as well. These include the fact that the matrix Z remains sparse, while the frequency of slots follows a power-law distribution proportional to $\text{Poisson}(\alpha)$. In practise, this captures the power-law distribution of relational slots in real-world semantic lexicons such as PropBank (Palmer et al., 2005). All of these properties stem directly from the choice of prior, and are a natural consequence of using an IBP.

As mentioned above, the second part of computing the posterior is the choice of the likelihood function. In this chapter, we use a linear-Gaussian model as the likelihood function. This is a popular model that has been applied to several problems, and for which different approximate inference strategies have been developed (Doshi-Velez et al., 2009; Doshi-Velez and Ghahramani, 2009). According to this model, the predicate embeddings are distributed as:

$$v_i \sim \text{Gaussian}(z_i W, \sigma_V^2 \mathbf{I}) \quad (5.4)$$

where W is a $K \times d$ matrix of weights and σ_V is a hyperparameter.

A detailed derivation of the posterior of an IBP prior with a linear-Gaussian likelihood is beyond the scope of this work. We point the reader to Griffiths and Ghahramani (2011), who provide a meticulous summary.

5.2.3 Optimization

Since our objective in equation 5.1 contains two distinct components, we can optimize using alternating maximization. Although guaranteed convergence for this technique only exist for convex functions, it has proven successful even for non-convex problems (Jain et al., 2013; Ne-trapalli et al., 2013).

We thus alternate between keeping Z fixed and optimizing the parameters V, U, S in the likelihood component of section 5.2.1, and keeping V fixed and optimizing the parameters Z in the posterior regularization component of section 5.2.2.

Intuitively, what we are doing is alternating between optimizing the local token level and global type level relations between predicates and their arguments. In particular, in one stage we are given a fixed template for our frame lexicon and we attempt to learn the best representations for predicates and their arguments. Then, in the next stage, we are given fixed representations for predicates and we attempt to learn the best template for the frame lexicon that captures regularities in these embeddings. We continue to alternate between these stages until some criteria are met.

In practise, the likelihood component is optimized using negative sampling with EM for the latent slots. In particular we use *hard* EM, to select a single slot before taking gradient steps with respect to the model parameters. This was shown to work well for Skip-gram style models with latent variables by Jauhar et al. (2015) – see Chapter 3.

In the E-Step we find the best latent slot for a particular predicate-argument pair:

$$\hat{k} = \arg \max_k Pr(a_i | p_i, z_{ik}, s_k) \quad (5.5)$$

We follow this by making stochastic gradient updates to the model parameters U, V, S in the M-Step using the negative sampling objective:

$$\log z_{i\hat{k}} \sigma((v_i \odot s_{\hat{k}}) \cdot u_i) + \sum_l \mathbb{E}_{a_{i'} \sim Pr_n(a)} [\log z_{i\hat{k}} \sigma((v_i \odot s_{\hat{k}}) \cdot u_{i'})] \quad (5.6)$$

where $\sigma(\cdot)$ is the sigmoid function, $Pr_n(a)$ is a unigram noise distribution over argument types and l is the negative sampling parameter.

As for optimizing the posterior regularization component, it is computationally intractable to perform IBP inference exactly. Therefore, an approximate inference technique such as Gibbs sampling must be used. In Gibbs sampling we iteratively sample individual z_{ik} terms from the posterior:

$$Pr(z_{ik} | X, Z_{-ik}) \propto Pr(X | Z) \cdot Pr(z_{ik} | Z_{-ik}) \quad (5.7)$$

where Z_{-ik} is the Markov blanket of z_{ik} in Z . The prior and likelihood terms are respectively those of equations 5.3 and 5.4. Doshi-Velez and Ghahramani (2009) present an accelerated version of Gibbs sampling for this model, that computes the likelihood and prior terms efficiently. We use this approach in our work since it has the benefits of mixing like a collapsed sampler, while maintaining the running time of an uncollapsed sampler.

In conclusion, the optimization steps iteratively refine the parameters V, U, S to be better predictors of the corpus, while Z is updated to best factorize the regularities in the predicate embeddings V , thereby capturing better relational slots. We thus jointly capture both the local and global levels of relational association between predicates and arguments.

5.2.4 Relational Variant

In addition to the standard model introduced above, we also experiment with an extension where the input corpus consists of predicate-argument-relation triples instead of just predicate-argument pairs. These relations are observed relations, and should not be confused with the latent slots of the model.

So, for example, instead of having a simple predicate-argument pair such as “eat, fork” we might have the predicate-argument-relation triple “eat, fork, R-with” indicating that the argument “fork” appears to the right of the predicate, connected by a preposition “with”.

To accommodate this change we modify the argument embedding matrix U to be of dimensions $m \times \frac{d}{2}$ and introduce a new $q \times \frac{d}{2}$ embedding matrix $R = \{r_i \mid 1 \leq i \leq q\}$ for the q observed relation types.

Then, wherever the original model calls for an argument vector u_i (which had dimensionality d) we instead replace it with a concatenated argument-relation vector $[u_i; r_j]$ (which now also has dimensionality d). During training, we must make gradient updates to R in addition to all the other model parameters as usual.

While this relation indicator can be used to capture arbitrary relational information, in this paper we set it to a combination of the directionality of the argument with respect to the predicate (L or R), and the preposition immediately preceding the argument phrase (or *None* if there isn’t one). Thus, for example, we have relational indicators such as “L-on”, “R-before”, “L-because”, “R-None”, etc. We obtain a total of 146 such relations.

Note, that in keeping with the goals of this work, these relation indicators still require no annotation (prepositions are closed-class words than can be enumerated).

5.3 Experiments and Evaluation

The task of evaluating the learned binary representations Z of predicates over latent relational arguments (see Figure 5.2), as well as the vector embeddings for predicates and arguments is a complicated one. The difficulty arises from the fact that there is no single gold standard of what the true set of relations should be.

In what follows, we detail experimental results on two quantitative evaluation tasks: at the local token and global type levels of predicate-argument structure. In particular we evaluate on pseudo disambiguation of selectional preference, and semantic frame lexicon overlap. We also qualitatively inspect the learned latent relations against hand-annotated roles. We first specify the implementational details.

5.3.1 Implementational Details

We begin by pre-training standard skip-gram vectors (Mikolov et al., 2013a) on the NY-Times section of the Gigaword corpus, which consists of approximately 1.67 billion word tokens. These vectors are used as initialization for the embedding matrices V and U , before our iterative optimization. Specifically we use the *target* word space of the skip-gram model to initialize the

predicate embeddings and the *context* word space of the model to initialize the argument embeddings.

While this step is not strictly required, we found that it leads to generally better results than random initialization given the relatively small size of our predicate-argument training corpus. Initialization helps because the two embedding matrices V and U are already somewhat semantically *consistent* with one another before training begins.

Note that in keeping with the aims of this chapter, this pre-training step requires no annotation or access to higher-level linguistic information whatsoever.

For training our models, we use a combination of the training data released for the CoNLL 2008 shared task (Surdeanu et al., 2008) and the extended PropBank release which covers annotations of the Ontonotes (Hovy et al., 2006) and English Web Treebank (Bies et al., 2012) corpora. We reserve the test portion of the CoNLL 2008 shared task data for one of our evaluations.

In this work, we only focus on verbal predicates. Our training data gives us a vocabulary of 4449 predicates, after pruning verbs that occur fewer than 5 times.

Then, from the training data we extract all predicate-argument pairs using gold standard argument annotations, for the sake of simplicity. Note that previous unsupervised frame induction work also uses gold argument mentions (Lang and Lapata, 2011a; Titov and Klementiev, 2012b). Our method, however, does not depend on this, or any other annotation, and we could as easily use the output from an automated system such as Abend et al. (2009) instead. However, to test the efficacy of our approach in a more controlled environment we use gold argument mentions as done in previous work.

In this manner, we obtain a total of approximately 3.35 million predicate-argument word pairs on which to train.

Using this data we train a total of 4 distinct models: a base model and a relational variant (see Section 5.2.4), both of which are trained with two different IBP hyperparameters of $\alpha = 0.35$ and $\alpha = 0.7$. The hyperparameter controls the avidity of the model for latent slots (a higher α implies a greater number of induced slots).

We set our embedding size to $d = 100$. The result of training with these parameters results in the learned number of slots ranging from 17 to 30. The conservative model averaging about 4 latent slots per word, while the permissive model averaging about 6 latent slots per word.

Since our objective is non-convex we record the training likelihood at each power iteration (which includes a single optimization step that alternates over both the predictive and IBP components of our objective), and save the model with the highest training likelihood.

5.3.2 Pseudo Disambiguation of Selection Preference

The pseudo disambiguation task aims to evaluate our models’ ability to capture predicate-argument knowledge at the local level. In this task, systems are presented with a set of triples: a predicate, a true argument and a fake argument. The systems are evaluated on the percentage of true arguments they are able to select.

For example, given a triple:

resign, post, liquidation

a successful model should rate the pair “resign-post” higher than “resign-liquidation”.

Model	α	Variant	k slots	% Acc
Skip-gram	-	-	-	0.77
pa2IBPVec	0.35	Standard	17	0.81
		Relational	15	0.84
	0.7	Standard	27	0.81
		Relational	30	0.81

Table 5.1: Results on pseudo disambiguation of selectional preference. Numbers are in % accuracy of distinguishing true arguments from false ones. Our models all outperform the skip-gram baseline.

This task has often been used in the selectional preference modelling literature as a benchmark task (Rooth et al., 1999; Van de Cruys, 2014).

To obtain the triples for this task we use the test set of the CoNLL 2008 shared task data. In particular, for every verbal predicate mention in the data we select a random nominal word from each of its arguments phrase chunks to obtain a true predicate-argument word pair. Then, to introduce distractors, we sample a random nominal from a unigram noise distribution. In this way we obtain 9859 pseudo disambiguation triples as our test set.

Note that while the term “selectional preference” is used to describe this task, it is different from what is traditionally understood in linguistics as the selectional preference of a predicate for the *head* of an argument.

Rather, it is the selectional preference of a predicate for *any* noun in the governed argument. This setup still works for purposes of our evaluation because the way our models and the baseline (see below) are trained, they should be capable of distinguishing the affinity between the predicate and *any* governed word – not simply the head word.

We use our models to score a word pair by taking the probability of the pair under our model, using the best latent slot:

$$\max_k z_{ik} \sigma((v_i \odot s_k) \cdot u_i) \quad (5.8)$$

where v_i and u_i are predicate and argument embeddings respectively, z_{ik} is the binary indicator of the k 'th slot for the i 'th predicate, and s_k is the slot specific weight vector. The argument in the higher scoring pair is selected as the correct one.

In the relational variant, instead of the single argument vector u_i we also take a max over the relation indicators – since the exact indicator is not observed at test time.

We compare our models against a standard skip-gram model (Mikolov et al., 2013a) trained on the same data. Word pairs in this model are scored using the dot product between their associated skip-gram vectors.

This is a fair comparison since our models as well as the skip-gram model have access to the same data – namely predicates and their neighboring argument words. They are trained on their ability to discriminate true argument words from randomly sampled noise.

The evaluation then, is whether the additionally learned slot structure helps in differentiating true arguments from noise. That is, if the set of induced semantic relations that give our frame lexicon its template structure actually help in defining the local token level of frame knowledge.

Model	α	Variant	Coarse			Fine		
			PU	CO	F1	PU	CO	F1
Syntax	-	-	0.71	0.87	0.78	0.70	0.91	0.79
pa2IBPVec	0.35	Standard	0.76	0.89	0.82	0.76	0.97	0.85
		Relational	0.73	0.90	0.81	0.73	0.97	0.83
	0.7	Standard	0.79	0.91	0.85	0.79	0.98	0.87
		Relational	0.80	0.92	0.85	0.80	0.98	0.88

Table 5.2: Results on the lexicon overlap task. Our models outperform the syntactic baseline on all the metrics.

The results of this evaluation are presented in Table 5.1.

The results show that all our models outperform the skip-gram baseline. This demonstrates that the added structural information gained from latent slots in fact help our models to better capture predicate-argument affinities in local contexts.

The impact of latent slots or additional relation information does not seem to impact basic performance, however. Both the addition of the relational indicator and the increase in the hyperparameter of the model α do not produce any significant difference in the overall score.

This could be because of the trade-off that occurs when a more complex model is learned from the same amount of limited data.

5.3.3 Frame Lexicon Overlap

Next, we evaluate our models at their ability to capture global type predicate-argument structure. Previous work on frame induction has focussed on evaluating instance-based argument overlap with gold standard annotations in the context of semantic role labelling (SRL). Unfortunately, because our models operate on individual predicate-argument words rather than argument spans, a fair comparison becomes problematic.

But unlike previous work, which clusters argument instances, our approach produces a model as a result of training. Specifically we produce the latent binary matrix Z (see Figure 5.2), which captures the presence or absence of individual relational arguments for each predicate in our vocabulary.

A hand-crafted lexicon such as PropBank gives us very similar information. It defines a set of (hand-crafted) relational arguments and then tells us for every predicate in the vocabulary, which of the relational arguments are accepted by that predicate.

We can thus directly evaluate our models’ latent slot factors against a gold standard frame lexicon. Our evaluation framework is, in many ways based on the metrics used in unsupervised SRL, except applied at the “type” lexicon level rather than the corpus-based “token” cluster level.

In particular, given a gold frame lexicon Ω with K^* real argument slots (i.e. the total number of possible humanly assigned arguments in the lexicon), we evaluate our models’ latent slot matrix Z in terms of its overlap with the gold lexicon.

Intuitively, consider Ω to be a different binary matrix than the one pictured for Z in Figure 5.2. This different matrix has the same number of rows (we consider the intersection of the

vocabularies of our model and the gold standard, in practise), but may have a different number of columns.

The columns in each of the matrices can be thought of as different, possibly overlapping clusters over predicates. They effectively define which relations (gold or induced) are accepted by which predicates. The idea, then is to measure, the degree of overlap between the gold clusters and the IBP induced ones.

To do this, we use the cluster *purity* and *collocation* metrics often used to evaluate clustering quality.

We define *purity* as the average proportion of overlap between predicted latent slots and their maximally similar gold lexicon slots:

$$PU = \frac{1}{K} \sum_k \max_{k'} \frac{1}{n} \sum_i \delta(\omega_{ik'}, z_{ik}) \quad (5.9)$$

where $\delta(\cdot)$ is an indicator function. Given that the ω 's and z 's we compare are binary values, this indicator function is effectively an "XNOR" gate.

Intuitively, the purity metric takes every column in the induced matrix Z , looks for it's most similar column in the gold matrix Ω and measures the overlap between the two. Similarity is itself defined as the sum of identical terms (essentially overlap) between the two column vectors under consideration. These overlap scores are then averaged over all the columns in Z .

Similarly we define *collocation* as the average proportion of overlap between gold standard slots and their maximally similar predicted latent slots:

$$CO = \frac{1}{K^*} \sum_{k'} \max_k \frac{1}{n} \sum_i \delta(\omega_{ik'}, z_{ik}) \quad (5.10)$$

Intuitively, the collocation metric is the same as purity, except applied in the reverse direction. That is, we take every column in the gold matrix Ω , look for it's most similar column in the induced matrix Z and measure the overlap between the two. The overlap scores are then averaged over all the columns in Ω .

Note that the purity and collocation metrics are different because the number of columns in Z and Ω are (likely) different, as well as the fact that the mapping between maximally similar columns is not one-to-one.

Given, the *purity* and *collocation* metrics we can also define the $F1$ score as the harmonic mean of the two:

$$F1 = \frac{2 \cdot CO \cdot PU}{CO + PU} \quad (5.11)$$

In our experiments we use the frame files provided with the PropBank corpus (Palmer et al., 2005) as gold standard. We derive two variants from the frame files.

The first is a coarse-grained lexicon. In this case, we extract only the functional arguments of verbs in our vocabulary as gold standard slots. There are 16 such gold slots and include types such as "prototypical agent", "prototypical patient", "instrument", "benefactive", etc. Each of these 16 gold slots are mapped to indices. For every verb the corresponding binary ω vector

marks the existence or not of the different functional arguments according to the gold frame files.

The second variant is a fine-grained lexicon. Here, in addition to functional arguments we also consider the numerical argument with which it is associated, such as “ARG0”, “ARG1” etc. So for example we might have slots such as “prototypical agent ARG0”, “prototypical patient ARG1” etc. Note that a single functional argument may appear with more than one numerical slot with different verbs over the entire lexicon. So, for example, we may additionally also have the “prototypical agent ARG1” slot type. The fine-grained lexicon yields 72 gold slots.

We compare our models against a baseline inspired from the syntactic baseline often used for evaluating unsupervised SRL models. For unsupervised SRL, syntax has proven to be a difficult-to-outperform baseline (Lang and Lapata, 2014).

This baseline is constructed by taking the 21 most frequent syntactic labels in the training data and associating them each with a slot. All other syntactic labels are associated with a 22nd generic slot. Given these slots, we associate a verbal predicate with a specific slot if it takes on the corresponding syntactic argument in the training data.

This process also yields a binary matrix with rows that correspond to predicates and columns that correspond to 22 syntactic slots. We can then evaluate this binary matrix against the gold standard Ω in the same way as our own matrices on the purity and collocation metrics described above.

The results on the lexicon overlap task are presented in Table 5.2.

They show that our models consistently outperform the syntactic baseline on all metrics in both the coarse-grained and fine-grained settings. We conclude that our models are better able to capture predicate-argument structure at a global type level.

Specifically, the semantic relations we induce to govern the template for our frame lexicon yields more semantically meaningful slots than those obtained by using syntax as a surrogate for semantics.

Inspecting and comparing the results of our different models seems to indicate that we perform better when our IBP posterior allows for a greater number of latent slots. This happens when the hyperparameter $\alpha = 0.7$.

