Luis Nieto Piña

Splitting rocks: Learning word sense representations from corpora and lexica

Data linguistica
<http://hum.gu.se/institutioner/svenska-spraket/publ/datal/>

Editor: Lars Borin

Språkbanken • Språkdata
Department of Swedish Language
University of Gothenburg

30 • 2019

Luis Nieto Piña

Splitting rocks: Learning word sense representations from corpora and lexica

Gothenburg 2019

Data linguistica 30
ISBN 978-91-87850-75-2
ISSN 0347-948X

Printed in Sweden by
GU Interntryckeri 2019

Typeset in $\LaTeX\,2_\varepsilon$ by the author

Cover design by Jessica Oscarsson

Front cover illustration:
Splitting Rocks
by Charlotta Duse and Luis Nieto Piña ©

Author photo on back cover by Charlotta Duse

Abstract

The representation of written language semantics is a central problem of language technology and a crucial component of many natural language processing applications, from part-of-speech tagging to text summarization. These representations of linguistic units, such as words or sentences, allow computer applications that work with language to process and manipulate the meaning of text. In particular, a family of models has been successfully developed based on automatically learning semantics from large collections of text and embedding them into a vector space, where semantic or lexical similarity is a function of geometric distance. Co-occurrence information of words in context is the main source of data used to learn these representations.

Such models have typically been applied to learning representations for word forms, which have been widely applied, and proven to be highly successful, as characterizations of semantics at the word level. However, a word-level approach to meaning representation implies that the different meanings, or senses, of any polysemic word share one single representation. This might be problematic when individual word senses are of interest and explicit access to their specific representations is required. For instance, in cases such as an application that needs to deal with word senses rather than word forms, or when a digital lexicon's sense inventory has to be mapped to a set of learned semantic representations.

In this thesis, we present a number of models that try to tackle this problem by automatically learning representations for word senses instead of for words. In particular, we try to achieve this by using two separate sources of information: corpora and lexica for the Swedish language. Throughout the five publications compiled in this thesis, we demonstrate that it is possible to generate word sense representations from these sources of data individually and in conjunction, and we observe that combining them yields superior results in terms of accuracy and sense inventory coverage. Furthermore, in our evaluation of the different representational models proposed here, we showcase the applicability of word sense representations both to downstream natural language processing applications and to the development of existing linguistic resources.

Sammanfattning

Att representera semantiken för skrivet språk är ett centralt problem inom språkteknologin. Semantiska representationer för språkliga enheter – framför allt ord men även meningar, stycken och hela dokument – används i en rad olika tillämpningar, allt från ordklassmärkning till sammanfattning. Dessa representationer är en förutsättning för att applikationer som hanterar språk ska kunna resonera om språkliga enheters betydelse. En grupp av metoder för ordrepresentation som har visat sig praktiskt användbara representerar ord genom att inbädda dem i ett vektorrum, och genom denna inbäddning kan semantiska relationer ges en geometrisk tolkning. Dessa metoder utnyttjar information från stora mängder textmaterial, framför allt statistik om ords samförekomst.

Sådana metoder har typiskt använts för att skapa representationer för enskilda ordformer, och har på senare år blivit självklara standardverktyg för att praktiskt hantera ords semantik i språkteknologiska tillämpningar. En nackdel med representationsmetoder som helt och hållet baseras på ordformer är att om ett ord har flera möjliga betydelser (på grund av homonymi eller polysemi) så kommer representationen att bestå av en blandning av dessa betydelser. Detta kan vara problematiskt i tillämpningar där det är viktigt att skilja på de olika betydelserna, till exempel då tillämpningen uttryckligen behöver förhålla sig till digitala lexikon där ordbetydelser ingår.

I denna avhandling presenteras ett antal olika modeller som kringgår denna svårighet genom att automatiskt skapa representationer för ordbetydelser i stället för ordformer. För att åstadkomma detta utnyttjas svenskspråkiga korpusar och lexikon. I de fem artiklar som presenteras i avhandlingen visar vi att det är möjligt att skapa representationer av ordbetydelser utifrån korpusdata och lexikondata dels separat och dels kombinerat, och vi konstaterar att en kombination av de olika datakällorna ger oss bättre kvalitet i tillämpningar och bättre täckning av ordens olika betydelser. I utvärderingarna av de olika representationerna kan vi se att de kan fungera i språkteknologiska tillämpningar som betydelsedisambiguering, samt i lexikografiska tillämpningar där de kan användas för att föreslå tillägg till existerande lexikon.

Acknowledgements

I would like to express my gratitude to the individuals who, through their support and contributions, have made my thesis work not only possible, but enjoyable. First in this list is my main supervisor, Richard. From our very first video call to our recent editorial meetings, he has shown an unwavering ability to gently guide, teach, and provide room for growth that has shaped my work and understanding of academia over the last five years. I count myself lucky to have been his student. My gratitude goes also to my co-supervisor, Lars, whose tireless work and knowledge is the foundation of a workplace I know I will miss sorely.

My colleagues at Språkbanken have become a second family: they go beyond collegial duties to build a truly enjoyable and supportive environment. Working and partying with such a bright group of people has been a pleasure. This is extensive to the larger Language Technology community in Gothenburg, at Chalmers University of Technology and at the University of Gothenburg, which forms a unique and creative academic setting that fosters outstanding research and makes us all proud when we present our work to the international community. A special thank you to my fellow PhD students, past and present, with whom I have spent so much time in and out of the office. At times academic discussion club, at times therapeutic support group, always a friendly bunch. You guys made me feel right at home. My appreciation also to the Running Lady, who has unwittingly been living proof that constancy beats form in the long run.

The work in this thesis would have little meaning if not considered in the context of the international NLP community at large. I have shared insightful discussions with and learned from the work of many of you, for which I am grateful. It is a strange and joyful feeling to travel half a world away only to find familiar faces time and again, always eager to talk about recent developments over a beer, jetlag be damned. To be part of such a productive and brilliant community is a point of pride. I am particularly grateful to Magnus Sahlgren, who acted as discussant for my final seminar and provided really insightful and detailed feedback on the first draft of this text.

I sincerely acknowledge the various entities that generously supported my PhD work and allowed me to become a proud member of the research community: the Swedish Research Council, the University of Gothenburg through Språkbanken and the Center for Language Technology, and the Filosofiska fakulteternas gemensamma donationsnämnd.

I am fortunate to count with the support of family both in Spain and Sweden. I could not possibly list all I am grateful for to my parents, José Luis and María Paz, and my sisters and their families, since they nurtured many traits that have led me to this point and supported my every step along the way. My transition to Sweden was a breeze thanks to my Swedish family and friends. They provide a support net and a feeling of belonging that makes life abroad to be just life.

Last but not least, I thank Lotta for her companionship, encouragement, sacrifice, and understanding, and for her help proofreading, translating Swedish, and designing the cover of this book. Without all of it this work would have been a far, far harder road; she has led me by my hand whenever I could not do it by myself.

Contents

Part I

Introduction and background

1 Introduction

1.1 Motivation

During the last decade, computer assistance performed through the use of human language is solidifying from a long-anticipated concept into an everyday sideshow that lets us interact with our ever-increasing layer of technological apparatus. This inconspicuous success is owed to a sustained research effort in the different fields that coexist under the umbrella of Language Technology (LT): Artificial Intelligence (AI) applied to human language, computerized linguistic models, and speech technology.

Interestingly, the comparatively fast development on LT that is occurring in the last few years, contextualized in the enthusiasm for any and all AI technologies that appears to be the norm nowadays, follows a long dry period known as *AI Winter*, starting in the late 1980s, which decelerated progress in the field of AI motivated by lack of interest and, hence, funding: "At its low point, some computer scientists and software engineers avoided the term artificial intelligence for fear of being viewed as wild-eyed dreamers." (Markoff 2005). That lack of interest was itself due to a number of reasons from failure to live up to the hype created to budget-cutting policies for universities. Not small among these factors was the unavailability of computational power needed for neural network models to fulfill their potential. And part of today's more optimist standpoint is precisely due to hardware advancements which increase the capabilities of neural networks.

However, the decade of the 1990s was not devoid of advances in LT. It was precisely during this period that the "statistical revolution" (Johnson 2009) took place, a paradigm switch from rule-based to data-driven systems: an increase in available digital data and computational power favored informing systems with statistical data over sets of rules grounded in linguistic theory. In this context, meaning representation models based on statistics thrived. A family of models focused on providing words with semantic representations in a vector space, usually configured using co-occurrence statistics gathered from corpora, grew and, slowly but surely, started paving the way towards widespread adoption (Deerwester et al. 1990; Schütze 1993; Lund and Burgess 1996; Landauer and Dumais 1997).

From a variety of approaches to obtain vector representations of words and other lexical units, representations learned by neural networks stemming from neural language models (Bengio et al. 2003) have recently attracted the community's attention for their efficiency generating accurate semantic representations from large collections of text. Having high quality semantic representations has proven beneficial in a large number of Natural Language Processing (NLP) tasks such as syntactic parsing (Socher et al. 2013), named entity recognition and chunking (Turian, Ratinov and Bengio 2010), part-of-speech tagging and semantic role labeling (Collobert et al. 2011), or sentiment analysis (Glorot, Bordes and Bengio 2011). This good record, paired with ML advancements facilitated by increased accessibility to new and old, revisited powerful neural network models, has resulted in a myriad of refined representation models. Given that the main data source on which these models feed is text, it is not surprising that the majority of these models focus on representing the key building brick of that kind of data: word forms.

However, word representations suffer from a well-known limitation: they ignore polysemy, homonymy, and other related phenomena by which one word form may have more than one meaning. Word representation models, by forcing each word to be represented by one vector, may conflate several meanings into one representation, making recovery of an individual meaning difficult or impossible to achieve. Since in many cases these vectors are used to represent the input to NLP systems that carry out the tasks on which they are applied, this misrepresentation is propagated through them early on and is hardly recoverable. This is the main issue addressed in this thesis: to develop semantic representation models that are aware of the multiple meanings of a word and consequently learn representations for each of them.

To tackle this task, we build on previous work on recent word representation models that learn automatically from text. However, as mentioned above, unannotated textual data may not be the most adequate source of information from which to derive knowledge about the different meanings, or senses, of a word, and producing annotations for the large amounts of text that such models consume is usually unfeasible or unreliable. For this reason we propose to engage an extra source of information where this missing knowledge is readily available: linguistic resources such as lexica. Computational linguists have built and curated a trove of resources that store formally structured knowledge in machine-readable format: thesauri (Borin, Allwood and de Melo 2014a), knowledge bases (Miller 1995), and lexica (Gellerstam 1999), among others, which have helped to develop countless NLP applications. In this work, we show that it is possible to combine the structured information contained in a lexicon with the running text from which neural models traditionally learn semantic representations, and to derive word sense representations from those separate sources of data.

All of the models presented in this thesis showcase their capabilities in the Swedish language; not in vain, this work was developed at Språkbanken (the Swedish Language Bank), a unit at the Department of Swedish of the University of Gothenburg which devotes a large part of its work in computational linguistics to developing resources for the Swedish language. Access to said resources and expert advice is granted in such an environment, and it would be unreasonable not to take advantage from it. However, there is a conscious choice behind the development of these models in order for them to not be dependent on any specific language: the models we present here do not make any language-specific assumptions and so they are able to learn from any language, provided that they are fed with adequate data. This choice is made in the hope that our contribution is maximally useful to the international community in which it has been nurtured.

1.2 Research questions

This thesis work is mainly concerned with the creation of word sense semantic representations. In particular, we are interested in applying neural network models to the task of automatically learning those representations as their internal parameters. (See chapter 3 for detailed descriptions of neural network architectures for this purpose.) We frame this task as an improvement over recent models dedicated to learning word representations, or word embeddings; a specific characteristic of recent word embedding models that has contributed to their successful implementation in NLP systems, and that we would like to conserve in our models, is their computational efficiency in dealing with large amounts of textual data to achieve high quality representations.

As such, we can formulate our first research question as follows:

Q1. Can embedding models be adapted to successfully transition from representing words to providing separate representations for different senses of a word, while keeping their semantic representation capabilities and computational efficiency?

Operationalizing this question requires us to test two characteristics of any model proposed in this frame of reference: (1) the quality of the word sense embeddings it learns, and (2) the computational overhead it would add relative to a comparable word embedding model. Evaluating the quality of embeddings is a complex task which, on account of their relative novelty, still lacks an evaluation standard accepted by the community. Usually, test applications like word similarity are designed to assess the intrinsic quality of embeddings, while their extrinsic utility is tested on downstream applications like sentiment analysis. (See a detailed discussion about evaluation techniques in chapter 4.) The computational efficiency of embedding models can be measured, for example, as the amount of time they require to be trained under controlled conditions; training times of different models can then be compared to give an assessment of their relative efficiency.

As mentioned before, it is our plan to include linguistic resources in these models. Specifically, our aim is to take advantage of knowledge about word senses encoded in lexica through inventories of senses per word and lexical and semantic relations between word senses to help steer the learning process of our models towards representations of word senses that accurately portray lexicographic definitions of senses. Thus, an addendum to question Q1 could be:

Q2. Can the knowledge manually encoded in lexicographic resources be leveraged to help improve representations learned by word sense embedding models trained on a corpus?

The quality of embeddings emerges again as the core of this question, which makes it necessary to assess the intrinsic and/or extrinsic performance of different models so that their respective capabilities can be compared against each other, as explained above. Ideally, we should be able to measure the differences in performance between models that do not use lexicographic knowledge as part of their training data and those that do so. The formulation of question Q2 deliberately contrasts the different natures of lexicographic and corpus data: while the latter consists mainly of unstructured text in which the main assets are repetition and words acting as context for other words (see chapter 3), the former's strength lies in carefully crafted structure and annotation rather than in the amount of data and its distribution. Part of the answer to this question must thus examine the success in integrating the two types of data, especially since we deal with models designed to work mainly with the latter type. In order to do this, we need to be able to tell apart the influence each type of data has on trained word sense embeddings and judge whether these influences contribute more or less equally to create sound meaning representations. (See chapter 8 for examples of experiments addressing this specific issue.)

Finally, we would also like to measure the value added by word sense representation models to the community. One way to do so is to test the performance of embeddings on downstream applications: for example, if precision scores on a semantic frame prediction task (chapter 8) rise when using word sense embeddings as features over using word embeddings while all other conditions remain equal, we can say that word sense embeddings do add value to this task. Counting with word sense-dedicated representations also might enable the use of embeddings as features in tasks like word sense disambiguation or induction, where it is not as straightforward to apply them when the object represented are word forms. Using downstream applications to evaluate models provides us then with a measure of added value. However, we could look at those same lexical resources we propose to use for training our models as objects that could also benefit from this work. Indeed, such resources are labor-intensive since they require human input to be built, so any means of automation would simplify their maintenance and expansion. Thus, we ask:

Q3. How well suited are word sense embeddings to improve lexicographic resources?

Answering this question requires us to specify what *improving* means for a specific resource. There exist several aspects of any lexicographic resource that might be improved, like coverage or correctness of existing content. For example, in chapter 9 we evaluate the capabilities of word sense embeddings to suggest new entries for a lexicon by selecting instances from a corpus that might contain word senses not included in the lexicon, or in chapter 8 we try to classify word senses into semantic frames for the Swedish FrameNet (Friberg Heppin and Gronostaj Toporowska 2012). Designing such tasks as evaluation strategies for our models allows us to measure their potential impact on resource building.

In summary, questions Q1, Q2, and Q3 encapsulate the goals of this thesis work, namely, to transition from word embedding models to word sense embedding models, to enroll the help of lexicographic resources in this endeavor, and to measure the capability of word sense embeddings to improve those resources. These questions also define the criteria by which we can measure the achievement of said goals: by assessing and comparing the intrinsic and extrinsic quality of embeddings learned by different models, along with these models' computational efficiency; by testing the integration and influence of different types of data sources that inform our models; and by posing evaluation strategies that let us identify the potential applications that word sense embeddings have.

1.3 Contributions

The main contributions of this thesis are presented through a compilation of published articles in part II. These comprise different models dedicated to automatically learning word sense semantic representations from corpora and lexica, along with evaluation methodologies intended to help determine the models' strengths and weaknesses. The different models developed for this work are intended to explore the possibilities at our disposal for distilling useful linguistic knowledge about word senses from existing resources and combining it with distributional data from corpora.

In order to achieve that, we work with a spectrum of the type of data used to train our models that ranges from pure text from a corpus to lexical-semantic relations from a lexicon. Training models on different points on this spectrum and assessing their performance on different tasks allows us to (1) determine the suitability of each type of data for the task of learning semantic representations for word senses, and (2) control the influence of each type of data on the resulting representations in order to establish the optimal proportion of each of them in terms of performance. In particular, we present the following models:

1. In article 1 (Nieto Piña and Johansson 2015), contained in chapter 6, a model is introduced that learns word sense embeddings solely from a corpus with the exception that the number of senses for any given word is derived from a lexicon. This model is based on Skip-gram (Mikolov et al. 2013a), a word embedding model known for its computational efficiency; our modifications allow it to learn several representations per word, while only introducing a 10% computational overhead. Throughout this study we observe that such an

approach is able to distinguish different meanings associated with the same word form and that these meanings correlate better with *word usage* rather than lexicographic senses; i.e., the *word senses* of a word discovered by the model are differentiated necessarily by the different contexts in which this word is used, since this is the only information available to the model.

2. The other end of the spectrum is explored in article 2 (Nieto Piña and Johansson 2016) (chapter 7), where a model is trained to learn word sense embeddings from data generated from a lexicon. This model is presented along with a word sense disambiguation method based on the word sense embeddings learned, in an effort to showcase their utility on this task. This serves to show that, even if the method does not reach state-of-the-art levels of performance, the type of model used, initially designed to be trained on corpora, is effectively able to extract useful information about separation between senses of a word from lexicographic information. Furthermore, the disambiguation method presented is orders of magnitude faster than other graph-based methods.

3. Having studied the prospects offered by each type of data, the middle ground of the spectrum is examined in article 3 (Nieto Piña and Johansson 2017), found in chapter 8. A new embedding model is presented here which is able to learn word sense representations jointly from textual and lexicographic data in adjustable proportions. Its aim is to put to work the lessons learned while designing the two previous models by trying to compensate one model's shortcomings with the other's advantages. As it is possible to control the proportion of each type of data that feeds the model, we are able to find a balance between them and measure their impact on the results and we show with our evaluation strategy that what can be considered the ideal proportion for one specific downstream task may not be optimal for a different one.

In addition to these models, we provide an extensive study of different evaluation strategies that can be used to measure the quality of word sense embeddings. This has proven to be a non-trivial endeavor for different reasons. (See chapter 4 for a detailed discussion on the topic.) On one hand, the very definition of *meaning of a word* is a contested issue (Kilgarriff 1997; Lenci 2008), which in turns makes it difficult to establish criteria for evaluating the quality of a semantic representation. While a number of tasks like word similarity have been adopted as a standard to test embedding models, these are usually geared towards word embeddings and are not straightforward to adapt to the case of word sense embeddings. (See, for instance, the approach taken to test word similarity by Neelakantan et al. 2014.) On the other hand, a more pragmatic obstacle is the lack of resources to be applied in evaluation. Many evaluation approaches involve comparing the results obtained by the system being evaluated against a standard which would be manually annotated or checked by humans. For example, in a word sense disambiguation task used for evaluation, the target words need the correct disambiguation to be provided in order to check the quality of the automatic disambiguation results. Such resources are not always readily available, especially for languages outside a small set of well resourced ones like English; this is the case for the Swedish language, which we used to build our semantic representations.

To counter these issues, we design evaluation plans that fit the model onto which they are applied in terms of providing an accurate assessment of its characteristics, in the hope that they may serve others in the community when presented with similar challenges. A key point in achieving this is a complete coverage of a model's attributes in the evaluation, so whenever possible we perform several assessment tasks on each model that are used to inspect its different aspects. Qualitative assessments are used to provide intuitive understanding of a model's capabilities, and quantitative evaluation is performed through different tasks that measure a model's performance in disparate scenarios; such tasks include comparison of sets of related terms in a vector space versus a lexicon, similarity tests, word sense disambiguation, or sentiment analysis.

Additionally, as one of our goals is to study the viability of automatically learned semantic representations for improvement of resources, we provide a framework for assessing word sense embeddings in this task. For this purpose, a system is developed in article 4 (Nieto Piña and Johansson 2018), contained in chapter 9, that extracts instances from a corpus containing word senses with a high probability of not being listed in a lexicon, as a way of providing suggestions to lexicographers for expansion of the lexicon and partially automating their work. Furthermore, we show in chapter 8 the capacity of word sense embeddings to predict membership of a term in a semantic frame of the Swedish FrameNet (Friberg Heppin and Gronostaj Toporowska 2012), in such a manner that could be applied to add new entries to the knowledge base. In article 5 (Borin, Nieto Piña and Johansson 2015) (chapter 10) we test the performance of different types of semantic representations of senses on the task of linking entries in a modern lexicon with entries in an older thesaurus, which serves to facilitate access and manipulation of an outdated resource, as well as to pave the way for its potential expansion and modernization with new entries from the contemporary lexicon.