Additionally our models consistently perform better on the fine-grained lexicon than on the coarse-grained one. The former itself does not necessarily represent an easier benchmark, since there is hardly any difference in the $F1$ score of the syntactic baseline on the two lexicons.

Overall it would seem that allowing for a greater number of latent slots does help capture global predicate-argument structure better. This makes sense, if we consider the fact that we are effectively trying to factorize a dense representation (the predicate embeddings) with IBP inference. Thus allowing for a greater number of latent factors permits the discovery of greater structural consistency within these embeddings.

This finding does have some problematic implications, however. Increasing the IBP hyperparameter α arbitrarily represents a computational bottleneck since inference scales quadratically with the number of latent slots K . There is also the problem of splitting argument slots too finely, which may result in optimizing purity at the expense of collocation. A solution to this trade-off between performance and inference time remains for future work.

Predicate	Latent Slot							
	1	2	3	5	6	8	10	12
provide	A0			A1	A2			A2
enter	A0	A1						AM-ADV
praise	A0	A1				A2		
travel	A0	A0				AM-PNC	AM-TMP	
distract	A0		A1	A2				
overcome	AM-TMP		A0	A0				

Table 5.3: Examples for several predicates with mappings of latent slots to the majority class of the closest argument vector in the shared embedded space.

5.3.4 Qualitative Analysis of Latent Slots

To better understand the nature of the latent slots induced by our model we conduct an additional qualitative analysis. The goal of this analysis is to inspect the kinds of generalities about semantic roles that our model is able to capture from completely unannotated data.

Table 5.3 lists some examples of predicates and their associated latent slots. The latent slots are sorted according to their frequency (i.e. column sum in the binary slot matrix Z). We map each latent slot to the majority semantic role type – from training data – of the closest argument word to the predicate vector in the shared embedding space.

The model for which we perform this qualitative analysis is the standard variant with the IBP hyperparameter set to $\alpha = 0.35$; this model has 17 latent slots. Note that slots that do not feature for any of the verbs in the table are omitted for visual compactness.

There are several interesting trends to notice here. Firstly, the basic argument structure of predicates is often correctly identified, when matched against gold PropBank frame files. For example, the core roles of “enter” identify it as a transitive verb, while “praise”, “provide” and “distract” are correctly shown as di-transitive verbs. Obviously the structure isn’t always perfectly identified, as with the verb “travel” where we are missing both an “ARG1” and an “ARG2”.

In certain cases a single argument type spans multiple slots – as with “A2” for “provide” and “A0” for “travel”. This is not surprising, since there is no binding factor on the model to produce one-to-one mappings with hand-crafted semantic roles. Generally speaking, the slots represent distributions over hand-crafted roles rather than strict mappings. In fact, to expect a one-to-one mapping is unreasonable considering we use no annotations whatsoever.

Nevertheless, there is still some consistency in the mappings. The core arguments of verbs – such as “ARG0” and “ARG1” are typically mapped to the most frequent latent slots. This can be explained by the fact that the more frequent arguments tend to be the ones that are core to a predicate’s frame. This is quite a surprising outcome of the model, considering that it is given no annotation about argument types. Of course, we do not always get this right as can be seen with the case of “overcome”, where a non-core argument occurs in the most frequent slot.

Since this is a data driven approach, we identify non-core roles as well, if they occur with predicates often enough in the data. For example we have the general purpose “AM-ADV” argument of “enter”, and the “ARG-PNC” and “ARG-TMP” (purpose and time arguments) of the verb “travel”. In future work we hope to explore methods that might be able to automatically

distinguish core slots from non-core ones.

In conclusion, our models show promise in that they are able to capture some interesting generalities with respect to predicates and their hand-crafted roles, without the need for any annotated data.

5.4 Conclusion and Future Work

In this chapter, we have focussed on a general problem of semantic representation learning where we *induced* a new set of relations rather than simply *leverage* an existing set of relations.

Specifically we have presented a first attempt at learning an embedded frame lexicon from data, using minimal annotated information. Our approach revolved around jointly capturing local predicate-argument affinities with global slot-level consistencies. We modeled this approach with a novel integration between a predictive embedding model and the posterior of an Indian Buffet Process.

The result of this process is a fully embedded semantic lexicon that is able to compute expected affinities between predicates and arguments. Crucially, the template structure of this lexicon – that is, the set of relations that define the frames therein – are induced automatically from data.

We experimented with our model on two quantitative tasks, each designed to evaluate performance on capturing local token and global type predicate-argument structure respectively. On both tasks we demonstrated that our models are able to outperform baselines, thus indicating our ability to jointly model the local and global level information of predicate-argument structure.

Additionally, we qualitatively inspected our induced latent slots and showed that we are able to capture some interesting generalities with respect to hand-crafted semantic role labels.

In the context of this thesis, this is an important contribution because we demonstrated that we can learn rich models of semantic representation, not only when we have a set of well-defined relations to work with, but also now in the more ambiguous case when relational information does not exist or annotated data with relations is not easily accessible.

Chapter 6

Structured Distributional Semantics

Syntactic and Latent Semantic Tensor Models for Non-constant and Non-unique Representation Learning

Most current methods for lexical representation learning yield a vector space model (Turney and Pantel, 2010) in which words are represented as distinct, discrete and static points in some high dimensional space. The most fundamental operation in such semantic vector spaces is comparison of meaning, in which the distance of two points is measured according to a distance metric.

However, the meaning of a word is neither constant nor unique.

Meaning depends on the context in which it occurs; in other words it depends on the relations that bind it to its compositional neighborhood. Consider the following fragment containing the predicate “killed”:

- Example: [**Ag** Booth] [**Pred** killed] [**Pat** Lincoln]

In this fragment, relational attachment matters and specifies that Booth was the one doing the killing while Lincoln was the sufferer of these actions. It is not equivalent to the fragment “Lincoln killed Booth”.

However, in a vector space model the word “killed” is learned beforehand. It is fixed to a single point in vector space, typically capturing the bag-of-words statistics of “killed”. It does not have the power to change or accommodate an *Agent* or *Patient* argument, for example.

Meaning is also not unique. It depends on what it’s compared to for proximity of meaning, and more specifically along what relational facet the comparison is being made. Take the following disambiguation triple:

- Example: shoe → tuxedo, football

Is the word “shoe” more similar to “tuxedo” or “football”? The answer, in fact, really depends on the *facet* along which we are making the comparison. Specifically, if we wish to know similarity along the *article_of_clothing* facet, we would say “shoe” and “tuxedo” are more sim-

ilar. Similarly, if what we cared about instead was the *sports* facet, we would say “shoe” and “football” were more similar.

Again, a single point in vector space cannot typically be de-composed into its relational facets to capture finer grained semantics along specific dimensions of interest.

These two problems, context-sensitivity and facetedness in semantics, are the representation problems that we tackle in this chapter. We introduce two different models that tackle one of the two problems, each.

However, both problems are defined by semantic relations: the one to neighborhood structure, the other to comparative concepts. Thus the solutions that we present are also very related. Their approaches fall into an umbrella paradigm we call *Structured Distributional Semantics*. This paradigm is similar to the work on *Distributional Memory* proposed by Baroni and Lenci (2010).

Structured Distributional Semantics aims to improve upon simple vector space models of semantics by hypothesizing that the meaning of a word is captured more effectively through its relational — rather than its raw distributional — signature.

In accordance, it extends the vector space paradigm by structuring elements with relational information that decompose distributional signatures over discrete relation dimensions. The resulting representation for a word is a semantic tensor (or matrix) rather than a semantic vector.

The easiest way to think of these representations are to associate them with semantic frames. Each word is represented by a frame tensor, with each frame element being captured by a specific row in the learned semantic matrix.

The move from simple Vector Space Models (VSMs) to Structured Distributional Semantic Models (SDSM) represents a perspective on semantic representation learning that aligns with the theme of this thesis. Specifically, that relations drive and give meaning to concepts. This view is not new and has gained much popularity with the field of frame semantics (Baker et al., 1998).

In this chapter we will show that decomposing a single word vector into distinct relation dimensions results in more expressive and empirically superior models for several tasks.

An important choice, with an SDSM is the nature of the discrete relation dimensions, or the frame elements of the semantic tensors – in practise the rows of the matrix. Nominally, this choice not only determines the process by which an SDSM is learned, but also what information is represented and the resulting strengths and weaknesses of the model. Our two models differ in this choice.

The first model is a Syntactic SDSM that represents meaning as distributions over relations in syntactic neighborhoods. This model tackles the problem of contextual sensitivity, with a dynamic SDSM that constructs semantic tensors for words and their relational arguments. We argue that our model approximates meaning in compositional configurations more effectively than standard distributional vectors or bag-of-words models. We test our hypothesis on the problem of judging event coreferentiality, which involves compositional interactions in the predicate-argument structure of sentences, and demonstrate that our model outperforms both state-of-the-art window-based word embeddings as well as simple approaches to compositional semantics previously shown in the literature.

The second model is a Latent Semantic SDSM whose relational dimensions are learned over neighborhood patterns using Latent Relational Analysis (LRA). In comparison to the dynamic SDSM of syntax, this model is static due to its latent nature and therefore does not model compositional structures. However, it frees the model from its reliance on syntax (or any other set

of pre-defined relations for that matter), and allows the data to dictate an optimal set of (albeit latent) relations. The result is a model that is much more effective at representing single word units and tackling the problem of facetedness in the comparison of relational facets. Evaluation of our model yields results that significantly outperform several other distributional approaches (including the Syntactic SDSM) on two semantic tasks and performs competitively on a third relation classification task.

Through both models we re-affirm the central claim of this thesis, namely that by integrating structured relational information, we can learn more expressive and empirically superior models of semantics.

6.1 Distributional Versus Structured Distributional Semantics

We begin by formalizing the notion of a Distributional Semantic Model (DSM) as a vector space, and its connections and extension to a tensor space in a Structured Distributional Semantic Model (SDSM).

A DSM is a vector space V that contains $|\Sigma|$ elements in \mathbb{R}^n , where $\Sigma = \{w_1, w_2, \dots, w_k\}$ is a vocabulary of k distinct words. Every vocabulary word w_i has an associated semantic vector \vec{v}_i representing its distributional signature. Each of the n elements of \vec{v}_i is associated with a single dimension of its distribution. This dimension may correspond to another word — that may or may not belong to Σ — or a latent dimension as might be obtained from an SVD projection or an embedding learned via a deep neural network. Additionally, each element in \vec{v}_i is typically a normalized co-occurrence frequency count, a PMI score, or a number obtained from an SVD or RNN transformation. The semantic similarity between two words w_i and w_j in a DSM is the vector distance defined by $\cos(\vec{v}_i, \vec{v}_j)$ on their associated distributional vectors.

An SDSM is an extension of DSM. Formally, it is a space U that contains $|\Sigma|$ elements in $\mathbb{R}^{d \times n}$, where $\Sigma = \{w_1, w_2, \dots, w_k\}$ is a vocabulary of k distinct words. Every vocabulary word w_i has an associated semantic tensor \vec{u}_i , which is itself composed of d vectors $\vec{u}_i^1, \vec{u}_i^2, \dots, \vec{u}_i^d$ each having n dimensions. Every vector $\vec{u}_i^l \in \vec{u}_i$ represents the distributional signature of the word w_i in a relation (or along a facet) r_l . The d relations of the SDSM may be syntactic, semantic, or latent. The n dimensional relational vector \vec{u}_i^l is configurationally the same as a vector \vec{v}_i of a DSM. This definition of an SDSM closely relates to an alternate view of Distributional Memories (DMs) (Baroni and Lenci, 2010) where the semantic space is a third-order tensor, whose modes are Word \times Link \times Word.

The semantic similarity between two words w_i and w_j in an SDSM is the similarity function defined by $\text{sim}(\vec{u}_i, \vec{u}_j)$ on their associated semantic tensors. We use the following decomposition of the similarity function:

$$\text{sim}(\vec{u}_i, \vec{u}_j) = \frac{1}{d} \sum_{l=1}^d \cos(\vec{u}_i^l, \vec{u}_j^l) \quad (6.1)$$

Mathematically, this corresponds to the ratio of the normalized Frobenius product of the two

matrices representing \vec{u}_i and \vec{u}_j to the number of rows in both matrices. Intuitively it is simply the average relation-wise similarity between the two words w_i and w_j .

6.2 Syntactic Structured Distributional Semantics

We begin by describing our first SDSM, which tackles the problem of contextual variability in semantics by a representation over syntactic relational dimensions.

Systems that use window-based DSMs implicitly make a bag of words assumption: that the meaning of a phrase can be reasonably estimated from the meaning of its constituents, agnostic of ordering. However, semantics in natural language is a compositional phenomenon, encompassing interactions between syntactic structures, and the meaning of lexical constituents.

It follows that the DSM formalism lends itself poorly to composition since it implicitly disregards syntactic structure. For instance, returning to the example presented in the introduction to this chapter, the representation for “Lincoln”, “Booth”, and “killed” when merged produce the same result regardless of whether the input is “Booth killed Lincoln” or “Lincoln killed Booth”. As suggested by Pantel and Lin (2000) and others, modeling the distribution over preferential attachments for each syntactic relation separately can yield greater expressive power.

Attempts have been made to model linguistic composition of individual word vectors (Mitchell and Lapata, 2009), as well as remedy the inherent failings of the standard distributional approach (Erk and Padó, 2008). The results show varying degrees of efficacy, and have been only partially successful in modelling deeper lexical semantics or compositional expectations of words and word combinations.

The Syntactic SDSM we propose is an extension of traditional DSMs. This extension explicitly preserves structural information and permits the approximation of distributional expectation over distinct dependency relations. It is also a dynamic model, meaning that statistics are stored in decomposed form as dependency triples, only to be composed to form representations for arbitrary compositional structures as and when desired.

This is in contrast to a standard VSM which learns its representations once and is then used in fixed form for all downstream semantic computations.

In the new model the tensor representing a word is a distribution over relation-specific syntactic neighborhoods. In other words, the SDSM representation of a word or phrase is several vectors defined over the same vocabulary, each vector representing the word or phrase’s selectional preferences for a different syntactic argument.

We show that this representation is comparable to a single-vector window-based DSM at capturing the semantics of individual word, but significantly better at representing the semantics of composed units. Our experiments on individual word semantics are conducted on the two tasks of similarity scoring and synonym selection. For evaluating compositional units, we turn to the problem of event coreference and the representation of predicate-argument structures. We use two different event coreference corpora and show that our Syntactic SDSM achieves greater predictive accuracy than simplistic compositional approaches as well as a strong baseline that uses window based word embeddings trained via deep neural-networks.

6.2.1 Related Work

Recently, a number of studies have tried to model a stronger form of semantics by phrasing the problem of DSM compositionality as one of vector composition. These techniques derive the meaning of the combination of two words a and b by a single vector $c = f(a, b)$. Mitchell and Lapata (2008) propose a framework to define the composition $c = f(a, b, r, K)$ where r is the relation between a and b , and K is some additional knowledge used to define composition.

While the framework is quite general, most models in the literature tend to disregard K and r and are generally restricted to component-wise addition and multiplication on the vectors to be composed, with slight variations. Dinu and Lapata (2010) and Séaghdha and Korhonen (2011) introduced a probabilistic model to represent word meanings by a latent variable model. Subsequently, other high-dimensional extensions by Rudolph and Giesbrecht (2010), Baroni and Zamparelli (2010) and Grefenstette et al. (2011), regression models by Guevara (2010), and recursive neural network based solutions by Socher et al. (2012) and Collobert et al. (2011) have been proposed.

Pantel and Lin (2000) and Erk and Padó (2008) attempted to include syntactic context in distributional models. However, their approaches do not explicitly construct phrase-level meaning from words which limits their applicability to modelling compositional structures. A quasi-compositional approach was also attempted in Thater et al. (2010) by a systematic combination of first and second order context vectors. Perhaps the most similar work to our own is Baroni and Lenci (2010) who propose a *Distributional Memory* that stores syntactic contexts to dynamically construct semantic representations. However, to the best of our knowledge the formulation of composition we propose is the first to account for K and r within the general framework of composition $c = f(a, b, r, K)$, as proposed by Mitchell and Lapata (2008).

6.2.2 The Model

In this section, we describe our Structured Distributional Semantic framework in detail. We first build a large knowledge base from sample english texts and use it to represent basic lexical units. Next, we describe a technique to obtain the representation for larger units by composing their constituents.

6.2.2.1 The PropStore

To build a lexicon of SDSM representations for a given vocabulary we construct a proposition knowledge base (the PropStore) by processing the text of Simple English Wikipedia through a dependency parser. Dependency arcs are stored as 3-tuples of the form $\langle w_1, r, w_2 \rangle$, denoting occurrences of words w_1 and word w_2 related by the syntactic dependency relation r . We also store sentence identifiers for each triple for reasons described later. In addition to the words' surface-forms, the PropStore also stores their POS tags, lemmas, and WordNet super-senses.