Finally, a word sense disambiguation mechanism based on word sense embeddings that was developed during this thesis work has been incorporated to Sparv[1] (Borin et al. 2016), Språkbanken's annotation tool. This disambiguation mechanism was introduced by Johansson and Nieto Piña (2015) and has been adapted for the evaluation of the models described in chapters 7 and 8 of this text. As part of Sparv's annotation pipeline, the mechanism is used to automatically disambiguate and label instances of Swedish words in an input corpus, using a sense inventory obtained from the lexicon SALDO (Borin, Forsberg and Lönngren 2013).

1.4 Thesis structure

The rest of the text is structured as follows. Part I, which includes the current introductory chapter, sets the context for the work and gives background and detailed descriptions of our models' main components. Chapter 2 formalizes our working definition of word senses and discusses the types of resources on which our models are trained: lexica and corpora; chapter 3 introduces the distributional hypothesis, then discusses distributional models for obtaining word and word sense embeddings, as well as options available to introduce lexicographic knowledge into them; chapter 4 reviews common evaluation methods used on embedding models and describes the evaluation strategies we applied on our models; finally, chapter 5 closes part I with conclusions reached in this thesis.

Part II consists of a compilation of articles published during the development of this thesis which contain the models and their applications that constitute the core of the thesis work. Chapter 6 presents an unsupervised model to learn word sense embeddings from corpora; as a counterpoint, chapter 7 introduces a model for learning word sense embeddings only from a lexicon which are applied to perform word sense disambiguation; chapter 8 describes a joint approach to learning word sense embeddings from both a corpus and a lexicon; chapter 9 explores the potential of linking word sense embeddings with lexicon entries in order to find word senses not listed in the lexicon; and chapter 10 investigates the applicability of word sense representations to link entries of two different lexical resources in order to facilitate access to and modernize outdated resources.

[1]`https://spraakbanken.gu.se/eng/research/infrastructure/sparv`

2 Linguistic resources

Linguistic resources provide us with the data needed to train and test our representation models. The kinds of data resources that we consider under this term are compiled (and possibly annotated) by computational linguists to contain language samples and lexicographic inventories relating to one or more languages; in particular, we are interested in lexica and corpora. Lexica provide inventories of a language's vocabulary, while corpora contain samples of written text intended to facilitate the conduction of linguistic analysis. In our models, we take advantage of that information to try and obtain semantic representations that are derived automatically from those resources; also, in some instances, the performance of these models is assessed with help of resources such as annotated corpora. (See chapter 3 for a description of different ways in which representational models learn from these data resources, and chapter 4 for an account of how models are evaluated using annotated data.)

In this chapter we also offer a description of the concept of word sense as used in this thesis work. Word senses are the target of our research as the linguistic unit for which we aim to create semantic representations. By processing the explicit and implicit information about word senses present in linguistic resources, our models are able to learn to represent them in a vector space, providing us with mathematically manipulable semantic objects easily handled by NLP applications. Such applications that need to process meaning, like machine translation, sentiment analysis, or named entity recognition, among many others, rely on using accurate semantic representations of the texts onto which they are applied. The linguistic unit most commonly represented, however, is the word form; since such representations are commonly obtained from corpora, composed of text documents in a more or less unprocessed form, it is straightforward for this to be the case. Nevertheless, employing one representation per word form may conflate several meanings in the cases of words that have more than one, which has the potential to damage the modeling power of the semantic vector space (Neelakantan et al. 2014; Yaghoobzadeh and Schütze 2016) and the performance of systems that work with semantic representations (Li and Jurafsky 2015; Pilehvar and Collier 2016). Our goal is to study ways in which word sense representations can be derived from corpora and lexica in order to obtain a more fine-grained representation of meaning that is oriented towards representing distinct word senses rather than word forms.

2.1 Word senses

The larger part of this thesis work concerns the automatic creation of suitable representations for word senses. While working at the word form level, any word w is given a single representation v; in many cases, such representations are vectors in a real-valued multidimensional space, so that $v \in \mathbb{R}^N$. (See chapter 3 for a discussion on semantic representations.) However, linguistic phenomena like polysemy, by which a single word form is assigned more than one meaning, raise an issue with this approach to semantic representation. For example, if the noun *rock* were represented by a vector v, both of its two main meanings ('a mineral material' and 'a type of music') would share one representation. Conflation of the different senses of a word might impact negatively the performance and quality of certain applications that use such representations (Li and Jurafsky 2015; Yaghoobzadeh and Schütze 2016). Our aim in these terms, then, is to devise ways in which a word w with multiple senses like *rock* can attain a separate representation v_i, $i \in 1, 2, \ldots, n$, for each of its n senses.

When a word can take several distinct meanings, through phenomena such as polysemy or homonymy, each of those meanings is known as a *word sense*. E.g., *a small rodent* is one sense of *mouse*, but another meaning of the word is *a computer peripheral used to move a pointer on a screen*. Given that there is no explicit indication of the intended meaning of an instance of a polysemous word, *word sense disambiguation* has to be performed on it in order to choose the word sense relevant for that occurrence and thus clarify its meaning. Such a process is informed by the context in which that instance is found; i.e., the meaning contributed by words accompanying the ambiguous word in a sentence, a document, or a collection of documents.

Context is the main source of information for the task of disambiguating an instance of a word, both for humans and machines. In order to prepare an inventory of word meanings for a lexicon, a lexicographer needs to inspect the context of instances of each word in a corpus in order to categorize those instances into separate word senses.

Similarly, in machine-based approaches (Navigli 2009) to automatic discrimination of word senses (for the purpose of word sense disambiguation or induction, for example,) data-driven techniques are usually deployed to compare contexts of different instances of a word and classify them into word senses.

The result of any of these disambiguation processes, performed by humans or machines, relies on the assumption that the corpus employed contains a more or less faithful representation of the language. This is so because any process of word sense discovery or disambiguation based on linguistic evidence from a corpus will be affected by the sense inventory found in the corpus: whether one particular word sense will result from such a process is subject to whether the corpus contains enough evidence for it. Especially in machine-based methods, where human insights into language are more difficult to operationalize, the dependence on corpus evidence to track the different meanings of a word tends to shift the concept of word sense towards word usage: automatic disambiguation or discovery of word senses solely based on corpus data gravitates towards identifying differences in usage of a word that may differ from lexicographic word sense definitions of that same word. For example, consider the noun *mushroom* to be defined in a coarse-grained lexicon as having a single meaning: *a fungal growth in the shape of a domed cap on a stalk, with gills on the underside of the cap;* it is conceivable that a process of automatic discovery of the word senses of *mushroom* based on corpus evidence could conclude with the word having two senses derived from two distinct contexts in which the word is commonly used: one pertaining to biology, and another to culinary subjects. This disparity can potentially be addressed by making lexicographic resources, such as lexica, available to the machine-based process in a way that the lexicographic descriptions of senses guide the sense discovery process.

Related to this, the granularity of word senses needs to be determined as a conscious choice. In the example for *mushroom* above, the lexicographers in charge of building that lexicon would have chosen it to be coarse-grained; it is entirely reasonable that another, more fine-grained lexicon would separate the biological and culinary meanings of *mushroom.* As an example of such discrepancies, Palmer, Dang and Fellbaum (2007) studied the differences in word sense granularity between the sense inventories in the Senseval-1 (Kilgarriff and Rosenzweig 2000) and Senseval-2 (Edmonds and Cotton 2001) tasks for automatic word sense disambiguation (WSD): Senseval-1 obtains its sense inventory from the Hector lexicon (Atkins 1992), which results in the verbs used having 7.79 senses on average; on the other hand, Senseval-2 extracts its English sense inventory from WordNet (Miller 1995) (known for its high granularity,) which gives the verbs used an average of 16.28 senses. Such design decisions need to be taken into account whenever a sense inventory is derived from a lexicon or other resource, since it will determine the behavior of the system that inherits it.

Furthermore, the definition of word sense and its relation to word usage is not free of debate. For example, Kilgarriff (1997) fails to find an operational definition for *word sense* in the context of WSD, and concludes that a fixed, general-purpose inventory of senses is not indicated for use in NLP applications; rather, word senses would only be defined as they are needed by the application of interest and, thus, they should emerge as abstractions of clusters of instances of word usage. That is, it is his view that word senses exist only as clusters of instances of a word, and that such clusters are only defined on a *need-to-exist* basis dictated by the task that calls for the clustering action. Thus, issues of completeness or granularity are resolved by stating a set of task-specific clustering guidelines. This implies that there cannot be a task-independent sense inventory.

While such ideas merit discussion, we intend to distance this work from theoretical debates about the nature of word meaning. The question that guides this work is whether computational models for meaning representation are able to capture different senses of a word and, in particular, whether lexica can help in such a task. Thus, for the purpose of this thesis, we consider a word sense for any given word when it is defined as such in the lexicon. As a result, our computational models usually work with a fixed, discrete, and finite word sense inventory that originates in the lexicon. In this context, we do not consider this a shortcoming since the lexicon's inventory is used as a gold standard for our models' testing or training: one form of model evaluation that we apply is to measure how well a model is able to represent the inventory of senses found in the lexicon (chapter 6); in other cases, the lexicographic sense inventory is used to steer the word sense learning process of the model (chapter 8). It is thus acknowledged that the lexicon used will have an influence on the results; this is not necessarily a negative effect since our goal is not to obtain the ideal sense inventory for a particular task but rather, given a sense inventory, find high-quality representations for it.

2.2 Lexica

A lexicon is the collection of lexical items, represented by lemmas, of a language. It is intended to function as a complete inventory of a language's main vocabulary and it can be complemented by additional information about its entries, such as their morphological characteristics or a structure of links between entries that specify relations between them (e.g., *synonymy* relation between word senses which share the same meaning, such as exists between *movie* and *film*; or *hypernymy-hyponymy* relation between a more general and a more specific term, such as *plant* and *ivy*.)

Opposed to traditional lexical compilations, like dictionaries, which are built for human consumption, modern lexica are intended for use in NLP processes, and store relevant lexical information in machine-readable formats that can effectively be used in such processes. For example, the meanings of entries can be encoded by establishing links between them (which exploit semantic relations as mentioned above; see also WordNet below) so that entries are defined in function of other entries; e.g., *car* is a *hyponym* of *vehicle*, and *tire*, *engine*, and *chassis* are all related to *car* as being *parts of* it. Entries in a lexicon can also be decomposed into primitive concepts that clarify their meaning and allow to relate different entries which share the same or a related meaning. For example, in a frame semantics approach to building electronic lexical resources, word meaning is defined by assigning words to semantic classes, or *frames*; in FrameNet (Baker, Fillmore and Lowe 1998), one such resource for English, *car* belongs to the frame VEHICLE, and *engine*, *trunk*, and *seatbelt* belong to the frame VEHICLE_SUBPART. These approaches to structuring lexical information are related to knowledge bases, or ontologies, which are used to encode human general knowledge or domain-specific information for processing by computer systems by structuring information via classes and subclasses linked by relations between them. For an example of a general knowledge ontology, see Google's Knowledge Graph (Singhal 2012).

Lexical resources also differ in the data and methods used for compiling them (Hazman, El-Beltagy and Rafea 2011): Obtaining lexical information from unstructured (corpora) or structured (databases) data, via a manual process by lexicographers or an automatic method that leverages statistics and patterns in the source data, or a semi-automatic method that filters the source data for further processing by humans.

Lexical resources have been the object of study and development on the field of Language Technology since its early days (Reichert, Olney and

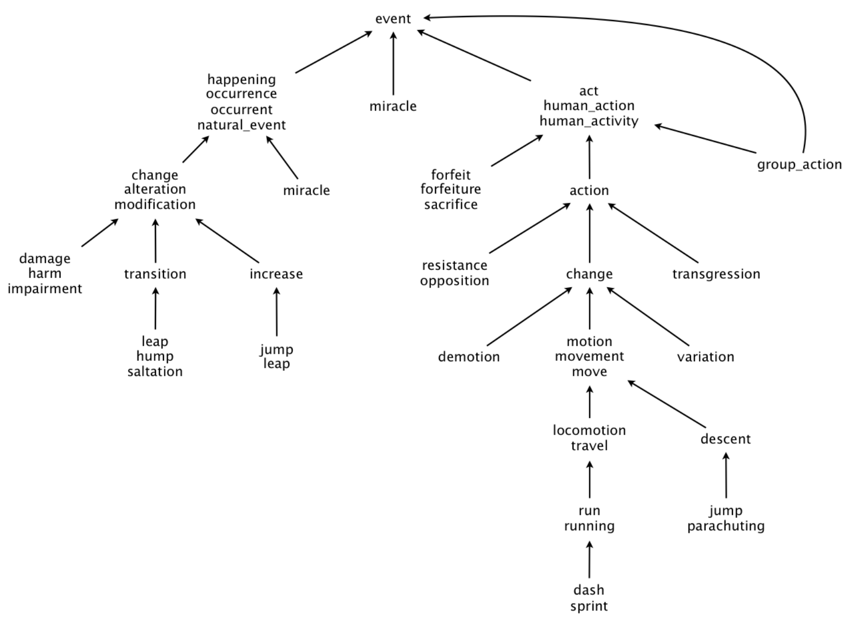


Figure 2.1: A sample of WordNet's synset graph.

Paris 1969; Smith and Maxwell 1973) with the goal of creating machine-readable resources that can be used to incorporate lexical knowledge into NLP systems. Computerized resources have the advantage of being able to store and process large amounts of information, which allows them to be enriched with additional information at a lower cost than their traditional, paper-based counterparts. Abstract data types in Computer Science, such as graphs, also allow greater flexibility in how the data is stored and used. These assets have been taken advantage of to create large, wide-ranging lexical resources which contain substantial quantities of information ready to be used for language processing. An example of this is WordNet (Miller 1995), an English lexical database built as a graph connecting groups of synonyms (*synsets*) by means of semantic and lexical relations, such as hypernymy-homonymy. (See figure 2.1 for a sample of WordNet's graph around the synset EVENT; relations in this graph are indicated by directed arrows signaling the origin as a hypernym of the destination.)

For our work on Swedish word sense representation, we have made use of such a resource for the Swedish language: SALDO (Borin, Forsberg and Lönngren 2013).

2.2.1 A Swedish lexicon: SALDO

SALDO is a lexical-semantic network which, similarly to WordNet, represents concepts in a graph's nodes and connects them using a variety of lexical-semantic relations. The principles followed to build this network, however, are different from WordNet's central concept of synonym sets. SALDO is organized as a hierarchy. Any of its entries, has one or several *semantic descriptors* of which one is unique and mandatory: the primary descriptor. Semantic descriptors are also entries in the lexicon, so any entry has at least one semantic descriptor, but can also be a semantic descriptor of other entries. The characteristics of the relation formed between an entry and one of its semantic descriptors establishes SALDO's hierarchical structure.

In the case of the *primary descriptor* (PD), an entry must be a *semantic neighbor* of, and more *central* than another in order to be its PD. Two entries in the lexicon are semantic neighbors when there exists a semantic relation between them, such as synonymy or hyponymy. Centrality is defined in terms of different criteria, such as frequency (words with higher frequency are more central than words with lower frequency), stylistic value (stylistically neutral words are more central than stylistically marked ones), derivation (words with lower derivational complexity are more central than those with higher complexity), and type of relation in the case of asymmetrical relations (e.g., a hypernym is more central than a hyponym). In practice, most PDs are synonyms or hypernyms of the entry they describe.

The stipulation by which any entry in SALDO must have one and only one PD (but can potentially be PD of several other less central, semantically related entries) confers its underlying structure a tree architecture. This also implies that there must be a root node, called PRIM, at the top of the PD hierarchy; this is an artificial entry created solely for this purpose, and bears no linguistic relation to the entries of which it is a PD. (See a portion of SALDO's PD tree around the term *music* in figure 2.2; relations in the tree are indicated by directed arrows signaling the PD of the arrow's origin.)

Other semantic descriptors are *secondary descriptors* (SD). An entry can have more than one SD, and their chief purpose is to assist in describing the entry's meaning, especially in the case of its PD not being a synonym. (Observe that in the case that the PD is a synonym, its semantic description is rather complete.) There are no restrictions on the type of relation that must exist between an entry and its SDs.

Each entry in SALDO is a sense of a word. A polysemous word, for

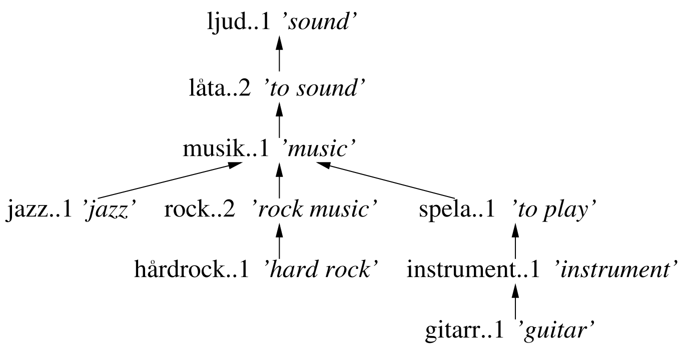


Figure 2.2: A sample of SALDO's primary descriptor tree.

instance, will have one entry for each sense; e.g., the Swedish word *rock* is described as having two meanings: 'coat' and 'rock music', so there are two entries, *rock$_1$* and *rock$_2$*, one for each sense of *rock*. Due to the principles followed for distinguishing senses to be included in this resource, SALDO's sense granularity is coarser than that of WordNet. As described in its original formulation by Borin, Forsberg and Lönngren (2013), the average number of senses for base forms in SALDO is 1.1 and approximately 7% of all base forms are polysemous, with the most polysemous one having 10 senses; meanwhile in WordNet 17% of base form-part of speech combinations are polysemous, with the most polysemous one having 59 senses. Furthermore, entries in SALDO are not restricted to single-word elements, but it also includes multi-word expressions. In addition to word sense information, entries contain information about their part-of-speech and their inflectional pattern.

2.3 Corpora

A corpus is a collection of texts which, in the field of corpus linguistics, are used to perform different kinds of linguistic analysis: gather statistics, retrieve occurrences and linguistic evidence, or conduct comparative studies, among others. Modern corpora are stored in computer-readable form, so that tools developed by computational linguists can be applied onto them. A corpus can have a general aim, by collecting texts from different types of sources, styles, and authors with the aim of providing a representative sample of the language (or languages) covered; or it can have a narrow focus to enable the study of a specific aspect of language, by sampling only texts relevant to the subject: a historical period, a specific language variety, or a particular form of online communication, for example. In the cases where corpora are used to train language models or semantic representations, as is the case in several models presented in this thesis, the selection of texts has an important influence over the resulting models. As was discussed earlier in this chapter, such models learn the semantics of the language by analyzing its usage in text; it can be inferred from this that the models trained on a corpus will reflect the language contained in it and, thus, this is a factor to be taken into account when choosing a corpus for these purposes. In our work, we have striven towards representing language as used in a wide range of genres, topics, registers, and styles in contemporary Swedish; to achieve this, we compiled a training corpus from different sources in order to account for the desired variation (see below).

Besides differences in the language type and topic covered, corpora may differ in a number of aspects that are defined when a corpus is compiled, such as the size of included texts and the proportion of different text types, or what annotation and metadata are to be added onto the raw text, among others. A type of annotation of special interest for our work is word-sense annotation, by which all or part of the words or lemmas contained in a corpus are annotated with the sense corresponding to each instance, according to a pre-specified word sense inventory which can be extracted from a lexicon, or related annotations such as semantic frames. Such corpora, while laborious to produce due to the amount of human input needed, have an added value for training and evaluating models such as are presented in this thesis, whose main goal is to identify and represent word senses. For an example of a contemporaneous corpus annotation effort which combines human input with help of language technology tools, see the descriptions provided by Johansson et al. (2016) for annotating a Swedish corpus with word senses.

The use of the Internet by an ever increasing part of the population to communicate, share knowledge and data, and access news and entertainment in the last decades generates an extremely large amount of written language in the form of articles, blog posts, chat logs, product reviews among many others. In the period from 1986 to 2007, Hilbert and López (2011) estimated the growing global storage capacity at 2.6, 15.8, 54.5, and 295 exabytes (1 EB equals 10^{18} bytes) in 1986, 1993, 2000, and 2007, respectively; according to this same study, the proportion of these amounts of data stored in digital versus analog platforms grew from 25% in 2000 to 94% in 2007. Even if most of this vast amount of data is not textual (according to a Cisco (2017) white paper, 73% of global IP traffic during 2016 was video traffic), the rapid growth and reach of digital data also affects this medium. Large collections of text available online have proven to be an invaluable source of data not only for the study of language itself, but for analyzing text-generating users' behavior from sociological and market points of view. The academic and industrial value of this data has in turn motivated the creation and refinement of language analysis tools able to leverage it. In summary, there currently exists a thriving ecosystem revolving around corpora that enables acquisition of insight from primary language data at an unprecedented level in terms of quantity, availability, and analytic capacity.

2.3.1 Swedish corpora used in this thesis

For those models presented in this thesis that need a corpus to be trained on, a Swedish language corpus is used consisting of approximately 1 billion words.

This corpus was compiled by aggregating a number of corpora[2] featuring different text sources in an attempt to achieve a balanced representation of written Swedish language. It comprises text from social media (corpora *Bloggmix 1998-2013; Twitter mix, August 2013; Swedish Wikipedia, August 2013*), print and online newspaper texts (*DN 1987; GP 1994, 2001-2012; Press 65, 76, 95-98*), texts from different science and popular science publications (*Forskning och framsteg; Läkartidningen 1996-2005; Smittskydd; Academic texts - Social science*), fiction literature (*Bonniersromaner I, II; SUC novels*), and corpora with mixed contents (*SUC 3; Parole*).

Furthermore, the texts in the corpus were tokenized, lemmatized, and POS-tagged using Språkbanken's Korp NLP pipeline (Borin, Forsberg and Roxendal 2012). The tokenizer and lemmatizer used are tools developed specifically for this pipeline, while the POS-tagger is HunPos (Halácsy, Kornai and Oravecz 2007). Automatic segmentation of compounds was also applied on the texts to split compound words into their components when a compound word's lemma was not found in SALDO (see section 2.2.1).

Besides the main corpus described above that was used to train our models, we used a number of additional corpora in some of the evaluation tasks applied to test the performance of models. In particular, these are corpora that include sense annotations for all or part of their contents that we used for the purpose of solving word sense disambiguation (WSD) tasks.

[2]Available for download at `https://spraakbanken.gu.se/eng/resources`.

Two of these corpora were compiled collecting sentences used as glossing to illustrate the use of Swedish word senses contained in the Swedish FrameNet (Friberg Heppin and Gronostaj Toporowska 2012) and SALDO (Borin, Forsberg and Lönngren 2013). These sentences have been selected by lexicographers as examples of word sense usage for entries in those resources and, thus, contain a word annotated with its sense each. In the case of the Swedish FrameNet glosses, a total of 1 197 sentences were annotated in terms of its semantic frames (for which a mapping to SALDO senses exist); the SALDO glosses correspond to 1 168 sentences and are annotated with SALDO senses.

Another sense-annotated corpus was compiled with sentences from the Swedish Senseval-2 task (Kokkinakis, Järborg and Cederholm 2001). This collection contains 8 237 sentences, originally divided into two subsets for training and testing. Each sentence contains an ambiguous word, from a list of 40 possible words, annotated with its correct sense. In this case, the word sense inventory used originally was obtained from the Gothenburg Lexical Database/Semantic Database (Allén 1981), but a manual mapping to SALDO word senses was used to homogenize it with the rest of corpora (Nieto Piña and Johansson 2016); due to the differences between sense inventories, the number of ambiguous words changed from 40 to 33.

Finally, the mixed-genre, sense-annotated corpus from the Koala annotation project (Johansson et al. 2016) was used. This corpus comprises seven sub-corpora containing Swedish texts from different genres: blogs, novels, Wikipedia articles, European Parliament proceedings, political news, newsletters from a government agency, and government press releases. The version we used (since the annotation project was still ongoing at the time) was composed of 11 167 sentences containing one sense-annotated word each, using the sense inventory from SALDO. The inter-annotator agreement for two annotators on this corpus is given by a κ coefficient (Cohen 1960) of 0.70 and an estimated agreement probability of 0.90.

3 Distributional representations

Distributional representation models allow us to generate representations for words and other linguistic units of meaning. A distributional representation is a collection of features that identify the meaning of a linguistic unit, such as a word, in terms of its distributional properties; i.e., the meaning of a linguistic unit is represented as a function of the contexts in which it tends to appear. Distributional representations are derived from word co-occurrence statistics obtained from text, either directly from counting co-occurrences, or indirectly through learning models that automatically analyze and transform such statistics (Turian, Ratinov and Bengio 2010; Levy and Goldberg 2014a). The shape that distributional representations usually take nowadays is that of high-dimensional, real-valued, dense vectors called *distributed* representations (Hinton et al. 1984) or *word embeddings* which are computationally efficient for use in NLP systems. When derived directly from co-occurrence counts, which produce sparse vectors, dense representations are obtained by means of dimensionality reduction techniques.

The kind of context used to derive such representations influences the semantics they portray. For example, when larger contexts such as whole documents are used, the semantics represented tend to be topical. Thus, related words to any given one will be *topically* similar, such as *concert* and *guitar*. On the other hand, when only words in close proximity to the target are considered as context, the represented semantics tend to be substitutional. In this case, related words to any given one will be *functionally* similar in such a way that one can be substituted by the other in a sentence, such as *spaghetti* and *cannelloni* (Bansal, Gimpel and Livescu 2014; Levy and Goldberg 2014b; Melamud et al. 2016).

While dense embeddings derived from distributional data occupy much of the research effort into semantic representations, these are not the only means to representing the meaning of words and other linguistic units. Symbolic representations are a counterpart to this approach: in a symbolic paradigm, the semantic unit is represented by a discrete atomic symbol such as a string of characters or an arbitrary sequence of numbers. Symbolic representations hold the advantage of being easily interpretable by humans, since knowing the correspondence between symbol and object allow us to understand the representation; they also facilitate representing composition, so that a sequence of objects like words can be represented as a sequence of symbols, for example. On the other hand, distributed representations, in the context of massive parallel computation brought by very large neural networks, or deep learning (LeCun, Bengio and Hinton 2015; Schmidhuber 2015), provide an efficient medium to store and manipulate meaning through large scale computation. This kind of representation also enables the notion of graded similarity: since real-valued features of the represented object are distributed across the vector's dimension, comparison of these features among different vectors is possible. Symbolic representations do not allow such comparison, since each symbol is equally different from all other symbols.

In the rest of this chapter, we discuss the distributional hypothesis that is at the base of distributional models. After a brief example of a classic model that illustrates how this hypothesis can be applied to generate semantic representations that are apt to be used in computational models, we explore current models used to automatically generate distributional representations for words and word senses from large collections of text. We also consider different approaches to use linguistic resources such as lexica as a source of data to train such models.

3.1 The distributional hypothesis

The *distributional hypothesis* (Harris 1954) states that

> the degree of similarity between lexical objects A and B is a function of the degree of similarity between the environments in which A and B appear.

In other words: if A and B are two words which tend to appear in the same contexts, they will have similar meanings. Or, as summarized by Firth (1957), "You shall know a word by the company it keeps."

This hypothesis brings forth the concept of distributional semantics, which attends to the study of word meaning based on context. Under this assumption, the meaning of words can be studied, at least partly (see Lenci 2008), by analyzing their distributional characteristics; i.e., the environments or contexts in which they appear. As an intuitive example, consider the sentence “After the mountain pass, we made our way down the rocky *kambotke* which run among trees and shrubs.” While this might be our first encounter with the made-up word *kambotke*, it could be inferred from the context that it is likely a feature of the landscape, one which possibly can be transited not unlike a dry river bed or other naturally occurring track.

The original framework in which this hypothesis was formulated was that of *distributional analysis*, which was intended to provide a formal scientific methodology for the study of linguistics in general, from the phonological to the semantic levels. With regards to word meaning, the distributional hypothesis helps explaining it in terms of *differences* (Sahlgren 2008): by providing an instrument to measure the distributional differences between words, a distributional model represents the meaning of one word in terms of how different it is from other words. Note that this implies that an isolated distributional representation of any given linguistic unit is not interpretable by itself, but it is rather in comparison to other units’ representations, by measuring how different they are, when it acquires significance.

Language Technology has made extensive use of the distributional hypothesis in the last couple of decades. In this recent context, the traditional approach to building *distributional models* was based on computing co-occurrence matrices to be processed for dimensionality reduction. A co-occurrence matrix usually has its rows indexed by words and its columns by contexts. (E.g., words, documents.) Its cells contain either raw co-occurrence counts between word and context, or a derived statistic such as *tf-idf*. Once computed, a co-occurrence matrix’s rows can be used as sparse vector representation of their indexed words, or it can be further processed to reduce the number of dimensions and avoid sparsity for improved computational performance using matrix factorization techniques such as Singular Value Decomposition. In either case, words which usually occur in similar contexts will have similar corresponding vector representations. Hence, vector similarity is an analogue of semantic similarity in this paradigm. Classic examples of this approach are Word Space (Schütze 1993), Hyperspace Analogue to Language (HAL) (Lund and Burgess 1996), Latent Semantic Analysis (LSA) (Landauer and Dumais 1997), or Random Indexing (Sahlgren 2005).

The renewed success of neural networks in different areas of Machine Learning that started at the turn of the century permeated research efforts in statistical language modeling, whose goal is to learn the probability distribution of sequences of words in text. Bengio et al. (2003) proposed such a model using a one-hidden layer neural network to model the probability of the next word w_t given a sequence of n preceding words:

$$P(w_t | w_{t-1}, \ldots, w_{t-n+1})$$

The neural network's hidden layer's weights, trained over samples of natural language texts using error backpropagation, are interpreted as word representations of the vocabulary in this model. These representations are real-valued, dense vectors of distributed features.

Since this model is trained by inspecting a word's context (the preceding sequence of n words), words appearing in similar contexts are expected to receive similar representations. This constitutes an instance of the distributional hypothesis at work. Note how this is coherent with the model's training objective: two closely related words will both have a high probability of continuing a sequence which constitutes a probable context for both. This approach to solving NLP tasks with a neural network which learns representations as a by-product proved to be rather productive. A good example of this is the architecture for multitask learning proposed by Collobert and Weston (2008), by which the model is trained to solve different tasks at the same time, from language modeling to part-of-speech tagging. The resulting representations are naturally able to represent different aspects of written natural language and, thus, adept at generalization.

The improved performance of neural networks' automatically learned representations inspired a shift in the research focus on this area: from using these models to solve one or more specific NLP tasks, the interest veered towards the networks' internal representations of linguistic objects themselves. By understanding and improving the process of training representations, powerful sets of features could be made available for any task susceptible of profiting from their representational ability. For example, Mikolov et al. (2013a) proposed applying simplifications over aspects of earlier neural language models that, while important to the task of language modeling, could be dropped when the main object of interest was the learned representations and not the learning task. In this particular case, this resulted in increased computational efficiency, which allows the model to learn from larger amounts of data in benefit of the learned representations.

Much of the recent work on learning semantic representations applying the distributional hypothesis defines *context* as preceding, succeeding, or surrounding words in a window of predefined size. However, it is worth noting that the concept of context can be defined in a number of ways depending on the type of semantics that need to be captured for the task at hand. For example, whole documents in which a word occurs can be used as its context, causing the learned representations to contain topical information, as is the case of LSA (Landauer and Dumais 1997). An alternative to define context as the words in a sequence surrounding the target is to define it in terms of syntactic relations given by dependency trees as a way of introducing linguistic structure into running text (Padó and Lapata 2007). Translations of the training data into other languages from parallel corpora have also been used as additional context as a way to diminish the ambiguity of polysemic words (Ghanimifard and Johansson 2015). Multimodal context, which includes other types of data such as images in addition to text, has also successfully been used to generate semantic representations (Lazaridou et al. 2015).

3.2 A simple distributional model: bag-of-words

From a language technology point of view, lexical representations provide characterizations of lexical units that are manipulable by computers for use in applications that handle human language. For example, a simple approach for representing text is the *bag-of-words* (BoW) model used to represent a document as a collection, or bag, of words that is useful for analyzing the document's contents and comparing it to other documents. Suppose that we intend to represent the following toy collection of *documents*, one per line:

I've seen it all
I've seen the dark
I've seen the brightness
In one little spark

(Guðmundsdóttir, Sigurðsson and von Trier 2000)

which, after tokenizing, results in the following vocabulary V of size $N = 11$:

$$V = \{\textit{I've, seen, it, all, the, dark, brightness, in, one, little, spark}\}$$

(Contractions have not been tokenized separately here for simplicity.) In the BoW model, after establishing the vocabulary, each document can be represented by a vector of size N in which each dimension corresponds to a word and its value indicates the frequency of that word in the document. In our example, following the ordering of V, our documents d_i, $i \in [1,4]$, could be represented as follows with 11-dimensional vectors:

$$d_1 = [1,1,1,1,0,0,0,0,0,0,0]$$

$$d_2 = [1,1,0,0,1,1,0,0,0,0,0]$$
$$d_3 = [1,1,0,0,1,0,1,0,0,0,0]$$
$$d_4 = [0,0,0,0,0,0,0,1,1,1,1]$$

In a rudimentary manner, such vector representations can be used to determine how similar two documents are by comparing their vectors; for instance, it is easy to tell that d_4 is more different from the other three documents in terms of vocabulary than these three are between them. However, this approach enables more sophisticated analysis by manipulation of the matrix M that represents the whole collection of documents, or term-document matrix:

$$M = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Each column in M represents a document; note that this also gives us representations for the words in V: one per line in their original order from top to bottom. When applied to real collections containing hundreds or thousands of documents, these representations tend to become very sparse as the vocabulary grows (since any document will not contain most words.) In order to address this, matrix factorization procedures like singular-value decomposition (SVD) of the term-document matrix are used in techniques like latent semantic analysis (LSA) (Deerwester et al. 1990) to reduce the dimensionality. The resulting dense vector representations for documents and words can be tested for similarity using simple geometrical operations like obtaining the cosine of the angle between vectors, which facilitates analyses of the data. Representations derived this way have been widely applied in NLP tasks like document summarization (Gong and Liu 2001), sentiment analysis (Maas et al. 2011), or word sense disambiguation (Pino and Eskenazi 2009), among many others.

In fact, while more modern models built on neural networks, some of which are discussed below, learn dense distributed representations directly without transforming a co-occurrence matrix, Levy and Goldberg (2014a) showed that there are indications that the representations learned by such models are asymptotically equivalent to those that could be obtained by performing matrix factorization on a co-occurrence matrix.

3.3 Word embedding models

In the context of the shift of focus from obtaining word vector representations as a by-product of solving an NLP task to making these representations the object of research, a number of models were developed like Skip-gram (Mikolov et al. 2013b), or GloVe (Pennington, Socher and Manning 2014). Under this approach, research efforts focused on identifying which characteristics make word vector representations into competitive linguistic feature sets, and on refining techniques to streamline their training process.

3.3.1 The Skip-gram model

A large part of the models and techniques presented in this thesis work is based on the Skip-gram model (Mikolov et al. 2013b). In turn, Skip-gram builds upon the neural probabilistic language model developed by Bengio et al. (2003) introduced in the previous section. Specifically, Skip-gram tries to improve the computational efficiency of the former model by eliminating architectural complexity in the neural network and introducing simplifications in the operations performed. These improvements make it feasible to train the model on larger corpora, which results in higher quality, more stable representations.

The Skip-gram model is based on learning word vector representations (also known as *word embeddings*) that are adequate for, given a target word w_t, predicting its surrounding context words, $w_{t-c}, \cdots, w_{t+c}$, where c is the size of the context. Its original formulation for this goal as

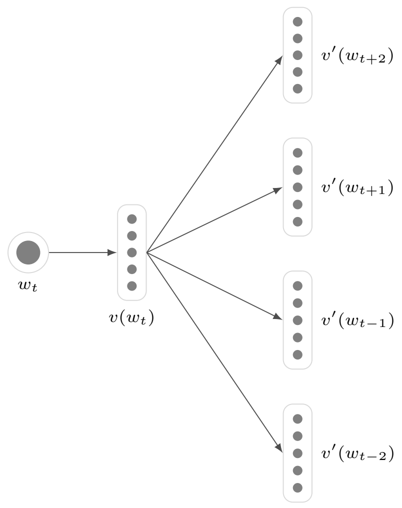


Figure 3.1: Skip-gram training objective: word vectors $v(w_t)$ for words w_t optimized as predictors of context word vectors $v'(w_{t\pm i})$.

training objective function is to, given a sequence of T words, $w_1, \ldots, w_T$, maximize the average log-probability:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{\substack{-c\leq j\leq c\\ j\neq 0}}\log p(w_{t+j}|w_t)$$

In order to implement this training objective, the Skip-gram model trains two separate sets of vector representations, one for target words (or *input*, denoted here $v(w)$ for any given word w) and one for context words (or *output*, denoted here $v'(w)$ for any given word w). The context representations are intended only to be used for the model's internal calculations, while the target representations are usually viewed as the model's trained word representations. With these two sets of vectors, the conditional probability $p(w_c|w_t)$ from the objective function formulated above is defined as follows:

$$p(w_c|w_t) = \frac{e^{v'(w_c)^\intercal v(w_t)}}{\sum_{w=1}^{|V|} e^{v'(w)^\intercal v(w_t)}}$$

where $|V|$ is the number of words in the vocabulary V; note how the denominator would be computationally very costly to calculate, given the usually large size of vocabularies. To address this problem, Skip-gram offers two alternatives to approximating the calculation of the vocabulary: hierarchical softmax and negative sampling.

In the approximation given by *hierarchical softmax*, a computational speed-up proposed by Morin and Bengio (2005), the $|V|$ words in the vocabulary are represented by the leaves of a binary Huffman tree. This structure substitutes the flat softmax output layer of the neural network. For each word $w \in V$ there exists a path from the node of the tree to the corresponding leaf and, at any given node in this path, the probability of going either left or right is calculated. Then, at the leaf node representing w, the probability of w is given as the product of probabilities obtained by tracing this path on the tree. Note that this probability is already normalized, since the probabilities of all leaf nodes in a binary tree must sum up to 1. This saves having to calculate $|V|$ probabilities, reducing this number to an average of $\log_2(|V|)$, the depth of a balanced binary tree.

On the other hand *negative sampling*, a variation of Noise Contrastive Estimation (Gutmann and Hyvärinen 2010; Mnih and Teh 2012), uses noisy samples to train the model to differentiate between genuine (positive) contexts from the training corpus and negative context words sampled from a noise distribution, hence modifying the model's training objective. For each positive context-target training sample from the corpus, k negative samples are generated; the model is then trained to increase the probability of predicting the positive context word for that given target and, conversely, to decrease the probability of predicting any of the k negative samples. Instead of performing $|V|$ calculations for a training iteration, only $k+1$ computations are required, where recommended values of k range between 2 and 20. Negative sampling thus defines a new expression that substitutes $p(w_c|w_t)$ in the original objective function:

$$\log \sigma(v'(w_c)^\intercal v(w_t)) + \sum_{w_j \in \mathrm{NS}_k(w_t)} \log \sigma(-v'(w_j)^\intercal v(w_t))$$

where σ denotes the logistic function, $\sigma(x) = 1/(1+e^{-x})$, and $\mathrm{NS}_k(w_t)$ is the set of k negative samples w_j, $j \in [1,k]$, generated for w_t. The training algorithm for word vectors in Skip-gram with negative sampling using backpropagation with stochastic gradient descent is given in algorithm 1.

In addition to the simplification alternatives sketched above, a *subsampling* of training words is also part of the Skip-gram model. This technique drops words from the training data with a probability proportional to their frequency in the corpus. This reduces the amount of training instances over very frequent words with two implications: the

Algorithm 1: Training of word vectors using Skip-gram with negative sampling.

```
Input: Sequence of words $w_1, \ldots, w_N$, window size $n$, learning rate $\alpha$,
  number of negative words $N_{neg}$
Output: Updated vectors $v(w_i)$ of words $w_i$, $i = 1, \ldots, N$
  for $t = 1, \ldots, N$ do
      /* Define target and context words for this iteration */
      $w = w_t$
      context($w$) $= \{c_1, \ldots, c_n \mid c_i = w_{t+i},\ i = -n, \ldots, n,\ i \neq 0\}$
      for $i = 1, \ldots, n$ do
          /* Calculate and propagate gradient for context words */
          $f = \sigma(v'(c_i)^\intercal v(w))$
          $g = \alpha(1 - f)$
          $\Delta = g \cdot v'(c_i)$
          $v'(c_i) = v'(c_i) + g \cdot v(w)$
          for $j = 1, \ldots, N_{neg}$ do
              /* Calculate and propagate gradient for negative words */
              $d_j \leftarrow$ word sampled from noise distribution, $d_j \neq c_i$
              $f = \sigma(v'(d_j)^\intercal v(w))$
              $g = -\alpha \cdot f$
              $\Delta = \Delta + g \cdot v'(d_j)$
              $v'(d_j) = v'(d_j) + g \cdot v(w)$
          end for
          /* Propagate gradient for target word */
          $v(w) = v(w) + \Delta$
      end for
  end for
```

training process becomes shorter, and the representations for rare words gain in quality (Mikolov et al. 2013b).