The PropStore can be used to query for preferred expectations of words, super-senses, relations, etc., around a given word. In the example in Figure 6.1, the query $(SST(W_1) = \text{verb.consumption}, ?, \text{dobj})$ i.e., “what is consumed”, might return expectations [pasta:1, spaghetti:1, mice:1 ...]. In our implementation, the relations and POS tags are obtained using the Fanseparser (Tratz

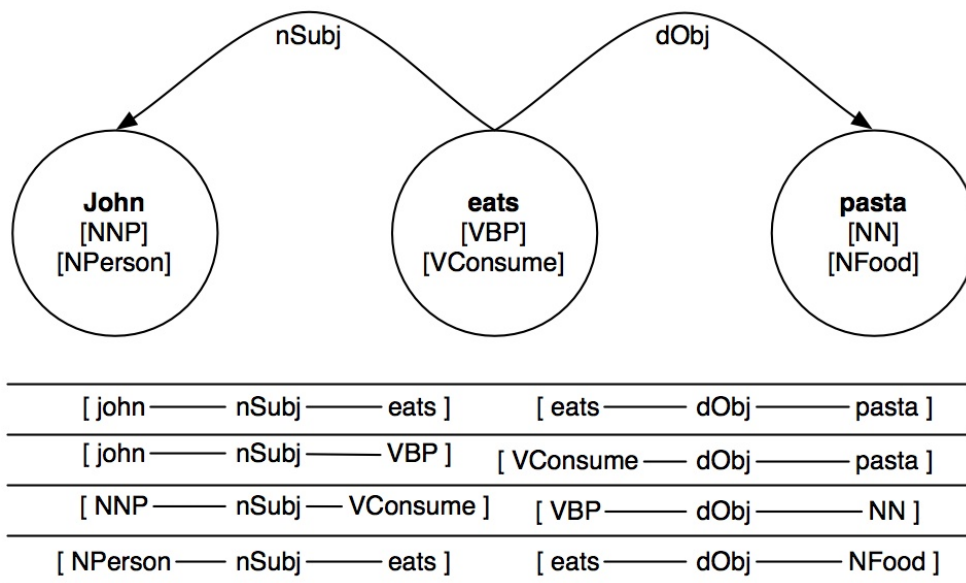


Figure 6.1: A sample parse fragment and some examples of triples that we extract from it. Triples are generated from each dependency arc by taking the cross-product of the annotations that the nodes connected to the arc contain.

and Hovy, 2011), super-sense tags using sst-light (Ciaramita and Altun, 2006), and lemmas are obtained from WordNet (Miller, 1995).

6.2.2.2 Building the Representation

Next, we describe a method to represent lexical entries as structured distributional matrices using the PropStore.

Recall from Section 6.1, that the canonical form of a concept w_i (word, phrase, etc.) in the SDSM framework is a matrix \vec{u}_i . In the Syntactic SDSM the elements u_i^{lj} of this matrix is the cardinality of the sentence identifiers obtained by querying the PropStore for contexts in which w_i appears in the syntactic neighborhood of the w_j linked by the dependency relation r_l . This cardinality is thus the content of every cell of the representational matrix and denotes the frequency of co-occurrence of a concept and word under the given relational constraint.

This canonical matrix form can be interpreted in several different ways. Each interpretation is based on a different normalization scheme.

1. **Row Norm:** each row of the matrix is interpreted as a distribution over words that attach to the target concept with the given dependency relation. The elements of the matrix become:

$$u_i^{lj} = \frac{u_i^{lj}}{\sum_{j' \in \vec{u}_i^l} u_i^{lj'}} \quad \forall i, j, l$$

2. **Full Norm:** The entire matrix is interpreted as a distribution over the word-relation pairs

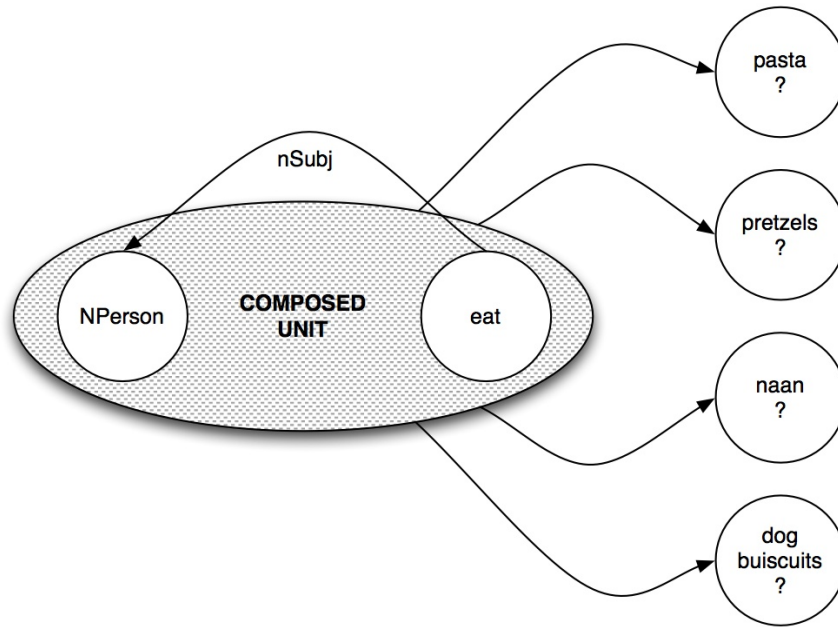


Figure 6.2: A depiction of the composition of two words. After composition the joint unit is treated as a single node, and the expectations of that combined node are then computed from the PropStore.

which can attach to the target concept. The elements of the matrix become:

$$u_i^{lj} = \frac{u_i^{lj}}{\sum_{l', j' \in \vec{u}_i} u_i^{l'j'}} \quad \forall i, j, l$$

3. **Collapsed Vector Norm:** The columns of the matrix are collapsed to form a standard normalized distributional vector trained on dependency relations rather than sliding windows. The elements of the matrix become:

$$u_i^j = \frac{\sum_{l', j' \in \vec{u}_i} u_i^{l'j'} \delta(j = j')}{\sum_{l', j' \in \vec{u}_i} u_i^{l'j'}} \quad \forall i, j, l$$

where δ is the indicator function.

6.2.3 Mimicking Compositionality

For representing intermediate multi-word phrases, we extend the above word-relation matrix symbolism in a bottom-up fashion. The combination hinges on the intuition that when lexical units combine to form a larger syntactically connected phrase, the representation of the phrase is given by its own distributional neighborhood within the embedded parse tree. The distributional

neighborhood of the net phrase can be computed using the PropStore given syntactic relations anchored on its parts. For the example in Figure 6.1, we can compose $SST(w_1) = \text{Noun.person}$ and $\text{Lemma}(w_2) = \text{eat}$ with relation ‘nsubj’ to obtain expectations around “people eat” yielding [pasta:1, spaghetti:1 . . .] for the *object* relation ([dining room:2, restaurant:1 . . .] for the *location* relation, etc.) (See Figure 6.2). Larger phrasal queries can be built to answer questions like “What do people in China eat with?”, “What do cows do?”, etc. All of this helps us to account for both relation r and knowledge K obtained from the PropStore within the compositional framework $c = f(a, b, r, K)$.

The general outline to obtain a composition of two words is given in Algorithm 3. Here, we first determine the sentence indices where the two words w_1 and w_2 occur with relation r_l . Then, we return the expectations around the two words within these sentences. Note that the entire algorithm can conveniently be written in the form of database queries to our PropStore.

Algorithm 3 ComposePair(w_1, r_l, w_2)

```

 $\vec{u}_1 \leftarrow \text{queryMatrix}(w_1)$ 
 $\vec{u}_2 \leftarrow \text{queryMatrix}(w_2)$ 
SentIDs  $\leftarrow \vec{u}_1^l \cap \vec{u}_2^l$ 
return  $((\vec{u}_1 \cap \text{SentIDs}) \cup (\vec{u}_2 \cap \text{SentIDs}))$ 

```

Similar to the two-word composition process, given a parse subtree T of a phrase, we can obtain its matrix representation of empirical counts over word-relation contexts. Let $E = \{e_1 \dots e_n\}$ be the set of edges in T , and specifically $e_i = (w_i^1, r_{l_i}, w_i^2) \forall i = 1 \dots n$. The procedure for obtaining the compositional representation of T is given in Algorithm 4.

Algorithm 4 ComposePhrase(T)

```

SentIDs  $\leftarrow$  All Sentences in corpus
for  $i = 1 \rightarrow n$  do
   $\vec{u}_i^1 \leftarrow \text{queryMatrix}(w_i^1)$ 
   $\vec{u}_i^2 \leftarrow \text{queryMatrix}(w_i^2)$ 
  SentIDs  $\leftarrow$  SentIDs  $\cap (\vec{u}_i^{l_i^1} \cap \vec{u}_i^{l_i^2})$ 
end for
return  $((\vec{u}_1 \cap \text{SentIDs}) \cup (\vec{u}_2 \cap \text{SentIDs}) \dots \cup (\vec{u}_n^1 \cap \text{SentIDs}) \cup (\vec{u}_n^2 \cap \text{SentIDs}))$ 

```

6.2.3.1 Tackling Sparsity

The SDSM model reflects syntactic properties of language through preferential filler constraints. But by distributing counts over a set of relations the resultant SDSM representation is comparatively much sparser than the DSM representation for the same word. In this section we present some ways to address this problem.

Sparse Back-off

The first technique to tackle sparsity is to back off to progressively more general levels of linguistic granularity when sparse matrix representations for words or compositional units are encountered or when the word or unit is not in the lexicon. For example, the composition “Balthazar eats” cannot be directly computed if the named entity “Balthazar” does not occur in the PropStore’s knowledge base. In this case, a query for a super-sense substitute – “Noun.person eat” – can be issued instead. When super-senses themselves fail to provide numerically significant counts for words or word combinations, a second back-off step involves querying for POS tags. With coarser levels of linguistic representation, the expressive power of the distributions becomes diluted. But this is often necessary to handle rare words. Note that this is an issue with DSMs too.

Densification

In addition to the back-off method, we also propose a secondary method for “densifying” distributions. A concept’s distribution is modified by using words encountered in its syntactic neighborhood to infer counts for other semantically similar words. In other terms, given the matrix representation of a concept, densification seeks to populate its null columns (which each represent a word-dimension in the structured distributional context) with values weighted by their scaled similarities to words (or effectively word-dimensions) that actually occur in the syntactic neighborhood.

For example, suppose the word “play” had an “nsubj” preferential vector that contained the following counts: [cat:4 ; Jane:2]. One might then populate the column for “dog” in this vector with a count proportional to its similarity to the word cat (say 0.8), thus resulting in the vector [cat:4 ; Jane:2 ; dog:3.2]. These counts could just as well be probability values or PMI associations (suitably normalized). In this manner, the k most similar word-dimensions can be densified for each word that actually occurs in a syntactic context. As with sparse back-off, there is an inherent trade-off between the degree of densification k and the expressive power of the resulting representation.

Dimensionality Reduction

The final method tackles the problem of sparsity by reducing the representation to a dense low-dimensional word embedding using singular value decomposition (SVD). In a typical term-document matrix, SVD finds a low-dimensional approximation of the original matrix where columns become latent concepts while similarity structure between rows are preserved. The PropStore, as described in Section 6.2.2.1, is an order-3 tensor with w_1 , w_2 and rel as its three axes. We explore the following two possibilities to perform dimensionality reduction using SVD.

Word-word matrix SVD: In this method, we preserve the axes w_1 and w_2 and ignore the relational information. Following the SVD regime ($W = U\Sigma V^T$) where Σ is a square diagonal matrix of the k largest singular values, and U and V are matrices with dimensionality $m \times k$ and $n \times k$ respectively. We adopt matrix U as the compacted concept representation.

Tensor SVD: To remedy the relation-agnostic nature of the word-word SVD matrix representation, we use tensor SVD (Vasilescu and Terzopoulos, 2002) to preserve relational structure

information. The mode- n vectors of an order- N tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ are the I_n -dimensional vectors obtained from \mathcal{A} by varying index i_n while keeping other indices fixed. The matrix formed by all the mode- n vectors is a mode- n flattening of the tensor. To obtain the compact representations of concepts we thus first apply mode w_1 flattening and then perform SVD on the resulting tensor.

6.2.4 Single Word Evaluation

In this section we describe experiments and results for judging the expressive power of the structured distributional representation for individual words. We use a similarity scoring task and a synonym selection task for the purpose of this evaluation. We compare the SDSM representation to standard window-based distributional vectors trained on the same corpus (Simple English Wikipedia). We also experiment with different normalization techniques outlined in Section 3.2, which effectively lead to structured distributional representations with distinct interpretations.

We experimented with various similarity metrics and found that the normalized city-block distance metric provides the most stable results.

$$\begin{aligned} \text{City-block}(X, Y) &= \frac{\text{ArcTan}(d(X, Y))}{d(X, Y)} \\ d(X, Y) &= \frac{1}{|R|} \sum_{r \in R} d(X_r, Y_r) \end{aligned}$$

Results in the rest of this section are thus reported using the normalized city-block metric. We also report experimental results for the two methods of alleviating sparsity discussed in Section 3.4, namely, densification and SVD.

6.2.4.1 Similarity Scoring

On this task, the different semantic representations are used to compute similarity scores between two (out of context) words. We used the dataset from Finkelstein et al. (2002) for our experiments. It consists of 353 pairs of words along with an averaged similarity score on a scale of 1.0 to 10.0 obtained from 13–16 human judges.

For example, an item in the dataset is “book, paper \rightarrow 7.46”. Systems are evaluated on the degree of correlation between the similarity scores they assign to pairs of words and those obtained from the gold standard reference scores.

6.2.4.2 Synonym Selection

In the second task, the same set of semantic representations are used to produce a similarity ranking on the Turney (2002) ESL dataset. This dataset comprises 50 words that appear in a context (we discarded the context in this experiment), along with 4 candidate lexical substitutions. Only one of the four candidates is an apt substitute.

For example, the dataset contains items of the form “rug \rightarrow sofa, ottoman, carpet, hallway” (with carpet being the correct synonym). We evaluate the semantic representations on the basis of their ability to discriminate the top-ranked candidate.

Model	Finklestein (ρ Correlation)	ESL (% Accuracy)
DSM	0.283	24.7
Collapsed	0.260	17.8
Full Norm	0.282	19.2
Row Norm	0.236	26.4
Densified Row Norm	0.259	26.7

Table 6.1: Single word evaluations on similarity scoring and synonym selections. SDSM does not clearly outperform DSM, likely due to issues of sparsity.

Model	Finklestein (ρ Correlation)
DSM	0.283
Matrix SVD 100	0.207
Matrix SVD 500	0.221
Tensor SVD 100	0.267
Tensor SVD 500	0.315

Table 6.2: Effects of SVD methods on SDSM representations. With higher order tensor SVD, SDSM outperforms DSM baseline.

6.2.4.3 Results and Discussion

Table 6.1 summarizes the results for the window-based baseline and each of the structured distributional representations on both tasks. It shows that our representations for single words are competitive with window based distributional vectors. Densification in certain conditions improves our results, but no consistent pattern is discernible. This can be attributed to the trade-off between the gain from generalization and the noise introduced by surrogate neighboring words.

We also explore dimensionality reduction as an additional means of reducing sparsity. Table 6.2 gives correlation scores on the Finkelstein et al. (2002) dataset when SVD is performed on the representations, as described in Section 6.2.3.1. We give results when 100 and 500 principal components are preserved for both matrix as well as tensor SVD techniques.

These experiments suggest that though afflicted by sparsity, the proposed Structured Distributional paradigm is competitive with window-based distributional vectors. In the following sections we show that that the framework provides considerably greater power for modeling composition when dealing with units consisting of more than one word.

6.2.5 Event Coreference Judgment

Given the SDSM formulation and assuming no sparsity constraints, it is possible to calculate SDSM matrices for composed concepts. However, are these correct? Intuitively, if they truly capture semantics, the two SDSM matrix representations for “Booth assassinated Lincoln” and “Booth shot Lincoln with a gun” should be (almost) the same. To test this hypothesis we turn to the task of predicting whether two event mentions are coreferent or not, even if their surface forms differ.

While automated resolution of entity coreference has been an actively researched area (Haghighi and Klein, 2009; Stoyanov et al., 2009; Raghunathan et al., 2010), there has been relatively little work on event coreference resolution. Lee et al. (2012) perform joint cross-document entity and event coreference resolution using the two-way feedback between events and their arguments.

In this paper, however, we only consider coreferentiality between pairs of events. Formally, two event mentions generally refer to the same event when their respective actions, agents, patients, locations, and times are (almost) the same. Given the non-compositional nature of determining equality of locations and times, we represent each event mention by a triple $\mathbf{E} = (e, a, p)$ for the event, agent, and patient.

While linguistic theory of argument realization is a debated research area (Levin and Rapaport Hovav, 2005; Goldberg, 2005), it is commonly believed that event structure (Moens and Steedman, 1988) centralizes on the predicate, which governs and selects its role arguments (Jackendoff, 1987). In the corpora we use for our experiments, most event mentions are verbs. However, when nominalized events are encountered, we replace them by their verbal forms. We use Semantic Role Labelling (SRL) Collobert et al. (2011) to determine the agent and patient arguments of an event mention. When SRL fails to determine either role, its empirical substitutes are obtained by querying the PropStore for the most likely word expectations for the role. The triple (e, a, p) is the composition of the triples (a, rel_{agent}, e) and $(p, rel_{patient}, e)$, and hence a complex object. To determine equality of this complex composed representation we generate three levels of progressively simplified event constituents for comparison:

- **Level 1: Full Composition:**

$$M_{full} = ComposePhrase(e, a, p).$$

- **Level 2: Partial Composition:**

$$M_{part:EA} = ComposePair(e, r_{ag}, a)$$

$$M_{part:EP} = ComposePair(e, r_{pat}, p).$$

- **Level 3: No Composition:**

$$M_E = queryMatrix(e)$$

$$M_A = queryMatrix(a)$$

$$M_P = queryMatrix(p).$$

where each M is a Structured Distributional Semantic matrix representing a single word or a multiword unit.

To judge coreference between events $\mathbf{E1}$ and $\mathbf{E2}$, we compute pairwise similarities $\text{Sim}(M1_{full}, M2_{full})$, $\text{Sim}(M1_{part:EA}, M2_{part:EA})$, etc., for each level of the composed triple representation. Furthermore, we vary the computation of similarity by considering different levels of granularity (lemma, SST), various choices of distance metric (Euclidean, city-block, cosine), and score normalization techniques (Row-wise, Full, Column collapsed). This results in 159 similarity-based features for every pair of events, which are used to train a classifier to make a binary decision for coreferentiality.

Model	IC Corpus				ECB Corpus			
	Prec	Rec	F-1	Acc	Prec	Rec	F-1	Acc
DSM	0.743	0.843	0.790	74.0	0.854	0.378	0.524	83.0
Senna	0.850	0.881	0.865	83.5	0.616	0.408	0.505	79.1
Senna+AVC	0.753	0.941	0.837	77.7	0.901	0.373	0.528	83.4
Senna+MVC	0.756	0.961	0.846	78.7	0.914	0.353	0.510	83.1
SDSM	0.916	0.929	0.922	90.6	0.901	0.401	0.564	84.3

Table 6.3: Cross-validation performance of Event Coreference Judgement on IC and ECB datasets. SDSM outperforms the other models on F-1 and accuracy on both datasets.

6.2.5.1 Datasets

We evaluate our method on two datasets and compare it against four baselines, two of which use window based distributional vectors and two that use simple forms of vector composition previously suggested in the literature.

IC Event Coreference Corpus: The dataset, drawn from 100 news articles about violent events, contains manually created annotations for 2214 pairs of co-referent and non-coreferent events each. Where available, events’ semantic role-fillers for *agent* and *patient* are annotated as well. When missing, empirical substitutes were obtained by querying the PropStore for the preferred word attachments.

EventCorefBank (ECB) Corpus: This corpus (Bejan and Harabagiu, 2010) of 482 documents from Google News is clustered into 45 topics, with event coreference chains annotated over each topic. The event mentions are enriched with semantic roles to obtain the canonical event structure described above. Positive instances are obtained by taking pairwise event mentions within each chain, and negative instances are generated from pairwise event mentions across chains, but within the same topic. This results in 11039 positive instances and 33459 negative instances.

6.2.5.2 Baselines

To establish the efficacy of our model, we compare SDSM against a purely window-based baseline (DSM) trained on the same corpus. In our experiments we set a window size of three words to either side of the target.

We also compare SDSM against the window-based embeddings trained using a neural network (SENNA) (Collobert et al., 2011) on both datasets. The SENNA pre-trained embeddings are state-of-the-art for many NLP tasks. This baseline uses SENNA to generate level 3 similarity features for events’ individual words (agent, patient and action), since no compositionality between words in a fixed VSM is possible.

As our final set of baselines, we extend two simple techniques proposed by Mitchell and Lapata (2008) that use element-wise addition and multiplication operators to perform composition. The two baselines thus obtained are Senna+AVC (element-wise addition) and Senna+MVC (element-wise multiplication).

6.2.5.3 Results and Discussion

We experimented with a number of common classifiers, and selected decision-trees (J48) as they gave the best classification accuracy across models. Table 6.3 summarizes our results on both datasets.

The results reveal that the SDSM model consistently outperforms DSM, SENNA embeddings, and the MVC and AVC models, both in terms of F-1 score and accuracy. The IC corpus comprises of domain specific texts, resulting in high lexical overlap between event mentions. Hence, the scores on the IC corpus are consistently higher than those on the ECB corpus.

The improvements over DSM and SENNA embeddings, support our hypothesis that syntax lends greater expressive power to distributional semantics in compositional configurations. Furthermore, the increase in predictive accuracy over MVC and AVC shows that our formulation of composition of two words based on the relation binding them yields a stronger form of composition than simple additive and multiplicative models.

We also performed an ablation study to determine the most predictive features for the task of determining event coreferentiality. The forward selection procedure revealed that the most informative attributes are the level 2 compositional features involving the agent and the action, as well as their individual level 3 features. This corresponds to the intuition that the agent and the action are the principal determiners for identifying events. Features involving the patient and level 1 representations are least useful. The latter involve full composition, resulting in very sparse representations and hence have low predictive power.

6.3 Latent Structured Distributional Semantics

We now turn to our second SDSM, which represents its relational dimensions as latent embeddings obtained from matrix factorization. This SDSM tackles the problem of relational facets in meaning comparison by moving away from syntax to a data-driven approach of choosing a set of relational dimensions.

One short-coming of DSMs derives from the fundamental nature of the vector space model, which characterizes the semantics of a word by a single vector in a high dimensional space (or some lower dimensional embedding thereof).

However, the semantics of a word is not unique. Rather, it is composed of many facets of meaning, and similarity (or dissimilarity) to other words is an outcome of the aggregate harmony (or dissonance) between the individual facets under consideration. For example, a shirt may be similar along one facet to a balloon in that they are both colored blue, at the same time being similar to a shoe along another facet for both being articles of clothing, while being dissimilar along yet another facet to a t-shirt because one is stitched from linen while the other is made from polyester.

Of course, in this example the facets of comparison are sensitive to a particular context instance. But at an aggregate level one would be inclined to say that a shirt is most synonymous to a t-shirt because it evidences a high similarity along the principal facet of relevance, or relation between the two words. It is our belief, as a result, that the semantics of a word may be captured more effectively by considering its *relations* to other words and their composite distributional

signature, rather than straightforwardly by co-occurrence statistics.

SDSMs in general provide a natural way of capturing the multiple facets of meaning of a concept by decomposing distributional signatures over discrete relation dimensions, or facets. This leads to a representation that characterizes the semantics of a word by a distributional tensor, whose constituent vectors each represent the semantic signature of a word along one particular relational facet of meaning.

In Section 6.2 we introduced one such SDSM. However it uses syntactic relational facets from a dependency parser as surrogates for semantic ones. Other similar previous approaches include work by Padó and Lapata (2007) and Baroni and Lenci (2010).

There are limiting factors to this approximation. Most importantly, the set of syntactic relations, while relatively uncontroversial, is unable to capture the full extent of semantic nuance encountered in natural language text. Often, syntax is ambiguous and leads to multiple semantic interpretations. Conversely, passivization and dative shift are common examples of semantic invariance in which multiple syntactic realizations are manifested. Additionally, syntax falls short in explaining more complex phenomena – such as the description of buying and selling – in which complex interactions between multiple participants must be represented.

Section 6.2 also revealed another very practical problem with a dynamic syntax-driven SDSM: that of sparsity.

While it is useful to consider relations that draw their origins from semantic roles such as Agent, Patient and Recipient, it remains unclear what the complete set of such semantic roles should be. This problem is one that has long troubled linguists (Fillmore, 1967; Sowa, 1991), and has been previously noted by researchers in NLP as well (Màrquez et al., 2008). Proposed solutions range from a small set of generic Agent-like or Patient-like roles in PropBank (Kingsbury and Palmer, 2002) to an effectively open-ended set of highly specific and fine-grained roles in FrameNet (Baker et al., 1998). In addition to the theoretic uncertainty of the set of semantic relations there is the very real problem of the lack of high-performance, robust semantic parsers to annotate corpora. These issues effectively make the use of pre-defined, linguistically ordained semantic relations intractable for use in SDSM.

In this section we introduce our second novel approach to structuring distributional semantic models, this time with *latent* relations that are automatically discovered from corpora. This approach solves the conceptual dilemma of selecting the most expressive set of semantic relations. It also solves the problems of sparsity plaguing the Syntactic SDSM, albeit at the expense of a dynamic model of composition. To the best of our knowledge this is the first work to propose latent relation dimensions for SDSM-type models. The intuition for generating these latent relation dimensions leads to a generic framework, which — in this paper — is instantiated with embeddings obtained from latent relational analysis (Turney, 2005).

We conduct experiments on three different semantic tasks to evaluate our model. On a similarity scoring task and another synonym ranking task the model significantly outperforms other distributional semantic models, including a standard window-based model, the previously described Syntactic SDSM, and a state-of-the-art semantic model trained using recursive neural networks. On a relation classification task, our model performs competitively, outperforming all but one of the models it is compared against.

Additionally, our analyses reveal that the learned SDSM captures semantic relations that broadly conform to intuitions of humanly interpretable relations. Optimal results for the first two

semantic tasks are reached when the number of latent dimensions vary in the proposed range of known fine-grained semantic relations (O’Hara and Wiebe, 2009). Moreover, a correlation analysis of the models trained on the relation classification task show that their respective weights distinguish fairly well between different semantic relations.

6.3.1 Related Work

Insensitivity to the multi-faceted nature of semantics has been one of the focal points of several papers. Earlier work in this regard is a paper by Turney (2012), who proposes that the semantics of a word is not obtained along a single distributional axis but simultaneously in two different spaces. He proposes a DSM in which co-occurrence statistics are computed for neighboring nouns and verbs separately to yield independent domain and function spaces of semantics.

This intuition is taken further by a stance which proposes that a word’s semantics is distributionally decomposed over *many* independent spaces – each of which is a unique relation dimension. Authors who have endorsed this perspective are Erk and Padó (2008), Goyal et al. (2013a), Reisinger and Mooney (2010) and Baroni and Lenci (2010). Our second model relates to these papers in that we subscribe to the multiple space semantics view.

However, this work differs from them by structuring the semantic space with information obtained from latent semantic relations rather than from a syntactic parser. In this section the instantiation of the SDSM with latent relation dimensions is obtained using LRA (Turney, 2005), which is an extension of LSA (Deerwester et al., 1990) to induce relational embeddings for pairs of words.

From a modelling perspective, SDSMs characterize the semantics of a word by a distributional tensor. Other notable papers on tensor based semantics or semantics of compositional structures are the simple additive and multiplicative models of Mitchell and Lapata (2009), the matrix-vector neural network approach of Socher et al. (2012), the physics inspired quantum view of semantic composition of Grefenstette and Sadrzadeh (2011) and the tensor-factorization model of Van de Cruys et al. (2013).

A different, partially overlapping line of research attempts to induce word embeddings using methods from deep learning, yielding state-of-the-art results on a number of different tasks. Notable research papers on this topic are the ones by Collobert et al. (2011), Turian et al. (2010) and Socher et al. (2010).

Other related work to note is the body of research concerned with semantic relation classification, which is one of our evaluation tasks. Research-community-wide efforts in the SemEval-2007 task 4 (Girju et al., 2007), the SemEval-2010 task 8 (Hendrickx et al., 2009) and the SemEval-2012 task 2 (Jurgens et al., 2012) are notable examples. However, different from our work, most previous attempts at semantic relation classification operate on the basis of feature engineering and contextual cues (Bethard and Martin, 2007). In this section the features for the task are induced automatically from word representations.

6.3.2 Latent Semantic Relation Induction

Recall from Section 6.1 the definition and notation of DSMs and SDSMs, and the relationship between the two.

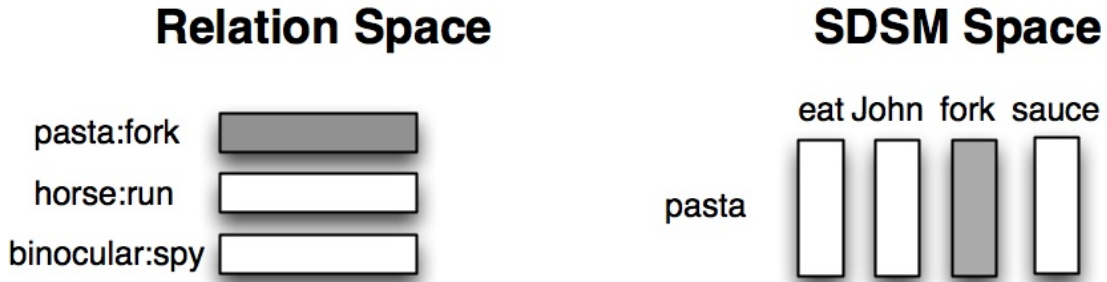


Figure 6.3: Visual representation of the intuition behind the Latent Relational SDSM. The idea revolves around inducing a relation space from which a row vector is transformed to a column vector in SDSM space.

6.3.2.1 Latent Relation Induction for SDSM

The intuition behind our approach for inducing latent relation dimensions revolves around the simple observation that SDSMs, while representing semantics as distributional signatures over relation dimensions, also effectively encode relational vectors between pairs of words. Our method thus works backwards from this observation — beginning with a relational embedding for pairs of words, that are subsequently transformed to yield an SDSM.

Concretely, given a vocabulary $\Gamma = \{w_1, w_2, \dots, w_k\}$ and a list of word pairs of interest from the vocabulary $\Sigma_V \subseteq \Gamma \times \Gamma$, we assume that we have some method for inducing a DSM V' that has a vector representation \vec{v}_{ij}^l of length d for every word pair $w_i, w_j \in \Sigma_V$, which intuitively embeds the distributional signature of the relation binding the two words in d latent dimensions. We then construct an SDSM U where $\Sigma_U = \Gamma$. For every word $w_i \in \Gamma$ a tensor $\vec{u}_i \in \mathbb{R}^{d \times k}$ is generated. The tensor \vec{u}_i has d unique k dimensional vectors $u_i^1, u_i^2, \dots, u_i^d$. For a given relational vector \vec{u}_i^l , the value of the j th element is taken from the l th element of the vector \vec{v}_{ij}^l belonging to the DSM V' . If the vector \vec{v}_{ij}^l does not exist in V' — as is the case where the pair $w_i, w_j \notin \Sigma_V$ — the value of the j th element of \vec{u}_i^l is set to 0. By applying this mapping to generate semantic tensors for every word in Γ , we are left with an SDSM U that effectively embeds latent relation dimensions. From the perspective of DMs we convert the third-order tensor into a two-dimensional matrix and perform truncated SVD, before restoring the resulting matrix to a third-order tensor.

This intuition is visually represented in Figure 6.3. The grey highlighted row vector is transposed from the “relation space” into a column vector of the representation for the word “pasta” in SDSM space.

Latent Relational Analysis

In what follows, we present our instantiation of this model with an implementation that is based on Latent Relational Analysis (LRA) (Turney, 2005) to generate the DSM V' . While other methods (such as RNNs) are equally applicable in this scenario, we use LRA for its operational simplicity as well as proven efficacy on semantic tasks such as analogy detection (which also

relies on relational information between word-pairs). The parameter values we chose in our experiments are not fine-tuned and are guided by recommended values from Turney (2005), or scaled suitably to accommodate the size of Σ_V .

The input to LRA is a vocabulary $\Gamma = \{w_1, w_2, \dots, w_k\}$ and a list of word pairs of interest from the vocabulary $\Sigma_V \subseteq \Gamma \times \Gamma$. While one might theoretically consider a large vocabulary with all possible pairs, for computational reasons we restrict our vocabulary to approximately 4500 frequent English words and only consider about 2.5% word pairs with high PMI (as computed on the whole of English Wikipedia) in $\Gamma \times \Gamma$. For each of the word pairs $w_i, w_j \in \Sigma_V$ we extract a list of contexts by querying a search engine indexed over the combined texts of the whole of English Wikipedia and Gigaword corpora (approximately 5.8×10^9 tokens). Suitable query expansion is performed by taking the top 4 synonyms of w_i and w_j using Lin’s thesaurus (Lin, 1998). Each of these contexts must contain both w_i, w_j (or appropriate synonyms) and optionally some intervening words, and some words to either side.

Given such contexts, patterns for every word pair are generated by replacing the two target words w_i and w_j with placeholder characters X and Y , and replacing none, some or all of the other words by their associated part-of-speech tag or a wildcard symbol “*”.

For example, consider the case when w_i and w_j are “eat” and “pasta” respectively, and the queried context is “I eat a bowl of pasta with a fork”. One would then generate all possible patterns, with the words “eat” and “pasta” substituted by the symbols X and Y , and the other words either being replaced by the part-of-speech tags, the wildcard symbol “*”, or not replaced at all. Here are some of the examples of patterns one might generate:

- * X * NN * Y IN a *
- * X DT bowl IN Y with DT *
- I X a NN of Y * * fork

For every word pair, only the 5000 most frequent patterns are stored.

Once the set of all relevant patterns $P = p_1, p_2, \dots, p_n$ have been computed a DSM V is constructed. In particular, the DSM constitutes a Σ_V based on the list of word pairs of interest, and every word pair w_i, w_j of interest has an associated vector \vec{v}_{ij} . Each element m of the vector \vec{v}_{ij} is a count pertaining to the number of times that the pattern p_m was generated by the word pair w_i, w_j .

SVD Transformation

The resulting DSM V is noisy and very sparse. Two transformations are thus applied to V . Firstly all co-occurrence counts between word pairs and patterns are transformed to PPMI scores (Bullinaria and Levy, 2007). Then given the matrix representation of V — where rows correspond to word pairs and columns correspond to patterns — SVD is applied to yield $V = M\Delta N$. Here M and N are matrices that have unit-length orthogonal columns and Δ is a matrix of singular values. By selecting the d top singular values, we approximate V with a lower dimension projection matrix that reduces noise and compensates for sparseness: $V' = M_d\Delta_d$. This DSM V' in d latent dimensions is precisely the one we then use to construct an SDSM, using the transformation described above.

Since the large number of patterns renders it effectively impossible to store the entire matrix V in memory we use a memory friendly implementation¹ of a multi-pass stochastic algorithm to directly approximate the projection matrix (Halko et al., 2011; Rehurek, 2010). A detailed analysis to see how change in the parameter d effects the quality of the model is presented in section 6.3.3.

The optimal SDSM embeddings we trained and used in the experiments detailed below are available for download at http://www.cs.cmu.edu/~sjauhar/Software_files/LR-SDSM.tar.gz. This SDSM contains a vocabulary of 4546 frequent English words with 130 latent relation dimensions.

6.3.3 Evaluation

Section 6.3.2 described a method for embedding latent relation dimensions in SDSMs. We now turn to the problem of evaluating these relations within the scope of the distributional paradigm in order to address two research questions:

1. Are latent relation dimensions a viable and empirically competitive solution for SDSM?
2. Does structuring lead to a semantically more expressive model than a non-structured DSM?