Since the goal of Skip-gram is to learn word representations, the simplifications it introduces are partly owed to disregarding constraints that have to be taken into account when the task is language modeling. These simplifications make Skip-gram an efficient approach to learning word representations.

3.4 Word sense embedding models

The elephant in the room for any given word-based embedding model is the existence of phenomena such as polysemy and homonymy. A word embedding model ignores the fact that some words may have more than one meaning, so in effect all meanings found in the training corpus for any given word are conflated into the same representation. When such representations are expected to portray accurately the meaning of words, as they often are, disregarding polysemy might become a shortcoming of the model.

While it is true that the learned representation for a polysemic word might contain information about all senses of the word present in the training data, it would not be a trivial task isolating a single sense from in an embedding, since there is no clear indication of which of the embedding's dimensions correspond to which sense. Furthermore, conflating multiple senses into one embedding affects the configuration of the semantic vector space as a metric space through the triangle inequality (Neelakantan et al. 2014): given a distance function d in the semantic space, and embeddings for any three words a, b, and c, the following relation holds:

$$d(a,c) \leq d(a,b) + d(b,c)$$

Consider a word b with two distinct senses, like *rock* ('music style' and 'mineral'), and unrelated words a and c each of which is closely related to one of the senses of b, like *blues* and *boulder*. As the training process configures the semantic space and pulls a and c towards a given their respective semantic relatedness, a and c are pulled towards each other despite not being related. An effect of conflating multiple senses in one embedding can be seen when inspecting the closest embeddings to a given word in the vector space according to a similarity measure like cosine distance. For example, consider the following list of the 10 nearest neighbors of the word *mouse* in a vector space trained with the Skip-gram model: *mickey, mice, keyboard, cat, rat, giant, disney, walt, pet, duck*. It contains words related to *mouse* as an animal, as a cartoon character, and as a computer component with no clear distinction among them, signaling an evident mix of unrelated meanings in one representation. A two-dimensional projection of the nearest neighbors of *mouse* from a 200-dimensional word vector space[3] can be seen in figure 3.2. Some attempts have been made to address this particular undesirable result of mixed senses in nearest neighbor lists. For example, Cuba Gyllensten and Sahlgren (2015) propose to structure these lists according to embeddings' locations relative

[3]The projection into two dimensions was obtained by applying principal components analysis (Jolliffe 1986) onto the original 200-dimensional space trained with Word2vec (Mikolov et al. 2013b) for a vocabulary of 71 291 words. The visualization was produced by TensorFlow's (Abadi et al. 2016) Embedding Projector (`https://projector.tensorflow.org`).

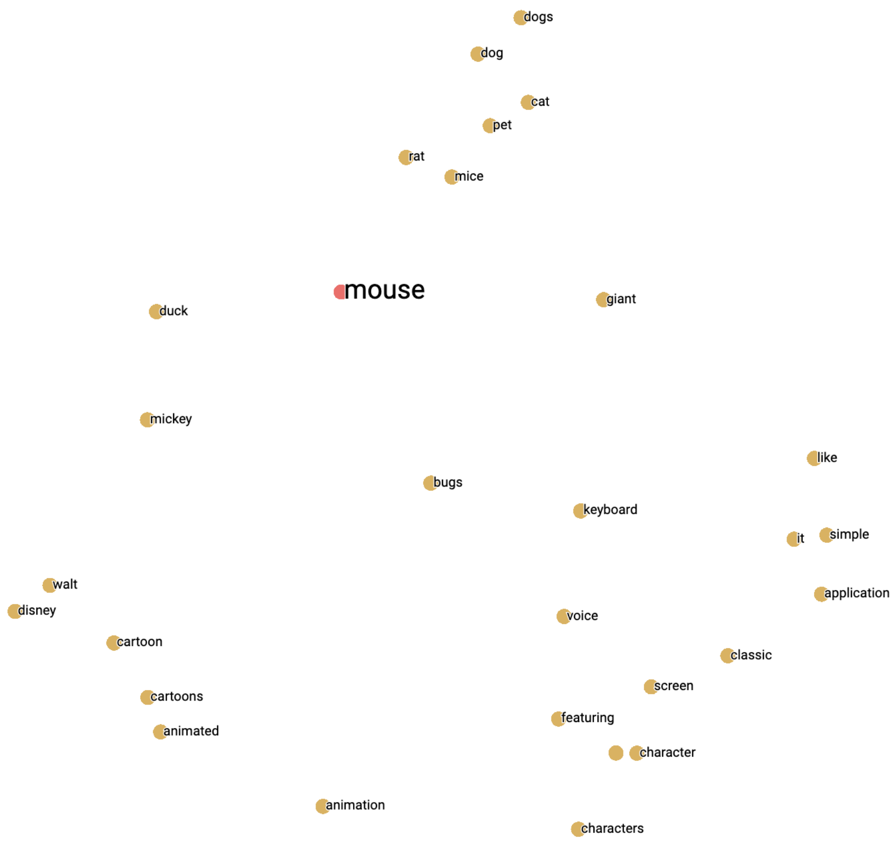


Figure 3.2: 2D projection of the nearest neighbors of *mouse* from a 200-dimensional word vector space.

to each other in the vector space. Such an approach makes it possible to disentangle neighborhoods in the presence of conflated senses and establish sub-regions that relate to different senses of a polysemous word. However, it increases the computational complexity of semantic similarity calculations using a distance function that is one of the main appeals of semantic spaces; in contrast, having separate representations for different word senses would eliminate the need for considerations of the vector space's structure in such situations. Besides its influence on the intrinsic properties of these spaces, and while meaning conflation might not affect the performance of NLP systems in some tasks, using separate embeddings for different word senses of a word does have an impact on tasks like part-of-speech tagging, semantic similarity, or semantic relation identification (Li and Jurafsky 2015). Furthermore, embeddings can also be applied to automating certain processes involved in the development of linguistic resources, such as suggesting new entries (see chapters 9 and 10). In cases where those resources differentiate between senses of a word, it is useful to have dedicated word sense embeddings so that a mapping between representations and entries in the resource can be established. This would allow to manipulate those entries in a semantic vector space and take advantage of its graded similarity properties.

In the wake of successful application of word embeddings as linguistic features in NLP, an amount of research has gone into adapting these models to make them able to learn separate representations for each meaning of any given word. Exploring the possibilities of learning separate representations for distinct meanings of a word and testing their potential applications are the main objective of this thesis.

Approaches to tackle this problem can be classified in a number of ways. In this text we focus on the distinction between approaches that are either supervised or unsupervised by a lexicon: we consider the supervision as a training signal coming from a linguistically informed resource different from the training corpus.

3.4.1 Lexicon-unsupervised models

Lexicon-unsupervised approaches, usually rely on methods that are able to classify instances of a word in context into one of its possible meanings. Models in this category differ in which method is used to perform classification and how the set of possible meanings is defined.

One of the earlier and most popular methods used for this purpose is clustering, applied either on word instances from a corpus or on word-level representations. For example, Schütze (1998) proposes to automatically find word sense representations in his earlier distributional Vector Space (Schütze 1993) as centroids of clusters of context vectors for any given word. Widdows and Dorow (2002) describe an incremental procedure for modeling semantic relatedness between nouns in a graph constructed using grammatical relations, such as lists of nouns, found in a corpus. This graph can be used to infer the different senses of a word used in the corpus by inspecting clusters of nouns related to a given word. Pantel and Lin (2002) use clustering of word feature vectors constructed using transformed word-context frequency counts. The clustering algorithm allows to identify subsets of features of a word that fit a particular cluster so that, effectively, a word can be assigned to more than one cluster. The different senses of a word are then identified by the separate clusters it is assigned to. Similarly, Véronis (2004) make use of clustering in a co-occurrence graph of words to detect distinct usages of a word in the text used to build the graph.

Reisinger and Mooney (2010) propose to cluster occurrences of a word w in a corpus into a preset number K of clusters based on the context of each occurrence; the centroid of each cluster is used as the *prototype* vector that represents the meaning of a particular usage of w defined by the context of the occurrences of w assigned to that cluster. Huang et al. (2012) also apply clustering based on context: first, a single-prototype-per-word is trained using a neural network; then, for each occurrence of a word w, a context vector c is calculated by averaging the vector representations of words in a context window around w; the context vectors c collected are clustered into K clusters and the resulting clusters are used to label each occurrence of w in the corpus; finally, the labeled corpus is used to train new representations.

Neelakantan et al. (2014) also use contextual clustering to modify the Skip-gram model. Given a context window around a word w from the training corpus, a context vector c is calculated by averaging the context words' word vectors. c is then clustered into one of K possible clusters for w; this cluster is in correspondence with one of K possible sense representations being learned by the model, which is then selected to be trained by this instance with a procedure similar to the original Skip-gram. The authors also propose a *non-parametric* version in which K is not fixed. An advantage of this model over the previous one is that sense representations are obtained in a single pass of the training algorithm.

In the model described in chapter 6 (Nieto Piña and Johansson 2015) we also propose to modify Skip-gram to train a number K_w of representations per word w. Instead of maintaining cluster centroids to which to compare an average context vector c, an approximation of the softmax function is used to calculate the probability of each sense of w given the instance's context; the most probable sense is selected to have its representation trained with that instance. The number of senses per word, K_w, can be individually specified for each word, which avoids keeping a fixed number of senses even for monosemous words. See figure 3.3 for a graphical representation of the training objective of such a model.

Li and Jurafsky (2015) and Bartunov et al. (2016) embed a prior from a Dirichlet process into the Skip-gram model, using the Chinese restaurant process (CRP) (Pitman 1995), in order to be able to account for a variable number of senses per word. For each training instance of

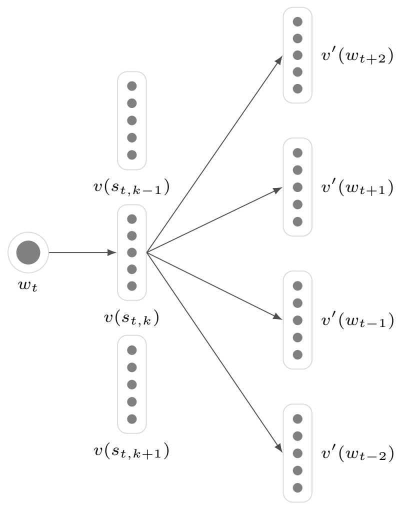


Figure 3.3: Training objective for a lexicon-unsupervised word sense embedding model: each word w_t can be mapped to several word sense vectors, $v(w_{t,k})$, each of which are optimized to predict their corresponding context word vectors, $v'(w_{t\pm i})$. Cf. the word embedding model shown in figure 3.1.

a word w, the stochastic process helps to assign it to any of the existing senses created with previous instances or, if the current instance is deemed to be different enough from those based on their similarity to the instance's context, creates a new sense. Once a sense is selected for an instance, Skip-gram's training algorithm updates the corresponding representation.

3.4.2 Lexicon-supervised models

The alternative to lexicon-unsupervised models is to take an existing inventory of senses per word, such as exists in a lexicon, and taking advantage of this information to inform the model. Lexica encoded in semantic networks, such as WordNet (Miller 1995), are particularly useful for this kind of approaches since they not only are useful for obtaining sense inventories, but also can establish semantic and lexical relations between concepts in their network structures.

Chen, Liu and Sun (2014) and Iacobacci, Pilehvar and Navigli (2015) propose to apply word sense disambiguation algorithms based on semantic networks (WordNet in the first case; BabelNet (Navigli and Ponzetto 2012) in the second) onto training corpora to automatically obtain sense-annotated data. This data can then be used to train a word embedding model that will learn representations for word senses. The quality of sense-annotated corpora depends heavily on the method used to disambiguate them: automatic disambiguation algorithms might produce incomplete or inaccurate results, while human annotation is usually costly to procure. For this reason, several models for training word sense representations have been developed that approach the task without making use of sense annotations. Johansson and Nieto Piña (2015) present a retrofitting algorithm that incorporates knowledge from a semantic network onto a pre-trained word vector space and splits single-word embeddings into several word sense embeddings. By approaching this task as an optimization problem, the pre-trained word embeddings are considered linear combinations of several word sense embeddings while minimizing the distances of these to related concepts as described by the semantic network. Similarly, Jauhar, Dyer and Hovy (2015) describe a method for retrofitting pre-trained word vectors to a semantic network. One of the models introduced by Jauhar, Dyer and Hovy (2015) is based on a graph learning technique applied to the task of finding representations for word senses.

3.5 Enriching embedding models with lexicographic data

In this and the previous chapter we have discussed the ways in which linguistic knowledge can be stored and operationalized for use in computational methods, as well as how computer models can automatically extract word meaning from text. In the present section we propose to intersect these two aspects of language technology in an attempt to create improved models of word meaning. We hypothesize that current automatic embedding models can benefit from formal lexical knowledge, with the added benefit that integrating existing linguistic resources in today's data-driven models could simplify the process of keeping them up-to-date by easing the need for manual work.

3.5.1 Embedding graphs

As a preliminary step before discussing the approaches to combining structured data from a lexicon with text from corpora as a mixed source of training data for embedding models, it is interesting to consider research efforts that have been made in the direction of creating distributed representations of information contained in graphs, given that such data structures are widely used to encode lexicographic information.

Graphs are well suited to representing relations between objects, such as those that exist between entries in a lexicon identifying semantic relations such as synonymy, hypoynymy, or meronymy. This is also the case in knowledge bases, where relations between entities give them structure. Being able to embed such structured information into a continuous vector space adds the potential to perform simple similarity measures based on the geometry of the vector space (as opposed to using graph-based similarity metrics.) It also adds the generalization power of embeddings; i.e., it is possible to embed new entities in an existing vector space in such a manner that it facilitates adding new knowledge to the corresponding graph-based resource. From our perspective, being able to embed graphs opens the possibility to use lexicographic information to our word sense embedding models.

There are different approaches to embedding graphs. In relation to language technology, embedding WordNet's (Miller 1998), a lexical database formed by sets of synonyms connected by relations between them, has been the focus of several experiments in this respect. For example, Bordes et al. (2013) propose to model the relationship r between two nodes in the graph, a and b, as three vectors in a vector space that are trained so that r functions as a translation of a into b in the vector space, or $a \sim b + r$. The learned embeddings are tested in a relation prediction between entries, which illustrates the potential application of embedded graphs for resource expansion. WordNet has also been used to learn word embeddings: Goikoetxea, Soroa and Agirre (2015) propose to generate a synthetic sentences that can be fed into a word embedding model like Skip-gram (Mikolov et al. 2013b) in replacement of a corpus. The synthetic data is generated by performing random walks over the graph, so that entries related by their lexical and semantic relations form sequences that are treated as sentences by the learning model. The resulting word embeddings are tested in word similarity and word relatedness tasks, showing that such an approach is able to capture word semantics from a lexical database.

3.5.2 Combining structured and unstructured data sources

The word and word sense representations learned by the methods introduced in previous chapters are, by the nature of said methods, obtained from unstructured data. That is, embeddings are able to express semantic information obtained from unannotated corpora by means of leveraging the distributional properties of words, and no other knowledge is taken into account when these representations are learned.

In chapter 2, however, linguistic resources were discussed as tools tailored to be used in NLP systems which contain lexicographic information in machine-readable form. The existence of these resources in the form of lexica usually predates the advent of modern automatic word embedding methods since lexica have been applied to NLP tasks for decades. As a result of this, a trove of linguistic knowledge is encoded in these resources and available to be used.

The specific work addressed in this thesis, namely automatically producing semantic representations for word senses, could potentially benefit from curated lexicographic knowledge as contained in these resources. As mentioned above, current embedding methods for words and word senses based on distributional information rely completely on the content of corpora, which can raise a number of issues related to the language contained in the corpus. For example, only those meanings of a word that are found sufficiently often in the corpus can be accurately represented: if we assume that the word *mouse* has two separate meanings, 'a small rodent' and 'a device to move the cursor on a computer screen', a system to represent these two meanings using only information from a corpus needs that both meanings are well represented in that corpus; if the texts included in the corpus are heavily biased towards technology articles, it is possible that the sense of *mouse* related to an animal may not appear at all in the corpus or, even if some of the texts contain references to *mouse* as an animal, they may not be enough to produce a good representation of it in comparison to *mouse* as a hardware item which is over-represented in the corpus.

It is in this light that we may view computer-readable lexica as a source of information that could be used to balance the shortcomings of representation models when they are exclusively based on corpora. In a situation as the one in the example above, we would like to supply the model with data from a lexicon that balances out the lack of information about *mouse* as an animal, since we expect the linguistic resource to provide a rather complete coverage of the language's lexicon.

A question that might be raised is, if we consider the coverage of word senses in lexica to be more balanced and complete than what may be found in corpora, why not make lexica the main source of data as opposed to corpora? There are several issues that might affect such a model. First is the difference in the nature of data that can be obtained from corpora and lexica: distributional models are conceived to learn representations from co-occurrence information, which is not present in lexica. While data can be generated from a lexicon in a format that mimics that of a corpus (see section 3.5.2.1 and chapter 7), statistics generated from it will not reflect any real co-occurrence patterns as found in corpora. For this reason, lexicographic data cannot substitute textual data.

An additional issue is that the representational power of modern embedding models relies on their ability to process large amounts of unstructured text to learn representations; it has been shown (Alsuhaibani et al. 2018) that the quality of representations tends to increase with the amount of text. The amount of data to train a distributional model that can be generated from a lexicon is limited by the number of possible combinations of lexicon entries. This number is bound to be smaller than the virtually limitless amount of variation that can be found in a corpus, and even smaller if those combinations of entries are restricted to be semantically coherent. This might affect the quality of the representations learned by the model.

Finally, while we expect a mature lexicon to have a wide coverage of a language's vocabulary, we cannot rule out the existence of gaps in this coverage. Since a lexicon is a manually crafted resource, it is entirely possible that some word senses are left out. The reasons for this could range from accidental omission to new senses appearing after the lexicon was created. Any such gaps would be inherited by any model trained exclusively on the lexicon. While it is not guaranteed that such gaps do not exist in a corpus, it is certainly simpler to increase the size of a corpus to ensure sufficient coverage than it is to do so in a lexicon.

In view of these issues it would seem than lexica and corpora might work better as complementary sources of data rather than one of them completely substituting the other. The question, then, would rather be whether lexica can be used to complement the distributional information contained in corpora in order to successfully train distributional models.

3.5.2.1 Generating training data from a lexicon

A common approach to introduce lexicographic data into representations learned by a model is to use a lexicon to generate a synthetic corpus that can subsequently be employed as training data for learning embeddings; for example, synthetic sentences can be generated from a lexicon encoded in a graph by performing random walks on the graph which output at random a series of nodes linked to words in the lexicon.

This is the approach taken by Goikoetxea, Soroa and Agirre (2015), who train Skip-gram and CBOW models (Mikolov et al. 2013b) on synthetic text generated by random walks on WordNet (Miller 1995), and report a performance on word similarity and relatedness tasks similar to that of a Skip-gram model trained on a corpus.

In a similar vein, the article presented in chapter 7 (Nieto Piña and Johansson 2016) explores the idea of building word sense embeddings for Swedish by training a modified Skip-gram model on a synthetic corpus generated by performing random walks on the Swedish lexicon SALDO (Borin, Forsberg and Lönngren 2013); in this case, the embeddings are intended to be used in a word sense disambiguation task.

3.5.2.2 Refining pre-trained embeddings with lexicographic knowledge

An alternative to explicitly generating data from lexica on which to train representations is to use their linguistic data to adjust, or retrofit, already trained embeddings on a corpus.

An example of this approach is the work by Faruqui et al. (2015), where the authors propose to iteratively adapt a set of word vectors to reflect relations in a graph-encoded lexicon such as WordNet: the retrofitting procedure tries to bring the vectors of words that share links in the lexicon graph closer together according to the Euclidean distance between them.

Johansson and Nieto Piña (2015) (not included in this thesis) also take this approach by posing the retrofitting task as an optimization problem whose goal is to obtain Swedish language word sense embeddings from word embeddings: by inspecting the lexicon to establish the different senses of any given word together with a set of semantically or lexically related word senses, or neighbors, a pre-trained word embedding is decomposed into a linear combination of word sense embeddings which are in turn optimized to minimize the distance from each of them to their respective neighbor senses in the lexicon's graph.