In order to answer these questions we evaluate our model on two generic semantic tasks and present comparative results against other structured and non-structured distributional models. We show that we outperform all of them significantly, thus answering both research questions affirmatively.

While other research efforts have produced better results on these tasks (Jarmasz and Szpakowicz, 2004; Gabrilovich and Markovitch, 2007; Hassan and Mihalcea, 2011), they are either lexicon or knowledge based, or are driven by corpus statistics that tie into auxiliary resources such as multi-lingual information and structured ontologies like Wikipedia. Hence they are not relevant to our experimental validation, and are consequently ignored in our comparative evaluation.

6.3.3.1 Word-Pair Similarity Scoring Task

The first task consists in using a semantic model to assign similarity scores to pairs of words. The dataset used in this evaluation setting is the WS-353 dataset from Finkelstein et al. (2002). It consists of 353 pairs of words along with an averaged similarity score on a scale of 1.0 to 10.0 obtained from 13–16 human judges. Word pairs are presented as-is, without any context. For example, an item in this dataset might be “book, paper → 7.46”.

System scores are obtained by using the standard cosine similarity measure between distributional vectors in a non-structured DSM. In the case of a variant of SDSM, these scores can be found by using the cosine-based similarity functions in Equation 6.1 of the previous section. System generated output scores are evaluated against the gold standard using Spearman’s rank correlation coefficient.

¹<http://radimrehurek.com/gensim/>

Model	WS-353 (Spearman’s rho)	ESL (% Accuracy)
Random	0.000	25.0
DSM	0.179	28.0
Syntactic SDSM	0.315	26.7
Senna	0.510	38.0
LR SDSM (300)	0.567	46.9
LR SDSM (130)	0.586	51.0

Table 6.4: Results on the WS-353 similarity scoring task and the ESL synonym selection task. LRA-SDSM significantly outperforms other structured and non-structured distributional semantic models.

6.3.3.2 Synonym Selection Task

In the second task, the same set of semantic space representations is used to select the semantically closest word to a target from a list of candidates. The ESL dataset from Turney (2002) is used for this task, and was selected over the slightly larger TOEFL dataset (Landauer and Dumais, 1997). The reason for this choice was because the latter contained more complex vocabulary words — several of which were not present in our simple vocabulary model. The ESL dataset consists of 50 target words that appear with 4 candidate lexical substitutes each. While disambiguating context is also given in this dataset, we discard it in our experiments. An example item in this dataset might be “rug → sofa, ottoman, carpet, hallway”, with “carpet” being the most synonym-like candidate to the target.

Similarity scores — which are obtained in the same manner as for the previous evaluation task — are extracted between the target and each of the candidates in turn. These scores are then sorted in descending order, with the top-ranking score yielding the semantically closest candidate to the target. Systems are evaluated on the basis of their accuracy at discriminating the top-ranked candidate.

6.3.3.3 Results

We compare our model (LR-SDSM) to several other distributional models in these experiments. These include a standard distributional vector space model (DSM) trained on the combined text of English Wikipedia and Gigaword with a window-size of 3 words to either side of a target, the Syntactic SDSM from Chapter 6.2 (synSDSM) trained on the same corpus parsed with a dependency parser (Tratz and Hovy, 2011) and the state-of-the-art neural network embeddings from Collobert et al. (2011) (SENNA). We also give the expected evaluation scores from a random baseline, for comparison.

An important factor to consider when constructing an SDSM using LRA is the number of latent dimensions selected in the SVD projection. In Figure 6.4 we investigate the effects of selecting different number of latent relation dimensions on both semantic evaluation tasks, starting with 10 dimensions up to a maximum of 800 (which was the maximum that was computationally feasible), in increments of 10. We note that optimal results on both datasets are obtained at 130

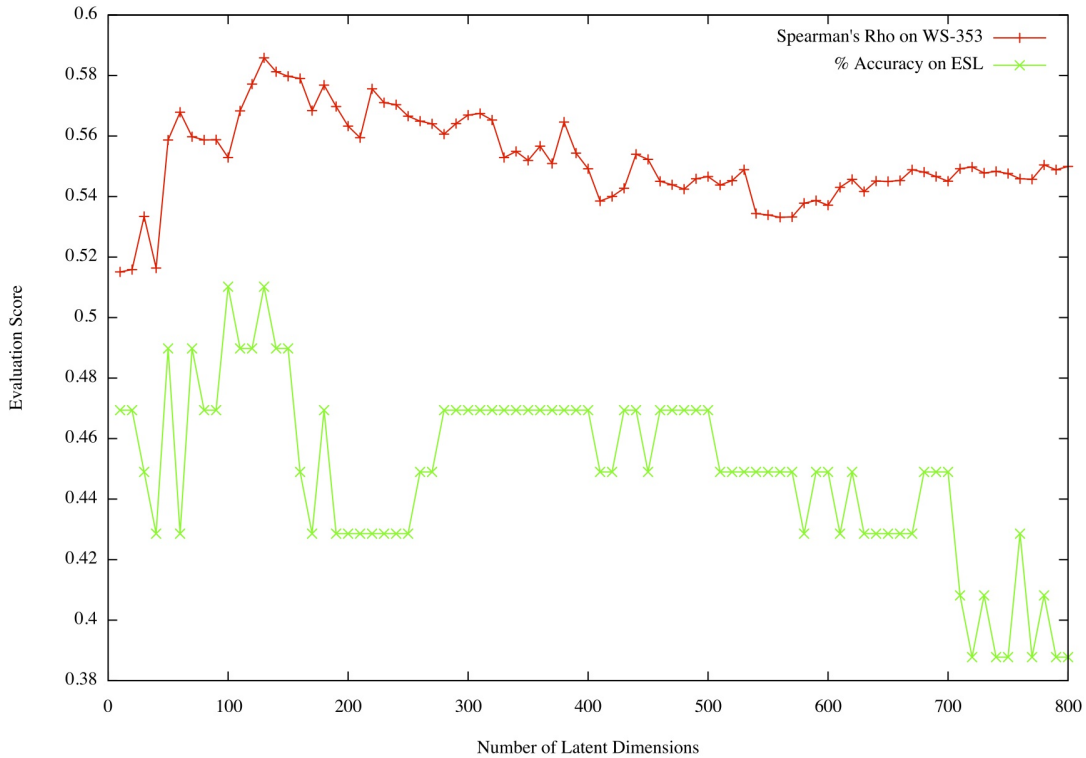


Figure 6.4: Evaluation results on WS-353 and ESL with varying number of latent dimensions. Generally high scores are obtained in the range of 100-150 latent dimensions, with optimal results on both datasets at 130 latent dimensions.

latent dimensions. In addition to the SDSM obtained in this setting we also give results for an SDSM with 300 latent dimensions (which has been a recommended value for SVD projections in the literature (Landauer and Dumais, 1997)) in our comparisons against other models. Comparative results on the Finkelstein WS-353 similarity scoring task and ESL synonym selection task are given in Table 6.4.

6.3.3.4 Discussion

The results in Table 6.4 show that LR-SDSM outperforms the other distributional models by a considerable and statistically significant margin (p -value < 0.05) on both types of semantic evaluation tasks. It should be noted that we do not tune to the test sets. While the 130 latent dimension SDSM yields the best results, 300 latent dimensions also gives comparable performance and moreover outperforms all the other baselines. In fact, it is worth noting that the evaluation results in figure 6.4 are almost all better than the results of the other models on either datasets.

We conclude that structuring of a semantic model with latent relational information in fact leads to performance gains over non-structured variants. Also, the latent relation dimensions we propose offer a viable and empirically competitive alternative to syntactic relations for SDSMs.

Figure 6.4 shows the evaluation results on both semantic tasks as a function of the number

Model	Relation Classification			
	Precision	Recall	F-1	% Accuracy
Random	0.111	0.110	0.110	11.03
Senna (Mik)	0.273	0.343	0.288	34.30
DSM+AVC	0.419	0.449	0.426	44.91
DSM+MVC	0.382	0.443	0.383	44.26
Senna+AVC	0.489	0.516	0.499	51.55
Senna+MVC	0.416	0.457	0.429	45.65
LR SDSM	0.431	0.475	0.444	47.48

Table 6.5: Results on Relation Classification Task. LR-SDSM scores competitively, outperforming all but the SENNA-AVC model.

of latent dimensions. The general trend of both curves on the figure indicate that the expressive power of the model quickly increases with the number of dimensions until it peaks in the range of 100–150, and then decreases or evens out after that. Interestingly, this falls roughly in the range of the 166 frequent (those that appear 50 times or more) frame elements, or fine-grained relations, from FrameNet that O’Hara and Wiebe (2009) find in their taxonomization and mapping of a number of lexical resources that contain semantic relations.

6.3.4 Semantic Relation Classification and Analysis of the Latent Structure of Dimensions

In this section we conduct experiments on the task of semantic relation classification. We also perform a more detailed analysis of the induced latent relation dimensions in order to gain insight into our model’s perception of semantic relations.

6.3.4.1 Semantic Relation Classification

In this task, a relational embedding is used as a feature vector to train a classifier for predicting the semantic relation between previously unseen word pairs. The dataset used in this experiment is from the SemEval-2012 task 2 on measuring the degree of relational similarity (Jurgens et al., 2012), since it characterizes a number of very distinct and interesting semantic relations. In particular it consists of an aggregated set of 3464 word pairs evidencing 10 kinds of semantic relations. We prune this set to discard pairs that don’t contain words in the vocabularies of the models we consider in our experiments. This leaves us with a dataset containing 933 word pairs in 9 classes (1 class was discarded altogether because it contained too few instances). The 9 semantic relation classes are: “Class Inclusion”, “Part-Whole”, “Similar”, “Contrast”, “Attribute”, “Non-Attribute”, “Case Relation”, “Cause-Purpose” and “Space-Time”. For example, an instance of a word pair that exemplifies the “Part-Whole” relationship is “engine:car”. Note that, as with previous experiments, word pairs are given without any context.

6.3.4.2 Results

We compare LR-SDSM on the semantic relation classification task to several different models. These include the additive vector composition (AVC) and multiplicative vector composition methods (MVC) proposed by Mitchell and Lapata (2009); we present both DSM and SENNA based variants of these models. We also compare against the vector difference method of Mikolov et al. (2013b) (SENNA-Mik) which sees semantic relations as a meaning preserving vector translation in an RNN embedded vector space. Finally, we note the performance of random classification as a baseline, for reference. We attempted to produce results of a syntactic SDSM on the task; however, the hard constraint imposed by syntactic adjacency meant that effectively all the word pairs in the dataset yielded zero feature vectors.

To avoid overfitting on all 130 original dimensions in our optimal SDSM, and also to render results comparable, we reduce the number of latent relation dimensions of LR-SDSM to 50. We similarly reduce the feature vector dimension of DSM-AVC and DSM-MVC to 50 by feature selection. The dimensions of SENNA-AVC, SENNA-MVC and SENNA-Mik are already 50, and are not reduced further.

For each of the methods we train a logistic regression classifier. We don't perform any tuning of parameters and set a constant ridge regression value of 0.2, which seemed to yield roughly the best results for all models. The performance on the semantic relation classification task in terms of averaged precision, recall, F-measure and percentage accuracy using 10-fold cross-validation is given in Table 6.5.

Additionally, to gain further insight into the LR-SDSM's understanding of semantic relations, we conduct a secondary analysis. We begin by training 9 one-vs-all logistic regression classifiers for each of the 9 semantic relations under consideration. Then pairwise correlation distances are measured between all pairs of weight vectors of the 9 models. Finally, the distance adjacency matrix is projected into 2-d space using multidimensional scaling. The result of this analysis is presented in Figure 6.5.

6.3.4.3 Discussion

Table 6.5 shows that LR-SDSM performs competitively on the relation classification task and outperforms all but one of the other models. The performance differences are statistically significant with a p-value < 0.5 . We believe that some of the expressive power of the model is lost by compressing to 50 latent relation dimensions, and that a greater number of dimensions might improve performance. However, testing a model with a 130-length dense feature vector on a dataset containing 933 instances would likely lead to overfitting and also not be comparable to the SENNA-based models that operate on 50-length feature vectors.

Other points to note from Table 6.5 are that the AVC variants of the the DSM and SENNA composition models tend to perform better than their MVC counterparts. Also, SENNA-Mik performs surprisingly poorly. It is worth noting, however, that Mikolov et al. (2013b) report results on fairly simple lexico-syntactic relations between words – such as plural forms, possessives and gender – while the semantic relations under consideration in the SemEval-2012 dataset are relatively more complex.

In the analysis of the latent structure of dimensions presented in Figure 6.5, there are few

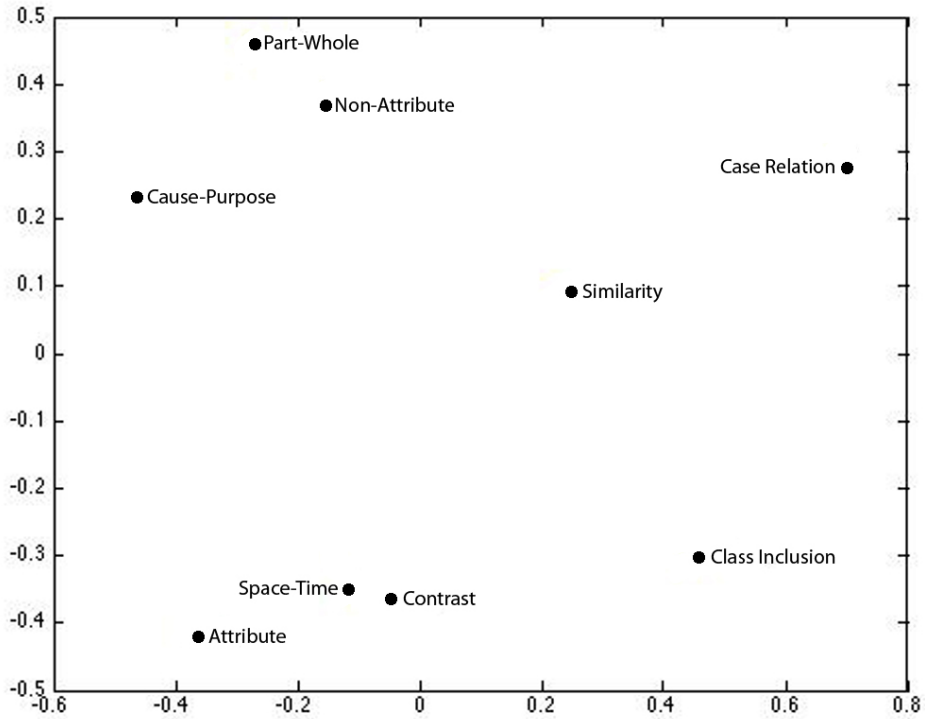


Figure 6.5: Correlation distances between semantic relations' classifier weights. The plot shows how our latent relations seem to perceive humanly interpretable semantic relations. Most points are fairly well spaced out, with opposites such as “Attribute” and “Non-Attribute” as well as “Similar” and “Contrast” being relatively further apart.

interesting points to note. To begin with, all the points (with the exception of one pair) are fairly well spaced out. At the weight vector level, this implies that different latent dimensions need to fire in different combinations to characterize distinct semantic relations, thus resulting in low correlation between their corresponding weight vectors. This indicates the fact that the latent relation dimensions seem to capture the intuition that each of the classes encodes a distinctly different semantic relation. The notable exception is “Space-Time”, which is very close to “Contrast”. This is probably due to the fact that distributional models are ineffective at capturing spatio-temporal semantics. Moreover, it is interesting to note that “Attribute” and “Non-Attribute” as well as “Similar” and “Contrast”, which are intuitively semantic inverses of each other are also (relatively) distant from each other in the plot.

These general findings indicate an interesting avenue for future research, which involves mapping the empirically learned latent relations to hand-built semantic lexicons or frameworks. This could help to validate the empirical models at various levels of linguistic granularity, as well as establish correspondences between different views of semantic representation.

6.4 Conclusion

In this chapter we outlined a general framework of representing semantics with a more expressive Structured Distributional formalism. This formalism tackled the problem of context-sensitivity and facetedness in semantics by a model that decomposes a vector representation into several discrete relation dimensions. We proposed two novel models that each instantiated these relation dimensions in a different way.

The first used dependency syntax and introduced a dynamic model that was capable of composing units into complex structure by taking relational context into account. By doing so it effectively tackled the problem of context-sensitivity in semantics. The Syntactic SDSM we introduced showed promising results on event coreference judgment, a task that specifically brings the problem of contextual variability into focus.

Our second instantiation of relation dimensions was a model that learned a set of latent relations from contextual pattern occurrences. This approach was better suited to resolve the issue of facetedness in semantics, or the facets along which a word may be represented or compared. Its data-driven approach allowed the SDSM to move away from syntax (and any set schema of relations for that matter), while also tackling the problem of sparsity that plagued the Syntactic SDSM. Experimental results of the model supported our claim that the Latent SDSM captures the semantics of words more effectively than a number of other semantic models, and presents a viable — and empirically competitive — alternative to Syntactic SDSMs. Finally, our analyses to investigate the structure of, and interactions between, the latent relation dimensions revealed some interesting tendencies on how they represented and differentiated between known, humanly-interpretable relations.

Chapter 7

Conclusion

We now conclude this thesis by summarizing our contributions and looking forward to avenues for future exploration.

7.1 Summary of Contributions

This thesis has presented a relation-centric view to the problem of representation learning in semantics.