A different take on leveraging lexicographic information to adapt pre-trained word embeddings is to make use of dictionary definitions or glosses. For example, Hill et al. (2016) initialize a neural network language model with existing word embeddings and train it to predict the word embedding of the word whose dictionary definition is given as input, as an attempt to study sentence representations.

3.5.2.3 Jointly training embeddings from a lexicon and a corpus

A third way of integrating automatic embedding models with linguistic knowledge from resources is to modify the learning algorithm so that it can perform a joint training process which uses data from both a corpus and a lexicon.

Yu and Dredze (2014) propose a simple approach to this idea: extending Skip-gram's objective function (which maximizes the probability of context words given a target word) to semantic relations obtained from a lexicon. In this way, their model codifies these relations into the embeddings by optimizing them to maximize the conditional probability of related words given a target word, while at the same time keeping the original objective which learns from corpus data; the influence of each separate learning objective can be moduled through a single parameter.

Kiela, Hill and Clark (2015) explore both retrofitting and joint-learning approaches to injecting lexicographic knowledge into embedding models. Their proposal for joint learning is, given a target word in the Skip-gram training process, to sample a word from the set of all words in the lexicon semantically related to the target. Then that word is used as an extra context word and used for training normally as if it had been read from the corpus. The rest of the training process proceeds normally with the rest of context words from the corpus.

The model described in chapter 8 (Nieto Piña and Johansson 2017) uses a joint approach to learning Swedish word sense embeddings with data from a corpus and a lexicon. This model builds upon a modified version of Skip-gram (Nieto Piña and Johansson 2015), detailed in chapter 6, that is able to learn several embeddings per word solely from a corpus as a way of capturing different meanings of polysemous words. In the newer model, a regularizer function is applied onto the objective function. This regularizer encourages embeddings of word senses which share direct connections in a lexicon's graph to be closer together in the vector space.

Similarly to the work on training embeddings from dictionary definitions and glosses discussed above, Tissier, Gravier and Habrard (2017) propose to obtain a large amount of such data from different resources and aggregate it to a corpus. The resulting dataset is fed into a Skip-gram algorithm modified to make optimal use of the dictionary data.

4 Model evaluation

The work presented in this thesis introduces several models that automatically learn word sense representations. Evaluating the quality of such representations, or embeddings, is a challenging endevour for several reasons. In the first place, the mere notion of *word sense* is hard to define (see the discussion in section 2.1) and, by extension, so is assessing what constitutes a *good* representation for a word sense. Consider two models applied to representing the meanings of the noun *rock*. The first one assigns it two representations: one to represent its mineral sense and another to represent the type of music; the second model, however, learns three representations: one for rock as a material, another for rock as an object made of that material, and a third for the music style. Could we compare these two models based on those results and determine that one is better than the other? Kilgarriff (1997) would possibly argue that the answer depends entirely on the intended application of the representations. Not having an objective frame of reference by which to judge the quality of semantic representations makes it difficult to formulate and execute evaluation strategies.

There is also the fact that the objects of study, word sense embeddings, or vectors representing the the semantics of word senses in a vocabulary, are not easily interpretable: they are collections of real numbers arranged in a fixed set of dimensions, but these dimensions do not have an explicit interpretations due to the nature of the models developed to train them. These models act as a *black box*, where an input of corpus text complemented with lexicographic data is used to automatically codify the meaning of word senses into vectors that are observed as the model's output. The learning process, usually stochastic, thus *decides* automatically which semantic aspects are represented by each dimension and there are no explicit indications that might lead to a clear understanding of them. The lack of interpretability impedes assessing the quality of embeddings directly as would be possible with symbolic representations, where the mapping between objects and symbols provides an understanding of the representation. Evaluating embeddings, thus, has to be done in indirect ways that rather focus on testing the properties of the vector space (for instance, by looking at clusters of embeddings) or the performance of embeddings in downstream tasks (such as word sense disambiguation.)

In order to evaluate the models presented in this thesis, we have tried to give a complete overview of their strengths and weaknesses. We have often applied a twofold evaluation strategy to achieve this: illustrating characterizing properties of the trained semantic vector spaces, on one hand, and applying the obtained word sense representations on downstream tasks, on the other. The first point allows us to provide intuitive insight into the semantics learned by the model, while the second provides a demonstration of the applicability of the representations as well as a quantifiable measure to compare their performance against a suitable counterpart, such as a comparable model from the literature or a linguistic resource used as reference.

Another potential source of difficulty is the lack of benchmarks on which to evaluate a model. While this might not be a problem when working with English and a few other resource-rich languages in the NLP community, it can hinder the development of models in other, less favoured languages. The context in which this thesis was developed made it natural to work mostly with Swedish language data and, while our models are not language-specific and could be applied to data in different languages, they were trained with Swedish resources and hence should be evaluated on Swedish benchmarks.

In the rest of this chapter we look at different ways of evaluating semantic representations using qualitative and quantitative approaches. We also give an overview of the evaluation methodologies used in the articles contained in this thesis.

4.1 Qualitative evaluation

As a way of offering some insight into this aspect of embeddings, we perform an qualitative evaluation of word sense embeddings on most models, based on exploring a set of nearest neighbors for some selected word senses. Note that the most salient characteristic of a semantic vector space is that it is configured so that, given any three word senses, the embeddings of those two that are more semantically related will be separated by a distance shorter than that between any of them to the third, less related one. A corollary of this is that clusters of embeddings (defined, for example, as the set of all embeddings located within a sub-region of the semantic space bounded by a maximum distance to a given point in the space) will share some semantic similarity that is not necessarily present in embeddings located outside this cluster. We take advantage of this property to perform a manual assessment of the quality of word sense embeddings by collecting the nearest neighbors of a word sense as a way of examining its meaning as learned by the model by inspecting which are its most related, or closest in the space, word senses. Thus we can ask questions like 'Is the model able to learn the difference between the two senses, s_1 and s_2, of word w?' or 'Which senses s_i of word w is the model able to learn?' By looking at the nearest neighbors of each sense of w, we can in many cases understand their meanings as learned by the model.

For example, given the Swedish word *rock*, which has two senses, 'coat' and 'rock music', we can check that a model we have trained has managed to capture these two separate meanings by inspecting the nearest neighbors of each sense's embeddings:

Sense 1	**Sense 2**
syrtut 'frock coat'	*punk* 'punk music'
kappa 'coat'	*rappa* 'to rap'
kåpa 'cowl'	*rap* 'rap music'
päls 'fur coat'	*pop* 'pop music'
mudd 'cuff'	*jam* 'music jam'

From these results (obtained from the model described in chapter 8 with a balanced mix of training data from a corpus and a lexicon) we could conclude that the model did learn to separate those two meanings of *rock*, from which 'coat' would correspond to 'sense 1', and 'rock music' to 'sense 2', given that the nearest neighbors of each of these two respresentations cluster neatly around those two different topics. (I.e., clothing items versus musical terminology.) However, we could instead have obtained the following results when querying for nearest neighbors:

Sense 1	**Sense 2**
syrtut 'frock coat'	*hårdrock* 'hard rock music'
Rythm 'rhythm music'	*pop* 'pop music'
rockband 'rock band'	*jazza* 'to jazz'
Peepshows 'Peepshows'	*punk* 'punk music'
skaband 'ska band'	*dödsmetall* 'death metal music'

In this instance (obtained from the same model from chapter 8 but where the influence of the lexicon on the training data is negligible,) the retrieved nearest neighbors evidence that the model struggles to acquire the ‘coat’ meaning of *rock*: most of the nearest neighbors of ‘sense 1’, and all of them for ‘sense 2’, are music-related terms. (Possibly signaling an over-representation of the ‘rock music’ sense in the training data to the detriment of the ‘coat’ sense.) Such cases may be of help when diagnosing the model’s shortcomings since they may lead to understanding and explaining their causes.

Word sense embeddings may be induced from textual data only (using an unsupervised model) or, if a lexicographic resource is used, they may be linked to lexicographic senses (in a supervised model). In the first case, exploring sets of nearest neighbors is a way to clarify the meaning of each sense s_i of a word w as learned by the unsupervised model: by studying those other word senses closest in the vector space to each s_i, a notion of the general meaning associated with s_i can be formed. Since in this case there are no explicit links from s_i to lexicographic senses, inspection of nearest neighbors (either manually or automatically; see chapter 9 for an example of automatic linking) can be used to establish such links if they are needed to, for example, use word sense embeddings to annotate a text with a predefined set of lexicographic senses. In the case of a supervised system, it is possible that links between learned representations s_i and lexicographic senses are set by the training algorithm; in such a system, inspecting nearest neighbors can be useful to ascertain whether the model correctly learns the expected meaning of each sense s_i of w. For instance, in the second example above, if the model has been trained in such a way that we know that s_1 corresponds to ‘coat’ and s_2 to ‘rock music’, the resulting sets of nearest neighbors would make it clear that the model has not been able to learn the meaning of s_1 satisfactorily.

This evaluation method, however, is limited in the sense that it cannot cover the whole inventory of senses learned by a model, since it requires manual inspection of lists of word senses and it would not be feasible to apply it to the whole vocabulary. It is then usually offered as a qualitative assessment of a select number of cases deemed interesting for the model at hand as a way of acquiring insight into its strengths and weaknesses, and of illustrating the representation characteristics it exhibits. This approach to manually selecting *interesting* cases risks introducing confirmation bias into the results by, wittingly or unwittingly, choosing clusters of word senses that corroborate the author’s hypotheses. In order to avoid such situations it is important to choose sets of examples that give a fair illustration of a model’s capabilities and shortcomings. In our experiments, we have attempted to achieve this by counterposing positive examples, where the model works as expected, to negative ones, where the model underperforms. Only by analyzing both kinds of outcomes can this qualitative assessment approach be put to use to improve current models.

4.2 Quantitative evaluation

Quantitative evaluation methods with regards to semantic representations revolve around tasks which offer measurable results. For example, through a metric that rates the performance of the representations applied on a well-defined problem, such as the precision and recall exhibited by a system that employs word sense embeddings to select synonyms. While the methods devised to apply representations on such tasks might or might not need annotated data to be trained on depending on the nature of each specific task, in order to obtain performance measurements, it is usually the case that manually annotated data is needed to compare against the evaluation task's results; for example, in the case of synonymy detection, sets of synonyms created by an automatic method using word sense embeddings would be compared against manually created lists of synonyms (possibly from a thesaurus) to measure their similarity. This requirement and the availability of annotated data might play a rather mundane but ineludible role in selecting an evaluation strategy.

The type of quantitative evaluation also depends on the object of evaluation itself. When the goal is to measure the quality of the representation themselves, an intrinsic evaluation task is used; evaluating the performance of the representations in a downstream task is referred to as extrinsic evaluation.

4.2.1 Intrinsic evaluation

Intrinsic evaluation strategies aim at evaluating the semantic characteristics that the vector space has acquired in the learning process that produced the representations; their goal is to measure the quality of said representations, as opposed to determining their performance on a downstream application.

Of the different semantic properties of representations that can be evaluated, one of the most common is semantic similarity. Due to how multidimensional semantic spaces are usually configured, relating geometric distance with semantic similarity, this property is one of the most straightforward ways of characterizing such spaces. A *semantic similarity* task is typically formulated by defining pairs of semantically similar lexical units, such as (*money, cash*), and evaluating a model's capability to replicate that similarity; the most common lexical unit for which this task is formulated is the word form. (Note that applying word sense representations to the types of tasks described in this section would introduce an additional level of difficulty; selecting the right word for a similarity pair or an analogy tuple only needs that the represented vocabulary and the one used in the task coincide, while using word sense representations additionally requires coinciding word sense inventories, if one is defined for the task, or introducing a ad hoc mechanism that selects senses for the words of interest in the task.) To this end, a number of benchmarks with particular characteristics have been created as standardized tests: WordSim-353 (Finkelstein et al. 2002) and SimLex-999 (Hill, Reichart and Korhonen 2015) contain pairs of English words along with an average score provided by human annotators which grades the pairs from less to more similar with scores from 0 to 10; the Stanford Contextual Word Similarity dataset (SCWS) (Huang et al. 2012) similarly contains pairs of words, but includes sample sentences for each of them so that similarity judgments are not made in isolation but rather in context, which could potentially allow for a more robust testing of word sense embeddings since contexts can be used to disambiguate ambiguous instances. Several SemEval tasks have expanded this idea by providing benchmark datasets for semantic similarity tests directed towards a wider range of semantic representations; for example, SemEval-2017 Task 1 (Cer et al. 2017) focuses on whole sentence similarity, while Task 2 (Camacho-Collados et al. 2017) contemplates word pair similarity in five languages, both for pairs in the same language as in different languages.

Word analogy is another task that can be used to perform intrinsic evaluation of word embeddings. In this case, three words a, a', and b' are supplied, with a and a' sharing a specific relation, and the task is to find the word b that has the same type of relation with b', so that, for example, *queen* would be the solution to the triad

king – *man*
? – *woman*

or *Berlin* would answer

Amsterdam – *Netherlands*
? – *Germany*

When using vector representations to solve this task, the geometric properties of the vector space can be taken advantage of by finding the missing word b using vector space algebra; for example, b could be the word with its representation being the closest to the vector resulting from the operation $a - a' + b'$. The ability of the model to capture the relation between the two types of words then plays a role in the success on this task; benchmarks like the Google Analogy dataset (Mikolov et al. 2013a) tend to loosely categorize analogies as either semantic (such as the examples given above) or syntactic, where the relation between words has a grammatical nature, such as a morphological analogy

traveled – *travel*
? – *walk*

where a past tense form *walked* would be the answer. Gladkova, Drozd and Matsuoka (2016) provide a more fine-grained classification of relations in their BATS dataset in an effort to formalize this type of task.

Other tasks usually applied to perform intrinsic evaluation of representations are *synonym detection* (Jarmasz and Szpakowicz 2004), where a synonym for a target word has to be selected from a set of words; *outlier detection* (Camacho-Collados and Navigli 2016), where one or more outlier words have to be identified in a set where the rest of the words are semantically related; or *concept categorization* (Baroni et al. 2010), where a set of words has to be divided into a number of clusters of semantically related words.

4.2.2 Extrinsic evaluation

An alternative evaluation strategy is to turn to an extrinsic approach: using word sense embeddings as features in a downstream application such as word sense disambiguation (WSD). The quality assessment obtained from such a process focuses then on the properties of embeddings applied to a specific task, instead of directly testing their semantic representational ability as is done with intrinsic methods. On one hand, this approach allows to evaluate embeddings on practical applications on which they could be applied; on the other, downstream applications might not offer a general assessment of the quality of embeddings, as different applications could make use of different aspects of semantic representations, thus making it difficult to extrapolate performance results on one application to another. Combining intrinsic and extrinsic evaluation methods is recommended as a way of obtaining a more complete overview of the strengths and weaknesses of semantic representations learned by a particular model.

Given the ubiquity of semantic representations as features in NLP tasks, the choice of application on which to evaluate is extensive. The task chosen will have an impact on the aspects being evaluated; e.g., the role of embeddings in a WSD task is typically different from that in a machine translation. Therefore, evaluation results obtained in a particular downstream application cannot be taken to be representative of the performance of the same set of embeddings on a different application. (See for example the comparative study on the performance of word sense embeddings on different downstream applications by Li and Jurafsky 2015.)

Word sense disambiguation as a downstream application consists of an ambiguous target word situated in context (usually given as surrounding words in a sentence, although larger contexts can be used such as a complete document) which needs to be disambiguated; that is, assigned the specific word sense that fits the context. For example, the noun *canteen* can either mean ‘restaurant’ or ‘bottle’; disambiguating it in the sentence *The soldier shook the metal canteen to check whether it still contained any water* would entail selecting the second sense (bottle) based on the provided context.

Given its focus on word senses, WSD is a task well suited to evaluate word sense representations, since it can be used to test the effectivity of the representations at discriminating between the different meanings of a word. WSD counts with a long history dating back to the genesis of NLP, and thus there exist a myriad of techniques and approaches applied to this task (Navigli 2009). Regarding the data needed for training supervised WSD systems (which consistently outperform unsupervised ones; Raganato, Camacho-Collados and Navigli 2017), it can either consist of an annotated corpus or a knowledge base like a graph-based lexicon. For WSD to be used as an evaluation application, besides any training data in the format required for the chosen disambiguation system, there needs to be a benchmark of correctly disambiguated samples against which to compare the automatic disambiguation results; ideally, said samples would be disambiguated by humans to obtain a measure of the disambiguation system in terms of human performance. Some of these benchmarks include Senseval-2 (Edmonds and Cotton 2001) with annotations in ten different languages, and Semeval-2015 task 13 (Moro and Navigli 2015) with annotations in English, Spanish, and Italian; examples of sense-annotated corpora are SemCor (Miller et al. 1994) in English and Koala (Johansson et al. 2016) in Swedish.

Other examples of downstream applications that can be used as extrinsic evaluation test beds for semantic representations are word sense induction (WSI) (Neelakantan et al. 2014; Pelevina et al. 2016), sentiment analysis, part-of-speech tagging, or named entity recognition (Li and Jurafsky 2015; Fang et al. 2016).

4.3 Evaluation strategies used in this thesis

4.3.1 Article 1

In article 1 (chapter 6), a model is introduced to automatically learn representations for Swedish word senses where the only training data extraneous to a corpus is the number of senses a given word is expected to have. (I.e., given a word, the model is informed of how many senses it should learn for it, but is not given any indications as to the meaning of those senses.) Since this implies that there is no mapping between lexicographic senses and the learned word sense embeddings, the approach taken to evaluate the model's performance must be able to work without sense annotation; for instance, evaluating on an WSD task based on selecting one word sense embedding out of a set of possible ones could not be used here given that there is no mapping between the embeddings and the word senses used to annotate the test data. WSI is also a relevant task that could potentially be applied to evaluation without a mapping between word sense embeddings and a fixed sense inventory, but usual approaches to evaluating performance on this task tend to rely on a sense-annotated corpus, which was not available in Swedish at the time when this work was developed.

An initial qualitative inspection of nearest neighbors is performed as a way to highlight descriptive characteristics of the model: since the model is forced to learn a fixed number of senses per word, it is expected that the meaning of the senses learned will not always coincide with those described in a lexicon, as successfully obtaining a representation for any given sense depends on sufficient available data in the corpus covering instances of that sense; furthermore, there might be word senses used in the corpus that are not listed in the lexicon. Contrasting examples are given to illustrate these phenomena by means of listing a word's senses' nearest neighbors . E.g., the two recorded senses of *flyga* ('to travel by airplane' and 'to move through the air') seem to correspond to the senses learned by the model; meanwhile, the two lexicographic senses of *böna* ('bean' and a slang term for 'girl') are not found by the model: the sense corresponding to a slang term for 'girl' is not common in the corpus, so the model has instead made a distinction between 'bean as a plant' and 'bean as a food' in order to satisfy the requirement that *böna* must have two meanings.

At the time of writing this article, as mentioned above, no significantly large sense-annotated corpora or word similarity test sets (such as described in section 4.2.1) existed for the Swedish language; these resources have been commonly used to evaluate English word sense embeddings in the literature. (Such as word similarity tests used to perform intrinsic evaluation of the semantic space, Iacobacci, Pilehvar and Navigli 2015; or sense-annotated corpora as a source of test data to assess their performance on downstream tasks like, for example, WSD; Turian, Ratinov and Bengio 2010.) In order to provide a more comprehensive evaluation of the model that circumvented this lack of test benchmarks, an alternative quantitative assessment of the learned word sense embeddings was performed.

The proposed evaluation idea is to compare the word senses learned by the embedding model with the corresponding word senses as described in the lexicon. Again, lists of nearest neighbors are obtained as descriptions of the word senses in the vector space; as a lexicon counterpart, lists of the most similar word senses to any given one are generated using a graph similarity metric on the lexicon's underlying graph. The comparison of pairs of lists is systematized using three different clustering metrics intended to measure how similar these lists are. These measurements, then, are intended to offer an evaluation of how well the word sense representations are able to recreate the word senses as listed in the lexicon by comparing the lists of the closest terms to a given word's senses in each space. Note that there was no lexicon supervision in this model, which implies that the model was not optimized for this task.

The evaluation task is performed on a list of 300 lemmas (100 nouns, 100 verbs, and 100 adjectives), selected based on frequency and to be representative of the different parts-of-speech. Different list lengths are used, from 10 to 160 nearest neighbors. In order to provide a baseline against which to compare, a comparable word sense embedding model by Neelakantan et al. (2014) is trained on the same data and evaluated on the same task. This evaluation strategy, thus, manages to deal with scarcity of annotated data by proposing an alternative which leverages an existing lexicographic resource and balances its novelty with a variety of metrics and a baseline from the literature for comparison purposes.

An example of lists of concepts related to the noun *smak* 'taste' from the lexicon and the vector space can be seen in table 4.1. The Swedish

Lexicon s_1	**Lexicon s_2**
citrussmak ‘citrus flavor’	***smakriktning*** ‘trend in taste’
metallsmak ‘metal flavor’	*smaksak* ‘matter of taste’
fruktsmak ‘fruit flavor’	*smakfullhet* ‘elegance’
blodsmak ‘blood flavor’	***tycke*** ‘inclination’
bismak ‘off flavor’	*motvilja* ‘dislike’
arrakssmak ‘arrack flavor’	*sympati* ‘sympathy’
viltsmak ‘wild flavor’	*gottegris* ’sweet tooth’
jordgubbssmak ‘strawberry flavor’	*kortoxe* ‘card game maniac’
eftersmak ‘aftertaste’	*kaffemoster* ‘coffee aunty’
blecksmak ‘tin flavor’	*kitsch* ‘kitsch’

Vector space s_a	**Vector space s_b**
smaksinne ‘sense of taste’	***citrussmak*** ‘citrus flavor’
tycke ‘inclination’	***fruktsmak*** ‘fruit flavor’
finsmakare ‘gourmet’	*arom* ‘aroma’
ekovin ‘ecologic wine’	*beska* ‘bitterness’
preferens ‘preference’	*pomerans* ‘Seville orange’
doftsinne ‘olfactory sense’	*citronsorbet* ‘lemon sorbet’
stil ‘style’	*sorbet* ‘sorbet’
efterapning ‘imitation’	*frukt* ‘fruit’
smakriktning ‘trend in taste’	*passionsfrukt* ‘passion fruit’
kombination ‘combination’	*citrus* ‘citrus’

Table 4.1: Word lists for *smak* ‘taste’ from the lexicon (top) and vector space (bottom). Matching items in bold font.

word *smak* is defined in the lexicon as having two senses: flavor that can be perceived with the sense of taste (s_1) and personal preference (s_2). Note that this distinction seems to also be captured by the unsupervised model based on the lists of nearest neighbors of each sense in the bottom table. (s_a corresponding to s_2, and s_b to s_1.) However, the typically large size of the vocabulary makes it difficult that the word lists from the lexicon and the vector space have matching entries, and even more difficult that they appear in the same order. This results in low values for the clustering metrics used to compare the lists; the use of three different metrics in our experiments is intended to compensate for this by providing several assessments of the similarities between lists in order to provide a reliable measure of performance. In most cases, the magnitude of the measurements obtained are consistent across metrics.

4.3.2 Article 2

In article 2 (chapter 7), the focus shifted from a corpus to a lexicon's graph as a source of data from which to learn word sense representations. Random walks over the graph are used to emit sequences of terms related to a given seed word sense; these sequences are then used as synthetic contexts for the seed word sense, and inputted as training data into an embedding model.

In this case, an explicit mapping between word sense embeddings and lexicon entries is possible since the representations are trained on sequences of lexicon entries. Furthermore, efforts were underway to create Swedish word sense-annotated corpora. Unlike in the model introduced in the previous article, these conditions allowed us to prepare a WSD task on which to evaluate this word sense embedding model. The evaluation method in this case is solely extrinsic, and seeks to validate the hypothesis that, if the embeddings are able to codify the lexicographic information on which they are trained on, they should be effective at disambiguating word senses. The way the model is trained on a lexicon marks a difference with the model in the preceding article regarding the evaluation: since the lexicon has been used as training data, an evaluation by comparing the resulting representations to entries in the lexicon would not provide a completely fair assessment beyond a confirmation that the model is indeed learning the lexicographic data correctly.

The WSD task is performed on a test dataset composed of sentences containing one identified ambiguous word, one of whose possible senses has been selected to disambiguate it by annotators; this annotation uses the sense inventory of SALDO (Borin, Forsberg and Lönngren 2013). A disambiguation mechanism (Nieto Piña and Johansson 2015) is applied onto these sentences to select one the ambiguous word's senses, and its success is measured by comparing its output to the annotation; its performance is thus measured by accuracy, or the proportion of the total number of sentences correctly disambiguated. The mechanism used in this article makes use of semantic vector spaces' property that associates lexico-semantic relatedness with a vector similarity metric, the dot product. Using word sense and word embeddings for the target ambiguous word and the rest of (context) words in the sentence, respectively, the disambiguation mechanism measures the similarity between each possible word sense and its context, and selects the most similar sense to disambiguate the sentence.

The evaluation is complemented with three baselines: a random-sense baseline which chooses one of the possible senses for a word at random, a first-sense baseline which selects the first (usually the most frequent) sense according to the ordering in the lexicon, and the disambiguation results from a comparable, but more technically complex, graph-based WSD mechanism, UKB (Agirre and Soroa 2009), based on the Personalized PageRank algorithm (Brin and Page 1998). These baselines are compared to the WSD results of two variants of the model introduced in this article, which differ in the way the random walks are parameterized to produce synthetic contexts.

The aim of the model presented in this article was to adapt the successful embedding models to be able to learn from a structured lexicographic resource instead of from running text as found in corpora. The evaluation of such a model, then, should strive towards assessing whether this is a feasible endeavor and to what point such a system is able to learn to represent the semantics of word senses. Besides, as mentioned above, an evaluation approach such as a was used in the preceding article (where word sense representation quality was assessed comparing clusters of related terms in the lexicon and the vector space) would not suffice here, since the test data used there (sets of terms related in the lexicon) is the training data for the present model. Hence, using a WSD task as evaluation strategy seems a natural fit for this purpose. On one side, it is an independent downstream task; on the other, a successful assimilation of the information contained in a lexicon, which describes lexical and semantic relations between word senses, should be helpful towards discriminating which particular sense of a word is being used in an instance presented in context.

4.3.3 Article 3

The model presented in article 3 (chapter 8) learns from corpus data with supervision from a lexicon, in an attempt to guide the automatic learning of word sense representations from text with lexicographic definitions of senses. Such an approach is intended to address problems detected in the previous studies by reinforcing the learning process with formal knowledge of word senses while keeping the robustness and coverage of corpus-based representational models. The system introduced here, thus, tries to achieve a balance by drawing information from both a corpus and a lexicon and merging these two signals into a single training objective. Differences in the way this information is used and what proportion of it comes from each source give rise to several variants of the model.

The first assessment of this proposal is again a manual inspection of nearest neighbors of selected word senses. As before, this qualitative evaluation is useful for the purpose of illustrating particularities of the model being introduced. In this case, it is able to offer an intuitive understanding of the model's mix parameter $\rho \in (0, 1)$ which decides how much information from the lexicon is used to train the embeddings. For the examples given, an increased lexicographic influence shows more accurate results in terms of separation between senses of a polysemous word. Additionally, filtering nearest neighbors to only those not listed in the lexicon helps clarifying how well the two sources of information are being combined by the joint model, by means of checking whether those neighbors that only receive information from the corpus are the kind of concepts that one would expect associated with the word sense of which they are neighbors in the vector space.

After the qualitative assessment, a more systematic, quantitative evaluation is performed through two downstream applications: WSD and frame prediction. The WSD task proceeds as explained in the previous section, with three variants of the model being compared against the same three baselines described before. Maintaining this evaluation task, then, allows us to draw comparisons between this model and the previous one. As a result, we observe an improvement when using data combined from a lexicon and a corpus over using only data from a lexicon. This, combined with what we noted in the inspection of nearest neighbors where it is apparent that the lexicographic information has a positive influence in separating the senses of a word, allows us to conclude that combining these two sources of information provides better results than what would be possible with just one of them.

The second downstream application on which this model is evaluated is a frame prediction task. In a frame-semantics approach to word meaning (Fillmore and Baker 2009), words are classified into broad semantic classes known as *frames* in such a way that their meaning is defined by said frames. For example, the word *pizza* would belong to the frame FOOD. Furthermore, words with more than one meaning can be associated to more than one frame. For example, the first sense of the polysemous word *slag*, 'type', would belong to the frame TYPE; its second sense, 'hit', would belong to the frame IMPACT; its third sense, 'battle', would belong to the frame HOSTILE_ENCOUNTER, etc.

In our second evaluation task, we intend to find out how effective the word sense embeddings learned by our model are as predictors of frame membership. To do so, we collect a number of frames from the Swedish FrameNet (Friberg Heppin and Gronostaj Toporowska 2012) and train a linear support vector machine (SVM) classifier (Cortes and Vapnik 1995) for each frame to predict whether a word sense as represented by a vector belongs to that frame or not. A similar approach has been shown to be useful for the task of automating the suggestion of new lexical units to be included in FrameNet (Johansson 2014). The metric used to evaluate this system's performance is *average precision* applied to the ranking derived from each word sense's SVM score on a particular frame. A successful outcome of this type of evaluation is thus the case where those word senses that belong to the frame achieve the highest scores. As a baseline, the scores of our model's variants are compared to an embedding model's trained lemma representations. Expanding this comparison with different parameterizations of our model with several values of ρ allow us to evaluate the impact of the lexicographic information on this task.

In summary, the extensive evaluation strategy applied in this article allows for a detailed assessment of the model's behavior under different conditions and requirements. It is specially useful in the case of a highly configurable model like the one proposed here, where different learning configurations and parameterizations cause variations in performance on different tasks.

4.3.4 Article 4

In article 4 (chapter 9), we assess the capability of an independently trained word sense embedding model to link its sense representations to entries in a lexicon. A model able to represent words with a variable number of senses, Adaptive Skip-gram (Bartunov et al. 2016), is trained on a Swedish corpus and a mechanism is put into place to find the most similar entry in a lexicon to any given word sense embedding based on neighbor similarity. Investigating the possible links between automatically trained word sense representations and potential counterparts in a manually crafted inventory, the model opens the possibility to study mismatches between the two resources in order to identify possible new additions to the lexicon of new meanings of a word or, conversely, to clarify limitations in the lexicographic coverage of the corpus. Ultimately, resolving such a mapping between lexicon entries and an indeterminate number of automatically learned word sense representations provides us with representations for those entries in the lexicon linked with an embedding plus representations for word senses absent from the lexicon by those embeddings that remain unlinked. This results in an increased vocabulary coverage not limited by a fixed sense inventory, but grounds autmatically learned representations in the lexicon whenever possible.

The evaluation strategy in this case is oriented towards that aspect: we propose to assess whether we can take advantage of the proposed mapping between sense representations and lexicon entries to identify instances in text that correspond to senses not listed in the lexicon. A number of words are manually selected as having instances in the corpus whose meanings are not listed in the lexicon. The words selected for testing are divided into two groups: the first composed of those words with observed out-of-lexicon, *new* meanings, and the second comprising words for which the model has learned out-of-lexicon meanings that we consider spurious, such as brand names or foreign words with the same spelling.

A dataset of full sentences containing these words is then used to perform this test. By disambiguating these instances, they are assigned one of the possible sense representations learned by the model with a certain probability. This probability is used to assign each sentence a score, and thus the sentences can be ranked from most to least probably containing an out-of-lexicon sense; the ranking is matched against a manual annotation of these sentences, and the resulting performance is assessed using *area under the receiver operating characteristic curve* (AUC) as a measure of the probability that out-of-lexicon instances will be ranked higher than instances of senses listed in the lexicon. AUC is chosen as the metric rather than, for instance, precision and recall since AUC can be interpreted in terms of a ranking, which suits our test data results. (E.g., the AUC can be understood as the expected proportion of positive classification results before a random negative one is uniformly sampled.)

Table 4.2 shows an example of sentences containing the adjective *fet* which, besides its two senses contained in the lexicon, 'fat'/'fatty' and 'fertile', is a slang term for 'cool', not listed in our lexicon. The three top sentences are ranked highly, and the three bottom ones are ranked low. The scores given in the table are the probabilities of each sentence to contain an instance of *fet* with a sense not listed in the lexicon. Note that the language used in highly ranked sentences is informal, as would typically be the case when using the *fet* as 'cool'; in particular, the second sentence is ranked high even if the meaning of *fet* is one listed in the lexicon; however, the language used in the text is informal as the context for the slang usage of *fet* would usually be, which may have confused the classifier.

The evaluation method followed in this work is geared towards a specific downstream application which is closely related to the linking mech-

Sentence	**Probability**
@lundwall @sativaulrika Frågan var vilken som var en ***fet*** *låtrad och @Pstq o @PetterAlexis nämndes med låten “Pissar på dig.”* ‘@lundwall @sativaulrika The question was which was a **cool** song verse and @Pstq and @PetterAlexis were mentioned with the song “Pissar på dig.”	0.310
[...] och sen var det någon kille som började mucka med oss och fjanta sig haha och en ***fet*** *kille ba till oss: va tittar ni på?!* ‘[...] and then there was some guy who began to mess with us and act foolish haha and a **fat** guy was just like to us: what are you looking at?!’	0.310
[...] snubben är sjuk hur han får igång publiken med sina sköna låtar och med hans snabba ***feta*** *rhymes!* ‘[...] the guy is insane how he energizes the public with his nice songs and with his fast, **cool** rhymes.’	0.310
En fatig chardonnay passar lika bra till ***fet*** *fisk som lax, som till ljust kött, framför allt kyckling och fläskfilé.* ‘A barrel chardonnay combines equally well with **fatty** fish like salmon, as with light meat, especially chicken and pork tenderloin.’	0.123
Ingredienser: 1 burk tonfisk 2 msk ***fet*** *creme fraiche 1 msk* ***fet*** *majonnäs (jag lägger till lite finhackad rödlök och en sväng med pepparkvarnen också.)* ‘Ingredients: 1 can of tuna 2 spoonfuls of **fatty** crème fraîche 1 spoonful of **fatty** mayonnaise (I add a little of finely chopped red onion and a sweep with the pepper mill too.)’	0.123

Table 4.2: Ranked sentences containing the adjective *fet* and their probability of being an instance of a sense not listed in the lexicon. The first three ones belong to the top of the ranking and the last two to the bottom.

anism being evaluated, namely taking advantage of lexicon entries not linked by the proposed mechanism in order to identify instances in text of out-of-lexicon word senses. Success in this task, measured by comparing these findings to annotations, is meant to exemplify a possible application of automatically learned word sense embeddings towards improving existing linguistic resources by suggesting potential new entries in a lexicon. In particular, sentences scored highly in this task indicate a high probability of them being an instance of a word sense not listed in the lexicon; these sentences could be reported to lexicographers for consideration under a lexicon expansion process, thus alleviating their workload by reducing the amount of instances that need to be revised.

4.3.5 Article 5

In article 5 (chapter 10), semantic representations are applied to creating links between entries in a modern lexicon and an outdated thesaurus. This serves the purpose of integrating the older resource into a contemporary ecosystem of language technology tools, and ultimately paving the way to modernizing it with addition of new entries from the more modern resource. Two kinds of word sense representations, one symbolic based on the lexicon's tree structure (Johansson 2014) and one distributed learned by a post-processed word embedding model (Johansson and Nieto Piña 2015), are used to link word senses from the lexicon to words in the thesaurus in those cases where the word is ambiguous by being an entry in more than one of the thesaurus' classes. (For example, the polysemic Swedish word *fil* is present in at least two classes, FRICTION and CONTINUITY, corresponding to the word's senses 'file-tool' and 'row.') Once entries in both resources have been assigned representations, these are used to resolve these ambiguities in two separate ways: as features in a logistic regression classifier that assigns word senses from the lexicon to classes in the thesaurus, and by measuring similarity of word senses to classes using distances in the vector space.

The two types of representations and the two techniques to disambiguate entries are tested in this linking task by comparing the performance of each approach to a manually disambiguated gold standard from a sample of the ambiguous entries to be linked. These experiments show that word sense representations can be applied to mapping existing resources to one another, thus expanding the possible applications of such resources. Additionally, the favorable results obtained open the way to automatize the modernization of an outdated resource with up-to-date knowledge from a modern resource.

5 Summary and conclusions

At the onset of this thesis work, our main research aim was to investigate whether contemporary word embedding models could be adapted to obtain more fine-grained representations of word meaning, as described in research question Q1 formulated in chapter 1, by means of transitioning from representing word forms to representing word senses that can provide accurate representations of a lexicographic inventory of word senses in a particular language. A more detailed plan to achieve this with the help of lexicographic resources was drawn in research question Q2, and an inquiry into potential benefits of word sense representations towards automatizing expansion and maintenance of said lexicographic resources was proposed through research question Q3.

In the rest of this chapter, we review the outcomes of this undertaking by examining our work through the lens of the aforementioned research questions and the contributions that resulted from this work. We also sketch possible future lines of research that could be based on these developments.

5.1 Conclusions

In addressing the research lines laid by question Q1, we showed that it is indeed possible to adapt current neural word embedding models to automatically represent the different meanings of a word in separate vectors in the article presented in chapter 6. Additionally, this was achieved with little computational overhead with respect to the original word-based model. In the evaluation of the model described in that article we learned that, given the number of senses expected from a word, the model is able to separate and represent meanings associated with it by inspecting instances in different contexts. While these meanings do not always correlate with lexicographic definitions of a word's senses, this shows that the model learns meanings associated with different usages of the word, which we posit as a device that might help list potentially new senses.

Furthermore, by implementing additional modifications to word embedding models in the articles contained in chapters 7 and 8, we showed that not only is it possible for these models to learn to separate several meanings of a word, but also to learn them solely from lexicographic data or a combination of that with textual data from a corpus.

Those efforts relate to research question Q2; we demonstrated that lexicographic data can be leveraged by these models to derive high quality word sense representations in a vector space. In chapter 7 we showed a possible way of operationalizing this kind of manually crafted knowledge to derive representations as real-valued vectors which are useful in downstream applications such as WSD. In chapter 8 we took advantage of that fact and used lexicographic data as a source of training data in order to address shortcomings of a corpus-based model such as the one presented in chapter 6, resulting in word sense representations that better resemble their lexicographic definitions. In this third model, enabling control over the influence of the two separate sources of data in the training process allowed us to observe in detail the effects of lexicographic data in such a model, showing that it is possible to achieve a balance between lexicon and corpus as sources of data in order to improve performance in downstream applications.

In the different evaluation strategies applied to these models, we demonstrated that word sense embeddings are conducive to improved semantic representations. Accurate representations of individual word senses originate in semantically coherent vector spaces, as evidenced by lists of nearest neighbors to distinct senses of polysemic words. Furthermore, we observed an improvement in performance of word sense embeddings over word embeddings at predicting semantic frame membership. We also showed that the applications of these semantic representations is not limited to established NLP problems, but are also apt to help improve existing lexicographic resources by providing manipulable representations for lexicon entries that can be used to automate lexicographic work. These results address our final research question, Q3.

Indeed, in the article contained in chapter 9 we showed that by means of establishing links between automatically learned word sense embeddings and word senses listed in a lexicon, it is possible to generate suggestions of potentially new entries for the lexicon extracted from a corpus along with linguistic evidence. We propose that such a system be used to partly automate the task of expanding a lexicon in a way that reduces the human work load. Another example of automating the expansion of resources is described in chapter 10: in this case, word sense representations are successfully applied to the task of linking ambiguous entries in an older thesaurus with word senses listed in a modern lexicon as a way of increasing the accessibility of the thesaurus for use in NLP applications.

In summary, throughout this thesis work we have demonstrated different approaches to automatically learning word sense representations from corpora and lexica in three separate models, and demonstrated their relevance both in several NLP downstream applications as well as in two instances of lexicographic resource improvement.

5.2 Future work

We envision several lines of research as a continuation of this thesis work. On one hand, we acknowledge the possibility of improving the quality of word sense embeddings as a way of delivering greater semantic representational power for NLP applications; improving models used to learn the representations and providing efficient evaluation methods are two key pieces in achieving increased quality in word sense representations. On the other hand, we propose that there are additional applications in the context of lexicographic resource expansion on which word sense representations could have an role.

Having shown that integrating lexicographic knowledge as part of the training data does have a positive impact in the meaning representations learned by word sense embedding models, we hypothesize that refining the mechanisms which are used to extract that knowledge from resources might still yield further benefits regarding the quality of representations. We explored several approaches to injecting data from a lexicon into the embedding model in chapter 8 and observed the influence of this step into the resulting embeddings' performance on downstream tasks. Continuing this line of inquiry in a more exhaustive study could help determine an optimal mechanism for merging lexicographic information and corpora as training data for models with similar architectures.