- We have argued for this relation-centric view by positing that relations and structure are a core component of producing meaning in language. This led to developing an overarching view of semantic representation learning. Specifically, we drew attention to the fact that every representation learning model operates on the basis of a so-called *context* and that different hypotheses of semantic understanding give rise to different kinds of contexts. Relations and structure represented a way of consolidating and understanding these hypotheses and their associated contexts in a single view. They also suggested that different problems require different semantic understanding, and that articulating and then leveraging the right (or at least more expressive) relations for these problems can lead to better models.
- We then developed a set of five key research questions that presented a framework in which to think about and implement solutions that adopt the relation-centric view of semantic representation learning. The rest of our thesis is organized as an example-based instantiation of this high-level framework. Each of the questions dealt with one of five sequential themes. The first begins by identifying a representation learning problem that can be articulated in terms of a linguistic or structural phenomenon in natural language data. Second, the framework suggests identifying a corpus or knowledge resource that contains information relevant to solving the problem. The third then seeks to articulate the principal relations in the resource, and how they organize the data. After this, the fourth deals with leveraging the relations and structure in the data to define how *contexts* will be generated, and then specifying a mathematical model that will actually use these contexts. Fifth and

finally, the framework requires an evaluation of the learned model in order to see if the inclusion of relations measurably improved it.

- In Chapter 3 we looked at the problem of polysemy from the perspective of representation learning models. We suggested that ontologies contain a complimentary source of knowledge to distributional statistics, and that their structure can be used to tease apart representations for different senses of words that share a unique surface forms. To this end we developed two general and flexible solutions that integrates this structure into the process of learning sense representations that are grounded in an ontology. The first was an easy and computationally attractive post-processing step, called retrofitting, that split representations of word vectors into different sense vectors. The second built on the first method by also introducing a corpus into the learning process. Crucially, both methods leveraged the structure and relations within an ontology to define smoothness properties over the underlying graph, and used the neighborhoods of senses to tease them apart from other senses with which they shared a word surface form.
- Chapter 4 tackled the problem of QA in the context of the question, “What is the right level of structure for background knowledge”? In particular, we looked at background knowledge in the form of semi-structured tables with text entries. These tables represented interesting semantics over relations defined in terms of rows and columns that we leveraged in two important ways. First, we used table relations to guide non-expert annotators to create a large bank of multiple-choice questions. As a benefit of the annotation setup we also obtained fine-grained alignments from these multiple-choice questions to table elements, with no added cost or effort to annotators. Second, we used this newly created dataset of questions and table alignments to implement two question answering solutions. Crucially, both solutions leveraged the inherent structure of tables to find and use information relevant to answering questions. The first solution modelled table structure with a rich-set of hand-crafted features, while the second mimicked these features with a neural network model that automatically generated representations for table elements. The manually engineered model proved to be very successful, capable of abstracting over both the content and structure of tables to find answers in unseen test cases. The neural network model was also successful, but only at abstracting over structure and was more tied to content – understandably since its representations were learned over the content of tables.
- We explored the task of learning a semantic lexicon in Chapter 5. Specifically, we attempted to induce embeddings for a set of predicates and arguments, as well a set of latent relational argument slots associated with predicates – all while using minimal annotation. In fact, to the best of our knowledge, our work was the first to induce latent semantic role information without requiring high-level annotations such as dependency parses. Our solution revolved around the insight that the relations between predicates and arguments occur both at token and type level. To put this insight into practise we developed a novel integration between a predictive embedding model and the posterior of an Indian Buffet Process. By jointly leveraging the local (token) and global (type) levels of information over a corpus of predicate-argument pairs we were able to learn lexicon embeddings and relation slots that were maximally able to account for both types of relational information. We evaluated

our model at both the local token level and the global type level and demonstrated that the relational structure that we learn from latent slots in fact benefits the expressive power of the semantic lexicon. Finally, we also qualitatively showed that our solution was capable of capturing some interesting generalities of known, humanly-interpretable relations while having no access or supervision to guide it towards this understanding.

- Finally, in Chapter 6 we tackled the two related problems of context-sensitivity and facetedness in learning distributional semantic models. We outlined a framework that we called Structured Distributional Semantics to address these problems. The core idea behind Structured Distributional Semantics echoes the overall theme of this thesis, namely that relations play a key role in giving rise to semantics. Consequently the model decomposes the representation of a concept from a single vector into a relational tensor. The rows of this tensor define the relations that are the principle dimensions of representations of the resulting semantic space. We instantiated the relations of the tensor in two different ways that were devised to tackle the problems of context-sensitivity and facetedness. The first instantiation used syntactic relations and relied on a large store of syntactic triples in order to build compositional and context-sensitive representations on the fly. The second instantiation directly induced a set of latent semantic relations and used them to approach the problem of semantic facetedness. Both kinds of models produced empirically competitive results on relevant evaluations. And in the case of the latent relational model we again discovered (as in Chapter 5) some interesting tendencies of the latent relations to capture and differentiate between known, humanly-interpretable relations.

7.2 Future Directions

Each of the problems we have summarized above, and their respective solutions open up future avenues of research, such as performance improvements, extensions or scaling-up. The papers published from each of the corresponding chapters detail these future directions.

However, to conclude this thesis we mention some of the bigger trends that merit exploration. In particular we list three major themes.

Digging Deeper

From the five research questions and their corresponding topics, introduced in Chapter 2, we can deepen and generalize our approach to at least three of them. The first and the last of the themes concern defining and evaluating a problem and are necessarily task dependent, but the others are high-level generalizations that could encompass more than one problem.

The first and the second of the remaining three themes concerns the selection of the data source and the articulation of the principle relations within it. These two themes are deeply connected both in terms of this thesis as well as our envisaged outline of their future exploration.

In particular, the examples in this thesis have been split roughly into two major categories. The first used existing, at least partially manually created structured data that influenced the way we learn relations (such as ontologies in Chapters 3 and tables in 4), while the second attempted

to use latent relations in data to induce a set of relations for our models (such as predicate-argument pairs in Chapters 5 and text patterns in 6). In other words, we explored research use cases where we selected a structured resource that had a pre-defined set of well articulated relations, and other use cases where we were only given plain text (or some variant thereof) and were required to define the set of relations in terms of the text as well as the relations we were looking to induce.

The latter use case, of course is a more general approach, but using latent relations is often more difficult and produces models that are less interpretable and not as empirically effective. Exploring these themes further would require characterizing and numerically evaluating the differences between models that leverage structured and unstructured resources, or that learn from manually defined and latent relations. For example, our approach in Chapter 3 uses an ontology for sense representation and grounding, but there are approaches (Reisinger and Mooney, 2010; Neelakantan et al., 2014) that induce the senses in a latent way directly from plain text data. How do the senses learned by each of these methods differ from one another? Which ones are “better”? In general, can we do away with structured resources altogether and do all our learning from plain text corpora? This would provide us with a great deal of modelling power and flexibility to tackle the problems we chose.

The third research theme we wanted to broach, in terms of deeper and more general future work, concerns the models we define to learn representations. Most of the models detailed during the course of this thesis have been (perhaps necessarily) task specific. However, is it possible to frame many relation-centric representation learning problems in terms of a unified mathematical framework? The general idea behind the model introduced in Equation 3.3 of Chapter 3.2 represents one possible unification. This is an idea that we return to again in Equation 5.1 of Chapter 5.2. The general form of this idea is a regularized maximum-likelihood model, in which the likelihood and regularization components can be instantiated in different task-specific ways. This represents a powerful framework that fits into the over-arching goals of this thesis: learning representations from data with the influence of relations and structure.

In the two examples we showed in this thesis, we tackled two very different problems showing that this framework is in fact quite flexible. The first example dealt with maximizing the likelihood of a corpus with latent senses, while accounting for a posterior regularizer over an ontology. The second dealt with the very different problem of maximizing the likelihood of local token predicate-argument affinities with latent relation slots, while accounting for a posterior regularizer that captured global type-level affinities between them. One can imagine several problems that attempt to integrate relations or structure into the learning process would want to accommodate both a component that learns from data as well as a component that accounts for the relations or structure. The regularized maximum-likelihood framework presents an over-arching theme that is capable of assimilating these two components, and could be an interesting avenue to explore as a unified general model for relation-centric semantic representation learning.

Putting it All Together

The holy grail of semantics is a model that is capable of processing and understanding all of human language, with all its richness, variation and ambiguity. How do we move the research in-

troduced in this thesis towards that holy grail? Currently our models represent disparate attempts at tackling different nugget problems within the larger semantic space.

While we are not arguing for or against an approach that attempts to solve a vast problem like semantics in bits and pieces, an important question arises when you do have solutions to these bits and pieces: how do you put them all together? One could envisage a joint approach that pools all the different model outputs into a single semantic representation. Something similar is done by Yao et al. (2012) and Riedel et al. (2013), who use a so-called Universal Schema that is simply the union of different knowledge base schemas and representations in one gigantic whole.

But putting different semantic representations together, learned with different target problems in mind raises the natural follow-up question: how do you use the representations together, or how do you decide when to use which relation? We believe that resolving this problem requires an important future step in relation-centric learning.

We have introduced the relation-centric view of semantics as one of different kinds of contexts. Specifically, every relation gives rise to a different kind of context. We have, up to this point, only defined relations and used their corresponding contexts to learn representations. But what if we were to reverse the direction of influence? Namely, can we look a context and identify the specific relation that we need to focus on for a particular problem. This would enable us to select from potential candidates in an integrated representation model to select precisely the one we require to use for a particular problem.

An exploration of relation identification also would require a deeper understanding of the connection between different problems and their corresponding relations. Up until this point, we have defined and used these relations in a fairly ad-hoc fashion, intuiting on what might or might not be useful for a particular problem. In the future, a deeper understanding of the connection might lead to a more informed understanding and choice of relations for different problems.

What is a Relation?

This thesis was concerned with using semantic relations to learn more expressive models of semantics. Relations encompass a very broad range of phenomenon in language and data formalisms, and in Chapter 2 we attempted to provide a taxonomy of these occurrences of relations as it pertains to our work. This taxonomy served, both as a technical summary of this thesis, as well as a high-level guideline on how to think about using relations in representation learning of a new domain or problem.

However, an important research issue when dealing with relations for representation learning is the relations themselves. In particular, for a particular language phenomenon or data formalism, how many relations are there? What are they? How do they organize the elements in the domain to one another? Is the existing structure ideal?

It may be noted that finding answers to these theoretical questions does not necessarily affect the problem of *using* these relations for learning representations. However, these questions are crucial to providing a comprehensive account of relation-centric semantic representation learning. More importantly, when existing structured data in the form of annotated language or data is non-existent (say for a new domain or low resource language), these questions are no longer theoretical but are a practical precursor to using relations to learn more expressive models of semantics.

These questions became especially relevant, for example, when dealing with latent relations and relation induction, as we did in the second part of this thesis (Chapters 5 and 6). And finding answers to these questions become all the harder *because* of the fact that we are dealing with latent relations. Except for the simplest count-and-normalize word vector models that represent contexts as vocabulary counts or probabilities in an explicit manner, most representation learning models are dense and opaque. Naturally, understanding and articulating relations that are learned from latent contexts and represented in dense vectors or tensors is challenging.

We attempted to map induced relations to known, humanly-interpretable ones to gain insight into how our models are organized, both in Chapter 5 and 6. But we believe this understanding is only scratching the surface. Hence, an important avenue for future research involves attempting to understand and characterize learned relations in dense embedded spaces as they pertain to a particular domain of interest. For example, Li et al. (2015) propose a way of visualizing representations in the context of semantic compositionality to better understand what properties are being captured by the un-interpretable representations of neural models.

Bibliography

- Abend, Omri, Reichart, Roi and Rappoport, Ari (2009). *Unsupervised argument identification for semantic role labeling*. 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: Volume 1-Volume 1*, pp. 28–36. Association for Computational Linguistics.
- Aydin, Bahadır Ismail, Yilmaz, Yavuz Selim, Li, Yaliang, Li, Qi, Gao, Jing and Demirbas, Murat (2014). *Crowdsourcing for multiple-choice question answering*. In *Twenty-Sixth IAAI Conference*.
- Baker, Collin F, Fillmore, Charles J and Lowe, John B (1998). *The Berkeley Framenet project*. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pp. 86–90. Association for Computational Linguistics.
- Baroni, Marco and Lenci, Alessandro (2010). *Distributional memory: A general framework for corpus-based semantics*. *Comput. Linguist.*, volume 36 (4), pp. 673–721.
URL: http://dx.doi.org/10.1162/coli_a00016
- Baroni, Marco and Zamparelli, Roberto (2010). *Nouns are vectors, adjectives are matrices: representing adjective-noun constructions in semantic space*. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP '10*, pp. 1183–1193. Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: <http://dl.acm.org/citation.cfm?id=1870658.1870773>
- Bejan, Cosmin Adrian and Harabagiu, Sanda (2010). *Unsupervised event coreference resolution with rich linguistic features*. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL '10*, pp. 1412–1422. Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: <http://dl.acm.org/citation.cfm?id=1858681.1858824>
- Bengio, Yoshua, Courville, Aaron and Vincent, Pierre (2013). *Representation learning: A review and new perspectives*. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, volume 35 (8), pp. 1798–1828.
- Bethard, Steven and Martin, James H (2007). *CU-TMP: temporal relation classification using syntactic and semantic features*. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pp. 129–132. Association for Computational Linguistics.
- Bies, Ann, Mott, Justin, Warner, Colin and Kulick, Seth (2012). *English web treebank*. Linguistic Data Consortium, Philadelphia, PA.
- Blank, Andreas (2003). *Polysemy in the lexicon and in discourse*. *TRENDS IN LINGUISTICS*

- STUDIES AND MONOGRAPHS, volume 142, pp. 267–296.
- Bruni, Elia, Tran, Nam-Khanh and Baroni, Marco (2014). *Multimodal Distributional Semantics*. Journal of Artificial Intelligence Research (JAIR), volume 49, pp. 1–47.
- Bullinaria, John A and Levy, Joseph P (2007). *Extracting semantic representations from word co-occurrence statistics: A computational study*. Behavior Research Methods, volume 39 (3), pp. 510–526.
- Cafarella, Michael J, Halevy, Alon, Wang, Daisy Zhe, Wu, Eugene and Zhang, Yang (2008). *Websites: exploring the power of tables on the web*. Proceedings of the VLDB Endowment, volume 1 (1), pp. 538–549.
- Carpenter, Bob (2008). *Lazy sparse stochastic gradient descent for regularized multinomial logistic regression*. Technical report, Alias-i. Available at <http://lingpipe-blog.com/lingpipe-white-papers>.
- Carpuat, Marine and Wu, Dekai (2007). *Improving Statistical Machine Translation Using Word Sense Disambiguation*. In *EMNLP-CoNLL*, volume 7, pp. 61–72.
- Cheung, Jackie Chi Kit, Poon, Hoifung and Vanderwende, Lucy (2013). *Probabilistic frame induction*. arXiv preprint arXiv:1302.4813.
- Ciaramita, Massimiliano and Altun, Yasemin (2006). *Broad-coverage sense disambiguation and information extraction with a supersense sequence tagger*. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, EMNLP '06*, pp. 594–602. Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: <http://dl.acm.org/citation.cfm?id=1610075.1610158>
- Ciaramita, Massimiliano and Johnson, Mark (2000). *Explaining away ambiguity: Learning verb selectional preference with Bayesian networks*. In *Proceedings of the 18th conference on Computational linguistics-Volume 1*, pp. 187–193. Association for Computational Linguistics.
- Ciresan, Dan, Meier, Ueli and Schmidhuber, Jürgen (2012). *Multi-column deep neural networks for image classification*. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 3642–3649. IEEE.
- Clark, Peter (2015). *Elementary school science and math tests as a driver for AI: Take the Aristo Challenge*. Proceedings of IAAI, 2015.
- Clark, Peter, Etzioni, Oren, Khot, Tushar, Sabharwal, Ashish, Tafjord, Oyvind, Turney, Peter and Khashabi, Daniel (2016). *Combining Retrieval, Statistics, and Inference to Answer Elementary Science Questions*. Proceedings of the 30th AAAI Conference on Artificial Intelligence, AAAI-2016.
- Collobert, Ronan and Weston, Jason (2008). *A unified architecture for natural language processing: deep neural networks with multitask learning*. In *Proceedings of the 25th international conference on Machine learning, ICML '08*, pp. 160–167. ACM, New York, NY, USA.
URL: <http://doi.acm.org/10.1145/1390156.1390177>
- Collobert, Ronan, Weston, Jason, Bottou, Léon, Karlen, Michael, Kavukcuoglu, Koray and Kuksa, Pavel (2011). *Natural Language Processing (Almost) from Scratch*. J. Mach. Learn. Res., volume 999888, pp. 2493–2537.