A common topic of discussion when addressing the automatic generation of word sense representations is the approach used to parameterize the number of senses per word. In our models, we have chosen to follow the lead given by a lexicon to determine this number, but this just relegates the question to lexicographers and we should note that there is little agreement on the ideal sense granularity between different lexica. (See the discussion about this point in chapter 2.) There have been attempts to automatize this parameterization based on contextual evidence from a corpus (Neelakantan et al. 2014; Kågebäck et al. 2015; Bartunov et al. 2016) but, again, the result depends heavily on how the underlying resource (in this case, the corpus) is constructed. In this case, the completeness of the sense inventory would depend on the coverage of contexts in the corpus; i.e., if a sense is underrepresented in the corpus, it would probably not acquire representation in the model. From our point of view, this information would ideally be rooted in both lexicographic input and distributional data. As such, we believe that a truly complete sense inventory should not ignore lexicographic definitions of word senses, but those should be complemented with senses emerging from language use as can be found in corpora; such information could effectively cover the gaps that might exist in the lexica. Research dealing with modeling language change across time in corpora (Mitra et al. 2014; Hamilton, Leskovec and Jurafsky 2016; Tahmasebi and Risse 2017) does provide interesting techniques that could be useful for this purpose. The empirical study of semantic change across time calls for a flexible definition of word meaning in order to allow for new word senses to emerge and be captured in the semantic space, and thus provides examples of how to approach the construction of variable word sense inventories. Combining sense discovery techniques from word sense induction (Pantel and Lin 2002; Brody and Lapata 2009; Amrami and Goldberg 2018) with lexicographic word sense definitions in order to define the sense inventory used by a representational model could help learning more reliable word sense representations with regards to achieving an adequate coverage based on the needs of the different scenarios in which they are applied.

Deep neural networks, as data-driven models that learn to generalize are tightly related to representation learning and their application in NLP reaches back to the early 2000s (Gers and Schmidhuber 2001; Bengio et al. 2003). Recent developments in deep learning methods have enabled innovative approaches to what kind of data can be used to train semantic representations, from strings of characters rather than whole words (Bojanowski et al. 2017; Athiwaratkun, Wilson and Anandkumar 2018) to complex features extracted from neural language models like internal representations of the textual context (Peters et al. 2018; Amrami and Goldberg 2018; Devlin et al. 2018). Such approaches avoid some constraints characteristic of models centered around word forms as the main unit of linguistic information, which has the potential to evade problems related to conflation of multiple meanings by allowing more flexibility in choosing contexts used to learn the different meanings of a word.

Reiterating our discussion about evaluating representations in chapter 4, it should be stated that aiming for *higher quality representations* without suitable tools to evaluate their quality would turn out to be a mindless endeavor. Particularly in the case of evaluating word sense representations, we have found that the lack of sense-annotated resources and standard non-English benchmarks hinders the evaluation process and, as a result, it is an impediment to the development of models. Based on that, we stress that creation and maintenance of resources is a key piece in the struggle for better semantic representations. It would also be important to achieve a standardized evaluation approach that settles a principled evaluation strategy in terms of tasks, downstream applications, metrics, and precisely defined semantic aspects to be tested. Counting with a broadly accepted evaluation approach would allow for a streamlined process for new semantic representations to be assessed relative to existing models. Some recent research efforts have been put forward towards achieving such an unified evaluation strategy for semantic representations (Camacho-Collados and Navigli 2016; Raganato, Camacho-Collados and Navigli 2017), but the community still needs to find a standard that is flexible enough to cater to the broad variety of approaches to learning semantic representations.

Finally, we illustrated the possibilities of word sense representations applied to resource expansion in chapters 8, 9, and 10. These serve as examples of how to alleviate a usual bottleneck in resource improvement: time-consuming and expensive human labor. We are confident that there are many more similar tasks that could benefit from semantic representations as a tool to expedite such tasks, as a way to tackle the issues raised in the previous paragraph.

Part II

Published articles

6 Learning word sense embeddings from corpora

This chapter is a postprint version of the following publication:

Luis Nieto Piña and Richard Johansson 2015. A simple and efficient method to generate word sense representations. *Proceedings of the International Conference Recent Advances in Natural Language Processing,* 465–472. Hissar, Bulgaria.

Abstract

Distributed representations of words have boosted the performance of many Natural Language Processing tasks. However, usually only one representation per word is obtained, not acknowledging the fact that some words have multiple meanings. This has a negative effect on the individual word representations and the language model as a whole. In this paper we present a simple model that enables recent techniques for building word vectors to represent distinct senses of polysemic words. In our assessment of this model we show that it is able to effectively discriminate between words' senses and to do so in a computationally efficient manner.

6.1 Introduction

Distributed representations of words have helped obtain better language models (Bengio et al. 2003) and improve the performance of many natural language processing applications such as named entity recognition, chunking, paraphrasing, or sentiment classification (Turian, Ratinov and Bengio 2010; Socher et al. 2011; Glorot, Bordes and Bengio 2011). Recently, the Skip-gram model (Mikolov et al. 2013a, b) was proposed, which is able to produce high-quality representations from large collections of text in an efficient manner.

Despite the achievements of distributed representations, polysemy or homonymy are usually disregarded even when word semantics may have a large influence on the models. This results in several distinct senses of one same word sharing a representation, and possibly influencing the representations of words related to those distinct senses under the premise that similar words should have similar representations. Some recent attempts to address this issue are mentioned in the next section.

We present a simple method for obtaining sense representations directly during the Skip-gram training phase. It differs from most previous approaches in that it does not need to create or maintain clusters to discriminate between senses, leading to a significant reduction in the model's complexity. It also uses a heuristic approach to determining the number of senses to be learned per word that allows the model to use knowledge from lexical resources but also to keep its ability to work withouth them. In the following sections we look at previous work, describe our model, and inspect its results in qualitative and quantitative evaluations.

6.2 Related work

One of the first steps towards obtaining word sense embeddings was that by Reisinger and Mooney (2010). The authors propose to cluster occurrences of any given word in a corpus into a fixed number K of clusters which represent different word usages (rather than word senses). Each word's is thus assigned multiple prototypes or embeddings.

Huang et al. (2012) introduced a neural language model that leverages sentence-level and document-level context to generate word embeddings. Using the approach by Reisinger and Mooney (2010) to generate multiple embeddings per word via clusters and training on a corpus whose words have been substituted by its associated cluster's centroid, the neural model is able to learn multiple embeddings per word.

Neelakantan et al. (2014) tried to expand the Skip-gram model (Mikolov et al. 2013a, b) to produce word sense embeddings using the clustering approach of Reisinger and Mooney (2010) and Huang et al. (2012). Notably, Skip-gram's architecture allows the model to, given a word and its context, select and train a word sense embedding jointly. The authors also introduced a *non-parametric* variation of their model which allows a variable number of clusters per word instead of a fixed K.

Also based on the Skip-gram model, Chen, Liu and Sun (2014) proposed to maintain and train context word and word sense embeddings conjunctly, by training the model to predict both the context words and the senses of those context words given a target word. To avoid using cluster centroids to represent senses, the number of sense embeddings per word and their initial values are obtained from a knowledge network.

Our system for obtaining word sense embeddings also builds upon the Skip-gram model (which is described in more detail in the next section). Unlike most of the models described above, we do not make use of clustering algorithms. We also allow each word to have its own number of senses, which can be obtained from a dictionary or using any other heuristic suitable for this purpose. These characteristics translate into *a*) little overhead calculations added on top of the initial word-based model; and *b*) an efficient use of memory, as the majority of words are monosemic.

6.3 Model description

6.3.1 From word forms to senses

The distributed representations for word forms that stem from a Skip-gram (Mikolov et al. 2013a, b) model are built on the premise that, given a certain target word, they should serve to predict its surrounding words in a text. I.e., the training of a Skip-gram model, given a target word w, is based on maximizing the log-probability of the context words of w, $c_1, \ldots, c_n$:

$$\sum_{i=1}^{n} \log p(c_i|w). \tag{1}$$

The training data usually consists of a large collection of sentences or documents, so that the role of target word w can be iterated over these sequences of words, while the context words c considered in each case are those that surround w within a window of a certain length. The objective then becomes maximizing the average sum of the log-probabilities from equation 1.

We propose to modify this model to include a sense s of the word w. Note that equation 1 equals

$$\log p(c_1, \ldots, c_n|w) \tag{2}$$

if we assume the context words c_i to be independent of each other given a target word w. The notation in equation 2 allows us to consider the Skip-gram as a Naïve Bayes model parameterized by word embeddings (Mnih and Kavukcuoglu 2013). In this scenario, including a sense would amount then to adding a latent variable s, and our model's behaviour given a target word w is to select a sense s, which is in its turn used to predict n context words $c_1, \ldots, c_n$. Formally:

$$\begin{aligned} p(s, c_1, \ldots, c_n | w) &= \\ p(s|w) \cdot p(c_1, \ldots, c_n | s) &= \\ p(s|w) \cdot p(c_1|s) \ldots p(c_n|s). \end{aligned} \tag{3}$$

Thus, our training objective is to maximize the sum of the log-probabilities of context words c given a sense s of the target word w plus the log-probability of the sense s given the target word:

$$\log p(s|w) + \sum_{i=1}^{n} \log p(c_i|s). \tag{4}$$

We must now consider two distinct vocabularies: V containing all possible word forms (context and target words), and S containing all possible senses for the words in V, with sizes $|V|$ and $|S|$, resp. Given a pre-set $D \in \mathbb{N}$, our ultimate goal is to obtain $|S|$ dense, real-valued vectors of dimension D that represent the senses in our vocabulary S according to the objective function defined in equation 4.

The neural architecture of the Skip-gram model works with two separate representations for the same vocabulary of words. This double representation is not motivated in the original papers, but it stems from `word2vec`'s code[4] that the model builds separate representations for context and target words, of which the former constitute the actual output of the system. (A note by Goldberg and Levy 2014 offers some insight into this subject.) We take advantage of this architecture and use one of these two representations to contain senses, rather than word forms: as our model only uses target words w as an intermediate step to select a sense s, we only do not need to keep a representation for them. In this way, our model builds a representation of the vocabulary V, for the context words, and another for the vocabulary S of senses, which contains the actual output. Note that the representation of context words is only used internally for the purposes of this work, and that context words are word forms; i.e., we only consider senses for the target words.

[4]`http://code.google.com/p/word2vec/`

6.3.2 Selecting a sense

In the description of our model above we have considered that for each target word w we are able to select a sense s. We now explain the mechanism used for this purpose. The probability of a context word c_i given a sense s, as they appear in the model's objective function defined in equation 4, $p(c_i|s)$, $\forall i \in [1,n]$, can be calculated using the *softmax* function:

$$p(c_i|s) = \frac{e^{v_{c_i}^\intercal \cdot v_s}}{\sum_{j=1}^{|V|} e^{v_{c_j}^\intercal \cdot v_s}} = \frac{e^{v_{c_i}^\intercal \cdot v_s}}{Z(s)},$$

where v_{c_i} (resp. v_s) denotes the vector representing context word c_i (resp. sense s), $v^\intercal$ denotes the transposed vector v, and in the last equality we have used $Z(s)$ to identify the normalizer over all context words. With respect to the probability of a sense s given a target word w, for simplicity we assume that all senses are equally probable; i.e., $p(s|w) = \frac{1}{K}$ for any of the K senses s of word w, senses(w).

Using Bayes formula on equation 3, we can now obtain the posterior probability of a sense s given the target word w and the context words $c_1, \ldots, c_n$:

$$\begin{aligned} p(s|c_1,\ldots,c_n,w) &= \\ \frac{p(s|w) \cdot p(c_1,\ldots,c_n|s)}{\sum_{s_k \in \text{senses}(w)} p(s_k|w) \cdot p(c_1,\ldots,c_n|s_k)} &= \\ \frac{e^{(v_{c_1} + \cdots + v_{c_n}) \cdot v_s} \cdot Z(s)^{-n}}{\sum_{s_k \in \text{senses}(w)} e^{(v_{c_1} + \cdots + v_{c_n}) \cdot v_{s_k}} \cdot Z(s_k)^{-n}} &. \end{aligned} \tag{5}$$

During training, thus, given a target word w and context words $c_1, \ldots c_n$, the most probable sense $s \in$ senses(w) is the one that maximizes equation 5. Unfortunately, in most cases it is computationally impractical to explicitly calculate $Z(s)$. From a number of possible approximations, we have empirically found that considering $Z(s)$ to be constant yields the best results; this is not an unreasonable approximation if we expect the context word vectors to be densely and evenly spread out in the vector space. Under this assumption, the most probable sense s of w is the one that maximizes

$$\frac{e^{(v_{c_1}+\cdots+v_{c_n}) \cdot v_s}}{\sum_{s_k \in \text{senses}(w)} e^{(v_{c_1}+\cdots+v_{c_n}) \cdot v_{s_k}}} \tag{6}$$

For each word occurrence, we propose to select and train only its most probable sense. This approach of *hard sense assignments* is also taken in Neelakantan et al. (2014)'s work and we follow it here, although it would

Algorithm 2: Selection of senses and training using Skip-gram with Negative Sampling. (Note that v_x denotes the vector representation of word/sense x.)

```
Input: Sequence of words $w_1, \ldots, w_N$, window size $n$, learning rate $\alpha$,
  number of negative words $N_{neg}$
Output: Updated vectors for each sense of words $w_i$, $i = 1, \ldots, N$
  for $t = 1, \ldots, N$ do
      $w = w_t$
      $K \leftarrow$ number of senses of $w$
      $\text{context}(w) = \{c_1, \ldots, c_n \mid c_i = w_{t+i},\ i = -n, \ldots, n,\ i \neq 0\}$
      for $k = 1, \ldots, K$ do
          $p_k = \frac{e^{(v_{c_1}+\cdots+v_{c_n})\cdot v_{s_k}}}{\sum_{j=1}^{K} e^{(v_{c_1}+\cdots+v_{c_n})\cdot v_{s_j}}}$
      end for
      $s = \arg\max_{k=1,\ldots,K}\ p_k$
      for $i = 1, \ldots, n$ do
          $f = \frac{1}{1+e^{v_{c_i}\cdot v_s}}$
          $g = \alpha(1 - f)$
          $\Delta = g \cdot v_{c_i}$
          $v_{c_i} = v_{c_i} + g \cdot v_s$
          for $j = 1, \ldots, N_{neg}$ do
              $d_j \leftarrow$ word sampled from noise distribution, $d_j \neq c_i$
              $f = \frac{1}{1+e^{v_{d_j}\cdot v_s}}$
              $g = -\alpha \cdot f$
              $\Delta = \Delta + g \cdot v_{d_j}$
              $v_{d_j} = v_{d_j} + g \cdot v_s$
          end for
          $v_s = v_s + \Delta$
      end for
  end for
```

be interesting to compare it with a *soft* updates of all senses of a given word weighted by the probabilities obtained with equation 5.

The training algorithm, thus, iterates over a sequence of words, selecting each one in turn as a target word w and its context words as those in a window of a maximum pre-set size. For each target word, a number K of senses s is considered, and the most probable one selected according to equation 6. (Note that, as the number of senses needs to be informed – using, for example, a lexicon – monosemic words need only have one representation.) The selected sense s substitutes the target word w in the original Skip-gram model, and any of the known techniques used to train it can be subsequently applied to obtain sense representations. The training process is drafted in algorithm 2 using Skip-gram with Negative Sampling.

Negative Sampling (Mikolov et al. 2013b), based on Noise Contrastive Estimation (Mnih and Teh 2012), is a computationally efficient approximation for the original Skip-gram objective function (equation 1). In our implementation it learns the sense representations by sampling N_{neg} words from a noise distribution and using logistic regression to distinguish them from a certain context word c of a target word w. This process is also illustrated in algorithm 2.

6.4 Experiments

We trained the model described in section 6.3 on Swedish text using a context window of 10 words and vectors of 200 dimensions. The model requires the number of senses to be specified for each word; as a heuristic, we used the number of senses listed in the SALDO lexicon (Borin, Forsberg and Lönngren 2013). Note, however, that such a resource is not vital and could be substituted by any other heuristic. E.g., a fixed number of senses per word, as Neelakantan et al. (2014) do in their parametric approach.

As a training corpus, we created a corpus of 1 billion words downloaded from Språkbanken, the Swedish language bank.[5] The corpora are distributed in a format where the text has been tokenized, part-of-speech-tagged and lemmatized. Compounds have been segmented automatically and when a lemma was not listed in SALDO, we used the parts of the compounds instead. The input to the software computing the embeddings consisted of lemma forms with concatenated part-of-speech tags, e.g. *dricka*-verb for the verb ‘to drink’ and *dricka*-noun for the noun ‘drink’.

The training time of our model on this corpus was 22 hours. For the sake of time performance comparison, we run an off-the-shelf `word2vec` execution on our corpus using the same parameterization described above; the training of word vectors took 20 hours, which illustrates the little complexity that our model adds to the original Skip-gram.

[5]`http://spraakbanken.gu.se`

6.4.1 Inspection of nearest neighbors

We evaluate the output of the algorithm qualitatively by inspecting the nearest neighbors of the senses of a number of example words, and comparing them to the senses listed in SALDO.

Table 6.1 shows the nearest neighbor lists of the senses of two words where the algorithm has been able to learn the distinctions used in the lexicon. The verb *flyga* 'to fly' has two senses listed in SALDO: to travel by airplane and to move through the air. The adjective *öm* 'tender' also has two senses, similar to the corresponding English word: one emotional and one physical. The lists are semantically coherent, although we note that they are topical rather than substitutional; this is expected since the algorithm was applied to lemmatized and compound-segmented text and we use a fairly wide context window.

Sense 1	**Sense 2**
flyg 'flight'	*flaxa* 'to flap wings'
flygning 'flight'	*studsa* 'to bounce'
flygplan 'airplane'	*sväva* 'to hover'
charterplan 'charter plane'	*skjuta* 'to shoot'
SAS-plan 'SAS plane'	*susa* 'to whiz'

(a) flyga 'to fly'

Sense 1	**Sense 2**
kärleksfull 'loving'	*svullen* 'swollen'
ömsint 'tender'	*ömma* 'to be sore'
smek 'caress'	*värka* 'to ache'
kärleksord 'word of love'	*mörbulta* 'to bruise'
ömtålig 'delicate'	*ont* 'pain'

(b) öm 'tender'

Table 6.1: Examples of nearest neighbors of the two senses of two example words.

In a related example, figure 6.1 shows the projections onto a 2D space[6] of the representations for the two senses of *åsna*: 'donkey' or 'slow-witted person', and those of their corresponding nearest neighbors.

For some other words we have inspected, we fail to find one or more of the senses. This is typically when one sense is very dominant, drowning

[6]The projection was computed using `scikit-learn` (Pedregosa et al. 2011) using multidimensional scaling of the distances in a 200-dimensional vector space.

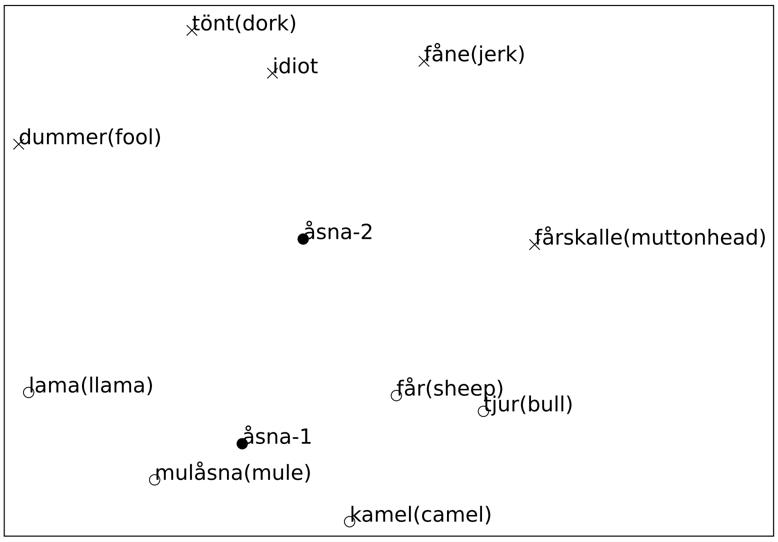


Figure 6.1: 2D projections of the two senses of *åsna* ('donkey' and 'slow-witted person') and their nearest neighbors.

out the rare senses. For instance, the word *rock* has two senses, 'rock music' and 'coat', where the first one is much more frequent. While one of the induced senses is close to some pieces of clothing, most of its nearest neighbors are styles of music.

In other cases, the algorithm has come up with meaningful sense distinctions, but not exactly as in the lexicon. For instance, the lexicon lists two senses for the noun *böna*: 'bean' and 'girl'; the algorithm has instead created two bean senses: bean as a plant part or bean as food. In some other cases, the algorithm finds genre-related distinctions instead of sense distinctions. For instance, for the verb *älska*, with two senses 'to love' or 'to make love', the algorithm has found two stylistically different uses of the first sense: one standard, and one related to informal words frequently used in social media. Similarly, for the noun *svamp* 'sponge' or 'mushroom'/'fungus', the algorithm does not find the sponge sense but distinguishes taxonomic, cooking-related, and nature-related uses of the mushroom/fungus sense. It's also worth mentioning that when some frequent foreign word is homographic with a Swedish word, it tends to be assigned to a sense. For instance, for the adjective *sur* 'sour', the lexicon lists one taste and one chemical sense; the algorithm conflates those two senses but creates a sense for the French preposition.