URL: <http://dl.acm.org/citation.cfm?id=2078183.2078186>

- Corduneanu, Adrian and Jaakkola, Tommi (2002). *On information regularization*. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, pp. 151–158. Morgan Kaufmann Publishers Inc.
- Curran, James Richard (2004). *From Distributional to Semantic Similarity*. Technical report, University of Edinburgh. College of Science and Engineering. School of Informatics.
- Dahl, George, Mohamed, Abdel-rahman, Hinton, Geoffrey E et al. (2010). *Phone recognition with the mean-covariance restricted Boltzmann machine*. In *Advances in neural information processing systems*, pp. 469–477.
- Dalvi, Bhavana, Bhakthavatsalam, Sumithra, Clark, Chris, Clark, Peter, Etzioni, Oren, Fader, Anthony and Groeneveld, Dirk (2016). *IKE-An Interactive Tool for Knowledge Extraction*. In *AKBC@ NAACL-HLT*, pp. 12–17.
- Das, D. and Petrov, S. (2011). *Unsupervised part-of-speech tagging with bilingual graph-based projections*. Proceedings of the.
- Das, Dipanjan, Schneider, Nathan, Chen, Desai and Smith, Noah A (2010). *Probabilistic frame-semantic parsing*. In *Human language technologies: The 2010 annual conference of the North American chapter of the association for computational linguistics*, pp. 948–956. Association for Computational Linguistics.
- Das, Dipanjan and Smith, Noah A. (2011). *Semi-Supervised Frame-Semantic Parsing for Unknown Predicates*. In *Proc. of ACL*.
- Deerwester, Scott C., Dumais, Susan T, Landauer, Thomas K., Furnas, George W. and Harshman, Richard A. (1990). *Indexing by latent semantic analysis*. *JASIS*, volume 41 (6), pp. 391–407.
- Deng, Li, Seltzer, Michael L, Yu, Dong, Acero, Alex, Mohamed, Abdel-rahman and Hinton, Geoffrey E (2010). *Binary coding of speech spectrograms using a deep auto-encoder*. In *Interspeech*, pp. 1692–1695. Citeseer.
- Dinu, Georgiana and Lapata, Mirella (2010). *Measuring distributional similarity in context*. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp. 1162–1172. Association for Computational Linguistics.
- Doshi-Velez, Finale and Ghahramani, Zoubin (2009). *Accelerated sampling for the Indian buffet process*. In *Proceedings of the 26th annual international conference on machine learning*, pp. 273–280. ACM.
- Doshi-Velez, Finale, Miller, Kurt T, Van Gael, Jurgen, Teh, Yee Whye and Unit, Gatsby (2009). *Variational inference for the Indian buffet process*. In *Proceedings of the Intl. Conf. on Artificial Intelligence and Statistics*, volume 12, pp. 137–144.
- Erk, Katrin (2007). *A simple, similarity-based model for selectional preferences*. In *Annual Meeting - Association For Computational Linguistics*, volume 45, p. 216.
- Erk, Katrin and Padó, Sebastian (2008). *A structured vector space model for word meaning in context*. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '08*, pp. 897–906. Association for Computational Linguistics, Stroudsburg, PA, USA.

- Erk, Katrin and Padó, Sebastian (2010). *Exemplar-based models for word meaning in context*. In *Proceedings of the ACL 2010 Conference Short Papers*, pp. 92–97. Association for Computational Linguistics.
- Etzioni, Oren, Banko, Michele, Soderland, Stephen and Weld, Daniel S (2008). *Open information extraction from the web*. *Communications of the ACM*, volume 51 (12), pp. 68–74.
- Fader, Anthony, Soderland, Stephen and Etzioni, Oren (2011). *Identifying relations for open information extraction*. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1535–1545. Association for Computational Linguistics.
- Fader, Anthony, Zettlemoyer, Luke and Etzioni, Oren (2014). *Open question answering over curated and extracted knowledge bases*. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1156–1165. ACM.
- Faruqui, Manaal, Dodge, Jesse, Jauhar, Sujay K, Dyer, Chris, Hovy, Eduard and Smith, Noah A (2014). *Retrofitting Word Vectors to Semantic Lexicons*. arXiv preprint arXiv:1411.4166.
- Fillmore, Charles J (1967). *The Case for Case*. ERIC.
- Finkelstein, Lev, Gabilovich, Evgeniy, Matias, Yossi, Rivlin, Ehud, Solan, Zach, Wolfman, Gadi and Ruppín, Eytan (2002). *Placing Search in Context: The Concept Revisited*. In *ACM Transactions on Information Systems*, volume 20, pp. 116–131.
- Firth, John R. (1957). *A Synopsis of Linguistic Theory, 1930-1955*. *Studies in Linguistic Analysis*, pp. 1–32.
- Fung, Pascale and Chen, Benfeng (2004). *BiFrameNet: bilingual frame semantics resource construction by cross-lingual induction*. In *Proceedings of the 20th international conference on Computational Linguistics*, p. 931. Association for Computational Linguistics.
- Gabilovich, Evgeniy and Markovitch, Shaul (2007). *Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis*. In *IJCAI*, volume 7, pp. 1606–1611.
- Ganitkevitch, Juri, Van Durme, Benjamin and Callison-Burch, Chris (2013). *PPDB: The Paraphrase Database*. In *Proceedings of NAACL-HLT*, pp. 758–764. Association for Computational Linguistics, Atlanta, Georgia.
URL: <http://cs.jhu.edu/ccb/publications/ppdb.pdf>
- Gildea, Daniel and Jurafsky, Daniel (2002). *Automatic labeling of semantic roles*. *Computational linguistics*, volume 28 (3), pp. 245–288.
- Girju, Roxana, Nakov, Preslav, Nastase, Vivi, Szpakowicz, Stan, Turney, Peter and Yuret, Deniz (2007). *SemEval-2007 task 04: Classification of semantic relations between nominals*. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pp. 13–18. Association for Computational Linguistics.
- Glorot, Xavier, Bordes, Antoine and Bengio, Yoshua (2011). *Domain adaptation for large-scale sentiment classification: A deep learning approach*. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 513–520.
- Goldberg, Adele E (2005). *Argument Realization*. J.-O. Östman, & M. Fried, Construction grammars: Cognitive grounding and theoretical extensions, pp. 17–43.

- Golub, Gene H and Reinsch, Christian (1970). *Singular value decomposition and least squares solutions*. Numerische mathematik, volume 14 (5), pp. 403–420.
- Goyal, Kartik, Jauhar, Sujay Kumar, Li, Huiying, Sachan, Mrinmaya, Srivastava, Shashank and Hovy, Eduard (2013a). *A Structured Distributional Semantic Model for Event Co-reference*. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL – 2013)*.
- Goyal, Kartik, Jauhar, Sujay Kumar, Li, Huiying, Sachan, Mrinmaya, Srivastava, Shashank and Hovy, Eduard (2013b). *A Structured Distributional Semantic Model: Integrating Structure with Semantics*. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL – 2013), CVSC Workshop*.
- Grefenstette, Edward and Sadrzadeh, Mehrnoosh (2011). *Experimental Support for a Categorical Compositional Distributional Model of Meaning*. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '11*, pp. 1394–1404. Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: <http://dl.acm.org/citation.cfm?id=2145432.2145580>
- Grefenstette, Edward, Sadrzadeh, Mehrnoosh, Clark, Stephen, Coecke, Bob and Pulman, Stephen (2011). *Concrete sentence spaces for compositional distributional models of meaning*. In *Proceedings of the Ninth International Conference on Computational Semantics, IWCS '11*, pp. 125–134. Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: <http://dl.acm.org/citation.cfm?id=2002669.2002683>
- Grenager, Trond and Manning, Christopher D (2006). *Unsupervised discovery of a statistical verb lexicon*. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pp. 1–8. Association for Computational Linguistics.
- Griffiths, Thomas L and Ghahramani, Zoubin (2005). *Infinite latent feature models and the Indian buffet process*. In *NIPS*, volume 18, pp. 475–482.
- Griffiths, Thomas L and Ghahramani, Zoubin (2011). *The indian buffet process: An introduction and review*. *The Journal of Machine Learning Research*, volume 12, pp. 1185–1224.
- Guevara, Emiliano (2010). *A regression model of adjective-noun compositionality in distributional semantics*. In *Proceedings of the 2010 Workshop on GEometrical Models of Natural Language Semantics, GEMS '10*, pp. 33–37. Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: <http://dl.acm.org/citation.cfm?id=1870516.1870521>
- Guo, Jiang, Che, Wanxiang, Wang, Haifeng and Liu, Ting (2014). *Learning sense-specific word embeddings by exploiting bilingual resources*. In *Proceedings of COLING*, pp. 497–507.
- Haghighi, Aria and Klein, Dan (2009). *Simple coreference resolution with rich syntactic and semantic features*. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3, EMNLP '09*, pp. 1152–1161. Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: <http://dl.acm.org/citation.cfm?id=1699648.1699661>
- Halko, Nathan, Martinsson, Per-Gunnar and Tropp, Joel A (2011). *Finding structure with*

- randomness: Probabilistic algorithms for constructing approximate matrix decompositions*. SIAM review, volume 53 (2), pp. 217–288.
- Harris, Zellig (1954). *Distributional structure*. Word, volume 10 (23), pp. 146–162.
- Hassan, Samer and Mihalcea, Rada (2011). *Semantic Relatedness Using Salient Semantic Analysis*. In *AAAI*.
- Hendrickx, Iris, Kim, Su Nam, Kozareva, Zornitsa, Nakov, Preslav, Ó Séaghdha, Diarmuid, Padó, Sebastian, Pennacchiotti, Marco, Romano, Lorenza and Szpakowicz, Stan (2009). *SemEval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals*. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, pp. 94–99. Association for Computational Linguistics.
- Hjelm, Hans (2007). *Identifying cross language term equivalents using statistical machine translation and distributional association measures*. In *Proceedings of NODALIDA*, pp. 97–104. Citeseer.
- Hjort, Nils Lid, Holmes, Chris, Müller, Peter and Walker, Stephen G (2010). *Bayesian nonparametrics*, volume 28. Cambridge University Press.
- Hochreiter, Sepp and Schmidhuber, Jürgen (1997). *Long short-term memory*. Neural computation, volume 9 (8), pp. 1735–1780.
- Hovy, Eduard, Marcus, Mitchell, Palmer, Martha, Ramshaw, Lance and Weischedel, Ralph (2006). *OntoNotes: the 90% solution*. In *Proceedings of the human language technology conference of the NAACL, Companion Volume: Short Papers*, pp. 57–60. Association for Computational Linguistics.
- Huang, Eric H, Socher, Richard, Manning, Christopher D and Ng, Andrew Y (2012). *Improving word representations via global context and multiple word prototypes*. In *Proceedings of the 50th ACL: Long Papers-Volume 1*, pp. 873–882.
- Jackendoff, Ray (1987). *The Status of Thematic Roles in Linguistic Theory*. Linguistic Inquiry, volume 18 (3), pp. 369–411.
- Jain, Prateek, Netrapalli, Praneeth and Sanghavi, Sujay (2013). *Low-rank matrix completion using alternating minimization*. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pp. 665–674. ACM.
- Jarmasz, Mario and Szpakowicz, Stan (2004). *Roget’s Thesaurus and Semantic Similarity*. Recent Advances in Natural Language Processing III: Selected Papers from RANLP, volume 2003, p. 111.
- Jauhar, Sujay Kumar, Dyer, Chris and Hovy, Eduard (2015). *Ontologically Grounded Multi-sense Representation Learning for Semantic Vector Space Models*. In *Proc. NAACL*.
- Jauhar, Sujay Kumar and Hovy, Eduard (2014). *Inducing Latent Semantic Relations for Structured Distributional Semantics*. In *Proceedings of the 25th International Conference on Computational Linguistics, COLING 2014*, pp. 698–708.
- Jauhar, Sujay Kumar and Hovy, Eduard (2017). *Embedded Semantic Lexicon Induction with Joint Global and Local Optimization*. In *Proceedings of the 6th Joint Conference on Lexical and Computational Semantics (*SEM 2017)*, pp. 209–219. Association for Computational

Linguistics, Vancouver, Canada.

URL: <http://www.aclweb.org/anthology/S17-1025>

- Jauhar, Sujay Kumar, Turney, Peter D and Hovy, Eduard (2016a). *Tables as Semi-structured Knowledge for Question Answering*. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL-2016*, pp. 474–483.
- Jauhar, Sujay Kumar, Turney, Peter D. and Hovy, Eduard (2016b). *TabMCQ: A Dataset of General Knowledge Tables and Multiple-choice Questions*. In *Submitted to LREC-2015. Under Review*.
- Ji, Heng and Grishman, Ralph (2011). *Knowledge base population: Successful approaches and challenges*. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pp. 1148–1158. Association for Computational Linguistics.
- Johnson, Mark (2007). *Why doesn't EM find good HMM POS-taggers?* In *EMNLP-CoNLL*, pp. 296–305. Citeseer.
- Joubarne, Colette and Inkpen, Diana (2011). *Comparison of semantic similarity for different languages using the Google N-gram corpus and second-order co-occurrence measures*. In *Advances in Artificial Intelligence*, pp. 216–221. Springer.
- Jurgens, David A, Turney, Peter D, Mohammad, Saif M and Holyoak, Keith J (2012). *SemEval-2012 task 2: Measuring degrees of relational similarity*. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pp. 356–364. Association for Computational Linguistics.
- Kawahara, Daisuke, Peterson, Daniel, Popescu, Octavian, Palmer, Martha and Kessler, Fondazione Bruno (2014). *Inducing Example-based Semantic Frames from a Massive Amount of Verb Uses*. In *EACL*, pp. 58–67.
- Khot, Tushar, Balasubramanian, Niranjan, Gribkoff, Eric, Sabharwal, Ashish, Clark, Peter and Etzioni, Oren (2015). *Exploring Markov Logic Networks for Question Answering*. Proceedings of EMNLP, 2015.
- Kilgarriff, Adam (1997). *I don't believe in word senses*. *Computers and the Humanities*, volume 31 (2), pp. 91–113.
- Kingsbury, Paul and Palmer, Martha (2002). *From TreeBank to PropBank*. In *LREC*. Citeseer.
- Krizhevsky, Alex, Sutskever, Ilya and Hinton, Geoffrey E (2012). *Imagenet classification with deep convolutional neural networks*. In *Advances in neural information processing systems*, pp. 1097–1105.
- Lafferty, John, McCallum, Andrew and Pereira, Fernando CN (2001). *Conditional random fields: Probabilistic models for segmenting and labeling sequence data*. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pp. 282–289.
- Landauer, Thomas K and Dumais, Susan T. (1997). *A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge*. *Psychological review*, pp. 211–240.

- Lang, Joel and Lapata, Mirella (2011a). *Unsupervised semantic role induction via split-merge clustering*. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pp. 1117–1126. Association for Computational Linguistics.
- Lang, Joel and Lapata, Mirella (2011b). *Unsupervised semantic role induction with graph partitioning*. In *Proceedings of the conference on empirical methods in natural language processing*, pp. 1320–1331. Association for Computational Linguistics.
- Lang, Joel and Lapata, Mirella (2014). *Similarity-driven semantic role induction via graph partitioning*. *Computational Linguistics*, volume 40 (3), pp. 633–669.
- Lee, Heeyoung, Recasens, Marta, Chang, Angel, Surdeanu, Mihai and Jurafsky, Dan (2012). *Joint entity and event coreference resolution across documents*. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL '12*, pp. 489–500. Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: <http://dl.acm.org/citation.cfm?id=2390948.2391006>
- Lehmann, Jens, Isele, Robert, Jakob, Max, Jentsch, Anja, Kontokostas, Dimitris, Mendes, Pablo N, Hellmann, Sebastian, Morsey, Mohamed, Van Kleef, Patrick, Auer, Sören et al. (2015). *DBpedia – A large-scale, multilingual knowledge base extracted from Wikipedia*. *Semantic Web*, volume 6 (2), pp. 167–195.
- Levin, Beth and Rappaport Hovav, Malka (2005). *Argument Realization*. Cambridge University Press.
- Levy, Omer and Goldberg, Yoav (2014). *Neural word embedding as implicit matrix factorization*. In *Advances in Neural Information Processing Systems*, pp. 2177–2185.
- Li, Jiwei, Chen, Xinlei, Hovy, Eduard and Jurafsky, Dan (2015). *Visualizing and understanding neural models in nlp*. arXiv preprint arXiv:1506.01066.
- Limaye, Girija, Sarawagi, Sunita and Chakrabarti, Soumen (2010). *Annotating and searching web tables using entities, types and relationships*. *Proceedings of the VLDB Endowment*, volume 3 (1-2), pp. 1338–1347.
- Lin, Dekang (1998). *Automatic retrieval and clustering of similar words*. In *Proceedings of the 17th international conference on Computational linguistics-Volume 2*, pp. 768–774. Association for Computational Linguistics.
- Lin, Dekang and Pantel, Patrick (2001). *Discovery of inference rules for question-answering*. *Natural Language Engineering*, volume 7 (04), pp. 343–360.
- Marcus, Mitchell, Kim, Grace, Marcinkiewicz, Mary Ann, MacIntyre, Robert, Bies, Ann, Ferguson, Mark, Katz, Karen and Schasberger, Britta (1994). *The Penn Treebank: annotating predicate argument structure*. In *Proceedings of the workshop on Human Language Technology*, pp. 114–119. Association for Computational Linguistics.
- Màrquez, Lluís, Carreras, Xavier, Litkowski, Kenneth C and Stevenson, Suzanne (2008). *Semantic role labeling: an introduction to the special issue*. *Computational linguistics*, volume 34 (2), pp. 145–159.

- McCarthy, Diana and Carroll, John (2003). *Disambiguating Nouns, Verbs, and Adjectives Using Automatically Acquired Selectional Preferences*. *Comput. Linguist.*, volume 29 (4), pp. 639–654.
- McCarthy, Diana, Koeling, Rob, Weeds, Julie and Carroll, John (2004). *Finding predominant word senses in untagged text*. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, ACL '04*. Association for Computational Linguistics, Stroudsburg, PA, USA.
- Mikolov, Tomas, Chen, Kai, Corrado, Greg and Dean, Jeffrey (2013a). *Efficient Estimation of Word Representations in Vector Space*. arXiv preprint arXiv:1301.3781.
- Mikolov, Tomas, Karafiát, Martin, Burget, Lukáš, Cernocky, Jan and Khudanpur, Sanjeev (2010). *Recurrent neural network based language model*. *Proceedings of Interspeech*.
- Mikolov, Tomas, Yih, Wen-tau and Zweig, Geoffrey (2013b). *Linguistic regularities in continuous space word representations*. *Proceedings of NAACL-HLT*, pp. 746–751.
- Miller, George and Charles, Walter (1991). *Contextual correlates of semantic similarity*. In *Language and Cognitive Processes*, pp. 1–28.
- Miller, George A (1995). *WordNet: a lexical database for English*. *Communications of the ACM*, volume 38 (11), pp. 39–41.
- Mitchell, Jeff and Lapata, Mirella (2008). *Vector-based models of semantic composition*. In *Proceedings of ACL-08: HLT*, pp. 236–244.
- Mitchell, Jeff and Lapata, Mirella (2009). *Language models based on semantic composition*. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1, EMNLP '09*, pp. 430–439. Association for Computational Linguistics, Stroudsburg, PA, USA.
- Mnih, Andriy and Teh, Yee Whye (2012). *A fast and simple algorithm for training neural probabilistic language models*. In *Proceedings of the 29th International Conference on Machine Learning*, pp. 1751–1758.
- Moens, Marc and Steedman, Mark (1988). *Temporal ontology and temporal reference*. *Computational linguistics*, volume 14 (2), pp. 15–28.
- Mohammad, Saif, Dorr, Bonnie and Hirst, Graeme (2008). *Computing word-pair antonymy*. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 982–991. Association for Computational Linguistics.
- Moldovan, Dan, Clark, Christine, Harabagiu, Sanda and Maiorano, Steve (2003). *Cogex: A logic prover for question answering*. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pp. 87–93. Association for Computational Linguistics.
- Narayanan, Sridhar and Harabagiu, Sanda (2004). *Question answering based on semantic structures*. In *Proceedings of the 20th international conference on Computational Linguistics*, p. 693. Association for Computational Linguistics.
- Neelakantan, Arvind, Le, Quoc V and Sutskever, Ilya (2015). *Neural Programmer: Inducing latent programs with gradient descent*. arXiv preprint arXiv:1511.04834.

- Neelakantan, Arvind, Shankar, Jeevan, Passos, Alexandre and McCallum, Andrew (2014). *Efficient non-parametric estimation of multiple embeddings per word in vector space*. In *Proceedings of EMNLP*.
- Netrapalli, Praneeth, Jain, Prateek and Sanghavi, Sujay (2013). *Phase retrieval using alternating minimization*. In *Advances in Neural Information Processing Systems*, pp. 2796–2804.
- Nivre, Joakim (2003). *An efficient algorithm for projective dependency parsing*. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*. Citeseer.
- Nivre, Joakim (2005). *Dependency grammar and dependency parsing*. MSI report, volume 5133 (1959), pp. 1–32.
- O’Hara, Tom and Wiebe, Janyce (2009). *Exploiting semantic role resources for preposition disambiguation*. *Computational Linguistics*, volume 35 (2), pp. 151–184.
- Padó, Sebastian and Lapata, Mirella (2007). *Dependency-based construction of semantic space models*. *Computational Linguistics*, volume 33 (2), pp. 161–199.
- Palmer, Martha, Gildea, Daniel and Kingsbury, Paul (2005). *The proposition bank: An annotated corpus of semantic roles*. *Computational linguistics*, volume 31 (1), pp. 71–106.
- Pantel, Patrick and Lin, Dekang (2000). *Word-for-word glossing with contextually similar words*. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference, NAACL 2000*, pp. 78–85. Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: <http://dl.acm.org/citation.cfm?id=974305.974316>
- Pantel, Patrick and Lin, Dekang (2002). *Discovering word senses from text*. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 613–619. ACM.
- Pasupat, Panupong and Liang, Percy (2015). *Compositional semantic parsing on semi-structured tables*. arXiv preprint arXiv:1508.00305.
- Pedersen, Ted and Kolhatkar, Varada (2009). *WordNet::SenseRelate::AllWords: a broad coverage word sense tagger that maximizes semantic relatedness*. In *Proceedings of human language technologies: The 2009 annual conference of the north american chapter of the association for computational linguistics, companion volume: Demonstration session*, pp. 17–20. Association for Computational Linguistics.
- Pennington, Jeffrey, Socher, Richard and Manning, Christopher D (2014). *Glove: Global vectors for word representation*. *Proceedings of the Empirical Methods in Natural Language Processing (EMNLP 2014)*, volume 12, pp. 1532–1543.
- Pilehvar, Mohammad Taher, Jurgens, David and Navigli, Roberto (2013). *Align, Disambiguate and Walk: A Unified Approach for Measuring Semantic Similarity*. In *ACL (1)*, pp. 1341–1351.
- Pimplikar, Rakesh and Sarawagi, Sunita (2012). *Answering table queries on the web using column keywords*. *Proceedings of the VLDB Endowment*, volume 5 (10), pp. 908–919.
- Pradet, Quentin, Baguenier-Desormeaux, Jeanne, De Chalendar, Gaël and Danlos, Laurence (2013). *WoNeF: amélioration, extension et évaluation d’une traduction française automatique de WordNet*. *Actes de TALN 2013*, pp. 76–89.

- Rabiner, Lawrence and Juang, B (1986). *An introduction to hidden Markov models*. *IEEE ASSP Magazine*, volume 3 (1), pp. 4–16.
- Raghunathan, Karthik, Lee, Heeyoung, Rangarajan, Sudarshan, Chambers, Nathanael, Surdeanu, Mihai, Jurafsky, Dan and Manning, Christopher (2010). *A multi-pass sieve for coreference resolution*. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP '10*, pp. 492–501. Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: <http://dl.acm.org/citation.cfm?id=1870658.1870706>
- Ravichandran, Deepak and Hovy, Eduard (2002). *Learning surface text patterns for a question answering system*. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pp. 41–47. Association for Computational Linguistics.
- Reddy, Siva, Klapaftis, Ioannis P, McCarthy, Diana and Manandhar, Suresh (2011). *Dynamic and Static Prototype Vectors for Semantic Composition*. In *IJCNLP*, pp. 705–713.
- Rehurek, Radim (2010). *Fast and Faster: A Comparison of Two Streamed Matrix Decomposition Algorithms*. NIPS 2010 Workshop on Low-rank Methods for Large-scale Machine Learning.
- Reisinger, Joseph and Mooney, Raymond J (2010). *Multi-prototype vector-space models of word meaning*. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 109–117. Association for Computational Linguistics.
- Resnik, Philip (1997). *Selectional preference and sense disambiguation*. In *Proceedings of the ACL SIGLEX Workshop on Tagging Text with Lexical Semantics: Why, What, and How*, pp. 52–57. Washington, DC.
- Riedel, Sebastian, Yao, Limin, McCallum, Andrew and Marlin, Benjamin M (2013). *Relation Extraction with Matrix Factorization and Universal Schemas*. In *HLT-NAACL*, pp. 74–84.
- Ritter, Alan, Mausam and Etzioni, Oren (2010). *A latent dirichlet allocation method for selectional preferences*. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 424–434. Association for Computational Linguistics.
- Rooth, Mats, Riezler, Stefan, Prescher, Detlef, Carroll, Glenn and Beil, Franz (1999). *Inducing a semantically annotated lexicon via EM-based clustering*. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pp. 104–111. Association for Computational Linguistics.
- Rubenstein, Herbert and Goodenough, John B. (1965). *Contextual Correlates of Synonymy*. *Commun. ACM*, volume 8 (10), pp. 627–633.
- Rudolph, Sebastian and Giesbrecht, Eugenie (2010). *Compositional matrix-space models of language*. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL '10*, pp. 907–916. Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: <http://dl.acm.org/citation.cfm?id=1858681.1858774>
- Salton, Gerard, Wong, Anita and Yang, Chung-Shu (1975). *A vector space model for automatic indexing*. *Communications of the ACM*, volume 18 (11), pp. 613–620.

- Schütze, Hinrich (1998). *Automatic word sense discrimination*. *Comput. Linguist.*, volume 24 (1), pp. 97–123.
- Schwenk, Holger, Rousseau, Anthony and Attik, Mohammed (2012). *Large, pruned or continuous space language models on a gpu for statistical machine translation*. In *Proceedings of the NAACL-HLT 2012 Workshop: Will We Ever Really Replace the N-gram Model? On the Future of Language Modeling for HLT*, pp. 11–19. Association for Computational Linguistics.
- Séaghdha, Diarmuid O (2010). *Latent variable models of selectional preference*. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 435–444. Association for Computational Linguistics.
- Séaghdha, Diarmuid Ó and Korhonen, Anna (2011). *Probabilistic models of similarity in syntactic context*. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '11*, pp. 1047–1057. Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: <http://dl.acm.org/citation.cfm?id=2145432.2145545>
- Shen, Dan and Lapata, Mirella (2007). *Using Semantic Roles to Improve Question Answering*. In *EMNLP-CoNLL*, pp. 12–21.
- Smith, Noah A (2004). *Log-linear models*.
- Smith, Noah A (2011). *Linguistic structure prediction*. *Synthesis lectures on human language technologies*, volume 4 (2), pp. 1–274.
- Socher, Richard, Huval, Brody, Manning, Christopher D. and Ng, Andrew Y. (2012). *Semantic compositionality through recursive matrix-vector spaces*. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL '12*, pp. 1201–1211. Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: <http://dl.acm.org/citation.cfm?id=2390948.2391084>
- Socher, Richard, Manning, Christopher D and Ng, Andrew Y (2010). *Learning continuous phrase representations and syntactic parsing with recursive neural networks*. *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*.
- Soderland, Stephen (1999). *Learning information extraction rules for semi-structured and free text*. *Machine learning*, volume 34 (1-3), pp. 233–272.
- Sowa, John F. (1991). *Principles of Semantic Networks*. Morgan Kaufmann.
- Srihari, Rohini and Li, Wei (1999). *Information extraction supported question answering*. Technical report, DTIC Document.
- Srivastava, Nitish, Hinton, Geoffrey E, Krizhevsky, Alex, Sutskever, Ilya and Salakhutdinov, Ruslan (2014). *Dropout: a simple way to prevent neural networks from overfitting*. *Journal of Machine Learning Research*, volume 15 (1), pp. 1929–1958.
- Stoyanov, Veselin, Gilbert, Nathan, Cardie, Claire and Riloff, Ellen (2009). *Conundrums in noun phrase coreference resolution: making sense of the state-of-the-art*. 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: Volume 2 - Volume 2, ACL '09*,

- pp. 656–664. Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: <http://dl.acm.org/citation.cfm?id=1690219.1690238>
- Subramanya, Amarnag and Bilmes, Jeff A (2009). *Entropic Graph Regularization in Non-Parametric Semi-Supervised Classification*. In *NIPS*, pp. 1803–1811.
- Suchanek, Fabian M, Kasneci, Gjergji and Weikum, Gerhard (2007). *Yago: a core of semantic knowledge*. In *Proceedings of the 16th international conference on World Wide Web*, pp. 697–706. ACM.
- Sun, Huan, He, Xiaodong, Yih, Wen-tau, Su, Yu and Yan, Xifeng (2016). *Table Cell Search for Question Answering*. In *Proceedings of the 25th International Conference on World Wide Web (to appear)*.
- Surdeanu, Mihai, Harabagiu, Sanda, Williams, John and Aarseth, Paul (2003). *Using predicate-argument structures for information extraction*. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pp. 8–15. Association for Computational Linguistics.
- Surdeanu, Mihai, Johansson, Richard, Meyers, Adam, Màrquez, Lluís and Nivre, Joakim (2008). *The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies*. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, pp. 159–177. Association for Computational Linguistics.
- Syed, Zareen, Finin, Tim, Mulwad, Varish and Joshi, Anupam (2010). *Exploiting a web of semantic data for interpreting tables*. In *Proceedings of the Second Web Science Conference*.
- Tellex, Stefanie, Katz, Boris, Lin, Jimmy, Fernandes, Aaron and Marton, Gregory (2003). *Quantitative evaluation of passage retrieval algorithms for question answering*. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 41–47. ACM.
- Thater, Stefan, Fürstenau, Hagen and Pinkal, Manfred (2010). *Contextualizing semantic representations using syntactically enriched vector models*. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL '10*, pp. 948–957. Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: <http://dl.acm.org/citation.cfm?id=1858681.1858778>
- Thater, Stefan, Fürstenau, Hagen and Pinkal, Manfred (2011). *Word Meaning in Context: A Simple and Effective Vector Model*. In *IJCNLP*, pp. 1134–1143.
- Tian, Fei, Dai, Hanjun, Bian, Jiang, Gao, Bin, Zhang, Rui, Chen, Enhong and Liu, Tie-Yan (2014). *A probabilistic model for learning multi-prototype word embeddings*. In *Proceedings of COLING*, pp. 151–160.
- Titov, Ivan and Klementiev, Alexandre (2012a). *A Bayesian approach to unsupervised semantic role induction*. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pp. 12–22. Association for Computational Linguistics.
- Titov, Ivan and Klementiev, Alexandre (2012b). *Crosslingual induction of semantic roles*. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pp. 647–656. Association for Computational Linguistics.

- Tratz, Stephen and Hovy, Eduard (2011). *A fast, accurate, non-projective, semantically-enriched parser*. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pp. 1257–1268. Association for Computational Linguistics, Stroudsburg, PA, USA.
URL: <http://dl.acm.org/citation.cfm?id=2145432.2145564>
- Turian, Joseph, Ratinov, Lev and Bengio, Yoshua (2010). *Word representations: a simple and general method for semi-supervised learning*. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 384–394. Association for Computational Linguistics.
- Turney, Peter (2005). *Measuring semantic similarity by latent relational analysis*. In *Proceedings of the 19th international Conference on Artificial Intelligence*, pp. 1136–1141.
- Turney, Peter (2007). *Attributes and Relations: Redder than Red*.
URL: <http://blog.apperceptual.com/attributes-and-relations-redder-than-red>
- Turney, Peter D. (2002). *Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL*. CoRR.
- Turney, Peter D. (2006). *Similarity of Semantic Relations*. *Comput. Linguist.*, volume 32 (3), pp. 379–416.
URL: <http://dx.doi.org/10.1162/coli.2006.32.3.379>
- Turney, Peter D (2008a). *The Latent Relation Mapping Engine: Algorithm and Experiments*. *Journal of Artificial Intelligence Research (JAIR)*, volume 33, pp. 615–655.
- Turney, Peter D (2008b). *A uniform approach to analogies, synonyms, antonyms, and associations*. In *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pp. 905–912.
- Turney, Peter D (2011). *Analogy perception applied to seven tests of word comprehension*. *Journal of Experimental & Theoretical Artificial Intelligence*, volume 23 (3), pp. 343–362.
- Turney, Peter D (2012). *Domain and function: A dual-space model of semantic relations and compositions*. *Journal of Artificial Intelligence Research*, volume 44, pp. 533–585.
- Turney, Peter D (2013). *Distributional semantics beyond words: Supervised learning of analogy and paraphrase*. *Transactions of the Association for Computational Linguistics*, volume 1, pp. 353–366.
- Turney, Peter D. and Pantel, Patrick (2010). *From frequency to meaning : Vector space models of semantics*. *Journal of Artificial Intelligence Research*, pp. 141–188.
- Van de Cruys, Tim (2014). *A Neural Network Approach to Selectional Preference Acquisition*. In *EMNLP*, pp. 26–35.
- Van de Cruys, Tim, Poibeau, Thierry and Korhonen, Anna (2011). *Latent vector weighting for word meaning in context*. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 1012–1022. Association for Computational Linguistics.
- Van de Cruys, Tim, Poibeau, Thierry and Korhonen, Anna (2013). *A tensor-based factorization model of semantic compositionality*. In *Proceedings of NAACL-HLT*, pp. 1142–1151.
- Vasilescu, M. Alex O. and Terzopoulos, Demetri (2002). *Multilinear Analysis of Image Ensem-*

- bles: *TensorFaces*. In *In Proceedings of the European Conference on Computer Vision*, pp. 447–460.
- Venetis, Petros, Halevy, Alon, Madhavan, Jayant, Paşca, Marius, Shen, Warren, Wu, Fei, Miao, Gengxin and Wu, Chung (2011). *Recovering Semantics of Tables on the Web*. Proc. VLDB Endow., volume 4 (9), pp. 528–538.
URL: <http://dx.doi.org/10.14778/2002938.2002939>
- Weston, Jason, Bordes, Antoine, Chopra, Sumit and Mikolov, Tomas (2015). *Towards AI-complete question answering: a set of prerequisite toy tasks*. arXiv preprint arXiv:1502.05698.
- Wong, S. K. M. and Raghavan, Vijay V. (1984). *Vector space model of information retrieval: a reevaluation*. In *Proceedings of the 7th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '84*, pp. 167–185. British Computer Society, Swinton, UK.
- Woodsend, Kristian and Lapata, Mirella (2015). *Distributed Representations for Unsupervised Semantic Role Labeling*. In *EMNLP*, pp. 2482–2491. Citeseer.
- Wu, Fei and Weld, Daniel S (2010). *Open information extraction using Wikipedia*. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pp. 118–127. Association for Computational Linguistics.
- Yao, Limin, Riedel, Sebastian and McCallum, Andrew (2012). *Probabilistic databases of universal schema*. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pp. 116–121. Association for Computational Linguistics.
- Yin, Pengcheng, Duan, Nan, Kao, Ben, Bao, Junwei and Zhou, Ming (2015a). *Answering Questions with Complex Semantic Constraints on Open Knowledge Bases*. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pp. 1301–1310. ACM.
- Yin, Pengcheng, Lu, Zhengdong, Li, Hang and Kao, Ben (2015b). *Neural Enquirer: Learning to Query Tables*. arXiv preprint arXiv:1512.00965.
- Yu, Mo and Dredze, Mark (2014). *Improving Lexical Embeddings with Semantic Knowledge*. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*.
- Zhang, Zhu (2004). *Weakly-supervised relation classification for information extraction*. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pp. 581–588. ACM.
- Zhong, Zhi and Ng, Hwee Tou (2010). *It makes sense: A wide-coverage word sense disambiguation system for free text*. In *Proceedings of the ACL 2010 System Demonstrations*, pp. 78–83. Association for Computational Linguistics.
- Zhu, Xiaojin, Ghahramani, Zoubin, Lafferty, John et al. (2003). *Semi-supervised learning using gaussian fields and harmonic functions*. In *ICML*, volume 3, pp. 912–919.