6.4.2 Quantitative evaluation

Most systems that automatically discover word senses have been evaluated either by clustering the instances in an annotated corpus (Manandhar et al. 2010; Jurgens and Klapaftis 2013), or by measuring the effect of the senses representations in a downstream task such as contextual word similarity (Huang et al. 2012; Neelakantan et al. 2014). However, Swedish lacks sense-annotated corpora as well as word similarity test sets, so our evaluation is instead based on comparing the discovered word senses to those listed in the SALDO lexicon. We selected the 100 most frequent two-sense nouns, verbs, and adjectives and used them as the test set.

To evaluate the senses discovered for a lemma, we generated two sets of word lists: one derived from the lexicon, and one from the vector space. For each sense s_i listed in the lexicon, we created a list L_i by selecting the N senses (for other words) most similar to s_i according to the graph-based similarity metric by Wu and Palmer (1994a). Conversely, for each sense vector v_j in our vector-based model, a list V_j was built by selecting the N vectors most similar to v_j, using the cosine similarity. We finally mapped the senses back to their corresponding lemmas, so that the two sets $L = \{L_i\}$ and $V = \{V_j\}$ of word lists could be compared.

These lists were then evaluated using standard clustering evaluation metrics. We used three different metrics:

- *Purity/Inverse-purity F-measure* (Zhao and Karypis 2001), where each of the lexicon-based lists L_i is matched to the vector-based list V_j that maximizes the F-measure, the harmonic mean of the cluster-based precision and recall:

  $$P(V_j, L_i) = \frac{|V_j \cap L_i|}{|C_j|} \quad R(V_j, L_i) = \frac{|V_j \cap L_i|}{|L_i|}$$

  The overall F-measure is defined as the weighted average of individual F-measures:

  $$F = \sum_i \frac{|L_i|}{\sum_k |L_k|} \max_j F(V_j, L_i)$$

- *B-cubed F-measure* (Bagga and Baldwin 1998), which computes individual precision and recall measures for every item occurring in one of the lists, and then averaging all precision and recall values. The F-measure is the harmonic mean of the averaged precision and recall.

- *V-measure* (Rosenberg and Hirschberg 2007), the harmonic mean of the *homogeneity* and the *completeness*, two entropy-based metrics. The homogeneity is defined as the relative reduction of entropy in V when adding the information about L:

  $$h(V,L) = 1 - \frac{H(V|L)}{H(V)}$$

  Conversely, the completeness is defined

  $$c(V,L) = 1 - \frac{H(L|V)}{H(L)}.$$

  Both measures are set to 1 if the denominator is zero.

Table 6.2 shows the results of the evaluation for nouns, verbs, and adjectives, and for different values of the list size N. As a strong baseline, we also include an evaluation of the sense representations discovered by the system of Neelakantan et al. (2014), run with the same settings as our system. This system is available only in its parametric version. (I.e., the number of senses per word is a fixed parameter.) As the words used in the experiments always have two senses assigned, this parameter is set to 2. This accounts for fairness in the comparison with our approach, which is given the *right* number of senses by the lexicon (and thus in this case also 2). We used the three metrics mentioned above: Purity/Inverse-purity F-measure (*Pu-F*), B-cubed F-measure (B^3-*F*), and V-measure (V). As we can see, our system achieves higher scores than the baseline in almost all the evaluations, despite using a simpler algorithm that uses less memory. Only for the V-measure the result is inconclusive for verbs and adjectives; for nouns, and for the other two evaluation metrics, our system is consistently better.

6.5 Conclusions and future work

In this paper, we present a model for automatically building sense vectors based on the Skip-gram method. In order to learn the sense vectors, we modify the Skip-gram model to take into account the number of senses of each target word. By including a mechanism to select the most probable sense given a target word and its context, only slight modifications to the original training algorithm are necessary for it to learn distinct representations of word senses from unstructured text.

To evaluate our model we train it on a 1-billion-word Swedish corpus and use the SALDO lexicon to inform the number of senses associated

	Pu-F		*B*^{3}*-F*		*V*	
N	**N-14**	**ours**	**N-14**	**ours**	**N-14**	**ours**
10	9.4	**10.7**	2.5	**2.8**	8.9	**10.6**
20	9.5	**10.8**	2.1	**2.4**	6.7	**8.9**
40	9.0	**9.9**	1.8	**2.0**	5.1	**7.2**
80	7.8	**8.9**	1.4	**1.7**	4.3	**5.6**
160	7.4	**8.2**	1.3	**1.5**	3.9	**4.7**

(a) Nouns.

	Pu-F		*B*^{3}*-F*		*V*	
N	**N-14**	**ours**	**N-14**	**ours**	**N-14**	**ours**
10	9.1	**10.8**	2.0	**2.5**	**11.3**	7.6
20	8.1	**9.3**	1.4	**1.7**	6.7	**7.5**
40	7.3	**8.2**	1.0	**1.3**	4.5	**4.5**
80	7.5	**8.7**	1.0	**1.3**	**3.7**	3.2
160	8.2	**10.3**	1.2	**1.7**	1.2	**1.5**

(b) Verbs.

	Pu-F		*B*^{3}*-F*		*V*	
N	**N-14**	**ours**	**N-14**	**ours**	**N-14**	**ours**
10	6.8	**7.6**	1.4	**1.7**	9.4	**10.7**
20	6.5	**7.6**	1.3	**1.5**	**8.5**	7.2
40	6.4	**7.3**	1.1	**1.3**	5.4	**5.8**
80	6.5	**7.0**	1.0	**1.1**	**5.2**	4.7
160	6.9	**7.5**	1.0	**1.1**	4.1	**4.4**

(c) Adjectives.

Table 6.2: Evaluation of the senses produced by our system and that of Neelakantan et al. (2014).

to each word. Over a series of examples in which we analyse the nearest neighbors of some of the represented senses, we show how the obtained sense representations are able to replicate the senses defined in SALDO, or to make novel sense distinctions in others. On instances in which a sense is dominant we observe that the obtained representations favour this sense in detriment of less common ones.

We also give a quantitative evaluation of the sense representations learned by our model using a variety of clustering evaluation metrics, and compare its performance with that of the model proposed by Neelakantan et al. (2014). In most instances of this evaluation our model obtains higher scores than this baseline, despite its relative lower complexity. Our model's low complexity is characterized by *a*) the simple word sense disambiguation algorithm introduced in section 6.3.2, which allows us to fit word sense embeddings into Skip-gram's existing architecture with little added computations; and *b*) the flexible number of senses per word, which takes advantage of the monosemic condition of most words to make an efficient use of memory. This low complexity is demonstrated by our training algorithm's small increase in running time with respect to that of the original, word-based Skip-gram model.

In this work, our use of a lexicon is limited to setting the number of senses of a given word, While this information proves useful for obtaining coherent sense representations, an interesting line of research lies in further exploiting existing knowledge resources for learning better sense vectors. E.g., leveraging the network topology of a lexicon such as SALDO, that links together senses of semantically related words, could arguably help improve the representations for those rare senses with which our model currently struggles, by learning their representations taking into account those of neighbour senses in the network.

Acknowledgments

We would like to thank the reviewers for their constructive comments. This research was funded by the Swedish Research Council under grant 2013–4944, *Distributional methods to represent the meaning of frames and constructions.*

7 Learning word sense embeddings from lexica

This chapter is a postprint version of the following publication:

Luis Nieto Piña and Richard Johansson 2016. Embedding senses for efficient graph-based word sense disambiguation. *Proceedings of TextGraphs-10: the Workshop on Graph-based Methods for Natural Language Processing, NAACL-HLT 2016,* 1–5. San Diego, USA.

Abstract

We propose a simple graph-based method for word sense disambiguation (WSD) where sense and context embeddings are constructed by applying the Skip-gram method to random walks over the sense graph. We used this method to build a WSD system for Swedish using the SALDO lexicon, and evaluated it on six different annotated test sets. In all cases, our system was several orders of magnitude faster than a state-of-the-art PageRank-based system, while outperforming a random baseline soundly.

7.1 Introduction

Word sense disambiguation (WSD) is a difficult task for automatic systems (Navigli 2009). The most accurate WSD systems build on supervised learning models trained on annotated corpora (Taghipour and Ng 2015), but because of the difficulty of the sense annotation task (Artstein and Poesio 2008), the luxury of supervised training is available for a few languages only.

An approach that circumvents the lack of annotated corpora is to take advantage of the information available in lexical knowledge bases (LKBs) like WordNet (Miller 1995, 1998). This kind of resource encodes word sense lexicons as graphs connecting lexically and semantically related concepts. Several methods are available that use LKBs for WSD (Navigli and Lapata 2007; Agirre and Soroa 2009). These approaches usually apply a relatively complex analysis of the underlying graph based on the context of a target word to disambiguate it; e.g., Agirre and Soroa (2009) use the Personalized PageRank algorithm to perform walks on the graph. However, these methods are computationally very costly, which makes them practically useless for large corpora.

In this paper, we investigate a more time-efficient approach to graph-based WSD. We represent the concepts in the LKB by training vector space models on synthetic datasets created using random walks on the LKB's graph. These synthetic datasets are built on the assumption that a random walk starting at a given node in the graph will be composed of inter-related concepts, effectively building a context for it. Training a vector space model on a collection of such data generated for each node in an LKB's graph would result in related concepts being represented near each other in the vector space, according to the distributional hypothesis (Harris 1954). We then use these representations to perform context-based disambiguation taking advantage of the geometric notions of similarity typical of vector space models. Using simple mechanisms for disambiguation and random walks allows our method to be orders of magnitude faster while keeping its accuracy well above the random-sense baseline.

7.2 Model

7.2.1 Word sense vector space model

The Skip-gram model (Mikolov et al. 2013b) is a neural network language model (Bengio et al. 2003) intended to produce high-quality word vector representations trained on large collections of text. In its original formulation these representations are limited to a vocabulary of word-forms extracted from the corpus used to train the model. The representations are dense vectors in a high-dimensional space in which it is expected that words with a similar meaning are represented near each other, which allows to associate a similarity measure with a geometrical distance measure. These representations are trained to, given a word, predict its context; the training algorithm, thus, works with two separate vector spaces in which context and target words are represented.

Skip-gram introduced a highly efficient approach to language modeling using a shallow neural architecture, which has also been extended to handle word *sense* representation (Neelakantan et al. 2014; Chen, Liu and Sun 2014; Johansson and Nieto Piña 2015; Nieto Piña and Johansson 2015). Our aim in this paper is to build *graph-based* word sense embeddings and apply them to the task of WSD as follows: Given a sentence with an ambiguous word, we can then compare the representation of its context words with each of the ambiguous word's sense representations to decide which of them fits the context better.

For this purpose we use a modified version of the original Skip-gram implementation by Levy and Goldberg (2014b), `word2vecf`, which specifies separate target and context vocabularies, making it possible to represent word senses as targets while keeping the context vocabulary restricted to word forms.

7.2.2 Random walks as contexts

Given a node in a graph G, a random walk generates a random sequence of interconnected nodes by selecting randomly from the edges of the current node at each step. The length of the random walk is controlled by a stop probability p_s. I.e., at each node visited in the walk, the probability of stopping is p_s; if the walk does not stop, one of the node's edges is followed to include another node in the walk. We repeat this process a number of times N_{walk} for each node in G to obtain $|G| \times N_{\text{walk}}$ random walks, where $|G|$ is the number of nodes in G.

The nodes in G are expected to represent word senses, while its edges connect semantically related word senses. Thus, a sequence of nodes generated by a random walk is a set of related word senses. Our assumption is that such a sequence can be considered a context of its starting node: a set of words that are related to, and can appear together in real texts with, the word sense represented by that node, thus emulating real text sentences; to what extent this assumption holds depends of course on the structure of the LKB we are using. Previous efforts in building word embeddings have shown the plausibility of this approach (Goikoetxea, Soroa and Agirre 2015).

It can also be argued that different senses of a word appear in different contexts (e.g., it is plausible that the *music* sense of *rock* appears together with *play* and *concert*, while not so much with *mineral* or *throw*). By generating contexts semantically related to a given sense of a word, we expect the resulting vectors trained on them to be effective in the task of word sense disambiguation. At the same time, as the same number of contexts (random walks) are generated for each word sense (node in G), no word sense in the vocabulary contained in G is under-represented, as can be the case in real text corpora.

In order to conform to the definition of context vocabulary given above, given that nodes in G represent senses, those senses that form part of a context in a random walk will have to be mapped to their word-forms using a dictionary.

7.2.3 WSD mechanism

Given an ambiguous target word w_i in context $c_{i,j}$, $j = 1, \ldots, n$, our disambiguation mechanism assigns a score to each of its senses $s_{i,k}$, $k = 1, \ldots, K$, based on the dot product of the sense vector $v(s_{i,k})$ with the sum of the context vectors $v(c_{i,j})$:

$$v(s_{i,k})^{\intercal} \cdot \sum_{j=1}^{n} v(c_{i,j}) \tag{7}$$

Note that all the information used to disambiguate originates from the LKB in the form of co-occurrence of concepts in RWs on the graph; no *external* information, like *a priori* sense probabilities, are used. The scores in equation 7 are derived from the probability of the context words given a sense, calculated using the softmax function:

$$p(c_{i,1}, \ldots, c_{i,n} | s_{i,k}) = \frac{e^{v(s_{i,k})^{\intercal} \cdot \sum_{j=1}^{n} v(c_{i,j})}}{\sum_{k'=1}^{K} e^{v(s_{i,k'})^{\intercal} \cdot \sum_{j=1}^{n} v(c_{i,j})}}.$$

This expression is based on Skip-gram's objective function used to maximize the probability of a context given a target word. In our method, then, each ambiguous word is disambiguated by maximizing its sense scores (Eq. 7) and selecting the highest scoring sense for that instance.

7.3 Experiments

We built a WSD system for Swedish by applying the random walk-based training described above to the SALDO lexicon (Borin, Forsberg and Lönngren 2013). In the experiments, we then evaluated this system on six different annotated corpora, in which the ambiguous words have been manually disambiguated according to SALDO, and compared it to random and first-sense baselines and UKB (Agirre and Soroa 2009), a state-of-the-art graph-based WSD system.

7.3.1 The SALDO lexicon

SALDO is the largest freely available lexical resource of this kind available for Swedish: the version used in this paper contains roughly 125,000 entries organized into a single semantic network. Similarly to WordNet (Miller 1998), SALDO is a large, manually constructed, and general-purpose lexicon that defines the senses in terms of a semantic network. But there are also important differences between WordNet and SALDO, first of all that the sense distinctions in SALDO tend to be more coarse-grained than in WordNet.

The SALDO network is defined in terms of semantic *descriptors*. A descriptor of a sense is another sense used to define its meaning. The most important descriptor is called the *primary* descriptor (PD), and since every sense in SALDO (except an abstract root sense) has a unique PD, the PD subgraph of SALDO forms a tree. A sense can be related to its primary descriptor through hyponymy, synonymy, meronym, antonymy, or some other relationship such as a predicate–argument relationship; this is another contrast with WordNet, where it is the hyponymy subgraph that forms the backbone. In practice, most PDs in SALDO are either synonyms or hypernyms.

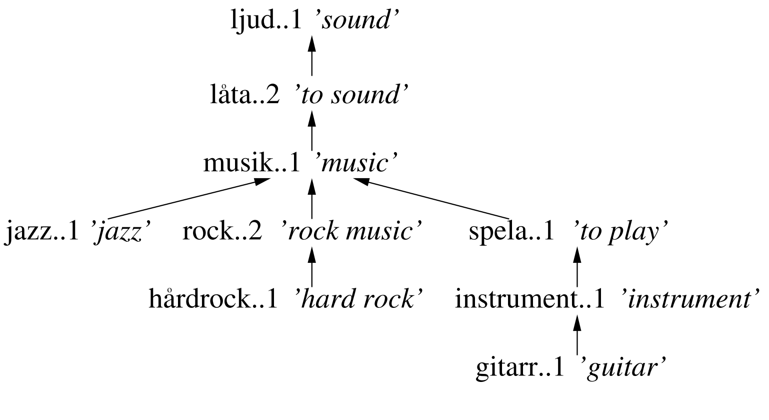


Figure 7.1: A part of the primary descriptor tree in SALDO.

To exemplify, figure 7.1 shows a fragment of the PD tree. In the example, there are some cases where the senses are connected through hyponymy, such as *hard rock* being a type of *rock music*, but there are also other types of relations, such as *to play* being defined in terms of *music*.

In this work, we use the PD tree to generate the random walks. For instance, a random walk starting at *rock music* might consist of the senses corresponding to *music*, *play*, *instrument*, *guitar*. As mentioned above, these senses are then mapped back to their corresponding lemmas before being used as context features in `word2vecf`.

7.3.2 Evaluation corpora

For development and evaluation, we used six different collections of sense-annotated examples. The first two, the *SALDO examples* (SALDO-ex) and *Swedish FrameNet examples* (SweFN-ex) consist of sentences selected by lexicographers to exemplify the senses (Johansson and Nieto Piña 2015). The former is dominated by the most frequent verbs, while the latter has a more even distribution. In our experiments, these two collections were used as a development set to tune the system's parameters.

The additional four collections are taken from an ongoing annotation project (Johansson et al. 2016); each collection corresponds to a domain: blogs, novels, Wikipedia, and Europarl (Koehn 2005). Unlike the two collections mentioned above, in which the instances have been selected by lexicographers to be prototypical and to have a good coverage of the sense variation, the instances in these four collections are sampled uniformly from running text.

Corpus	**Size**	$\bar{n}_s$
SALDO-ex	1055	3.1
SweFN-ex	1309	2.9
Blogs	1014	2.9
Europarl	1282	2.7
Novels	1204	3.0
Wikipedia	1311	2.7

Table 7.1: Evaluation corpus statistics.

We preprocessed the examples in the six collections to tokenize, compound-split, and lemmatize the texts, and to determine the set of possible senses in a given context. We used content words only: nouns, verbs, adjectives, and adverbs. All unambiguous instances were removed from the sets, and we also excluded sentences where the target was a multiword expression or a part of a compound word. We also removed a few instances that could not be lemmatized unambigously.[7] Table 7.1 shows the number of instances in each collection, as well as the average number of senses per instance ($\bar{n}_s$).

7.3.3 Evaluation

A model is trained on synthetic datasets compiled from random walks on SALDO. These walks are parameterized by their stop probability p_{stop}, which effectively controls the length of the random walk and has two effects: it impacts the size of training data (a lower p_{stop} will generate longer walks on average, and vice versa); and it controls the level of relatedness between the target sense and the words included in the context—a longer walk will wander away from the initial sense, including increasingly unrelated concepts, while a shorter one will keep its concepts closely related.

We tuned the model by training several versions with different p_{stop} and evaluated their performance on the development datasets. As the best-performing parameterization, we chose $p_{\text{stop}} = 0.25$, which generates random walks with an average length of 3.75 nodes and achieves an accuracy of 51.6% on the development datasets. In all cases, the vector space's dimensionality for senses and contexts is 200, and 10 iterations of the training algorithm are used.

Using this parameterization, we trained models on two different RW datasets: on one, random walks were performed on an unweighted version of SALDO (i.e., all edges are equally probable from any given node); on the other, the graph was weighted favoring the selection of a node's unique PD, with probability 0.5, over inverse (incoming) PD connections, which were uniformly distributed over the remaining probability mass.

The disambiguation mechanism explained in section 7.2.3 is applied to sentences containing one ambiguous word using the sense and context representations that result from training the models: A score is calculated for each of the senses of an ambiguous target word in a context window of size 10 (to each side of the target word) and the highest scoring sense is selected to disambiguate the entry. The accuracy of the method is then obtained by comparing these selections with the annotations of the test datasets.

[7]Note that this is done only to facilitate comparison to the UKB model; it is not necessary for our system.

The results of evaluating this models on each component of the test dataset are shown in table 7.2. The performance of the UKB model[8] by Agirre and Soroa (2009) on our datasets is also shown in this table, along with first-sense (S1) and random-sense baselines (Rand). These figures show that the first-sense approach is still a hard baseline. Amongst our two models (RW), the one trained on a weighted graph (w) performs consistently better; both of them outperform by a wide margin the random-sense baseline. The accuracy on the development sets is generally lower, especially in the case of the first-sense baseline, underlying their difference in nature with respect to the test sets (see section 7.3.2).

Corpus	**RW (uw)**	**RW (w)**	**UKB**	**S1**	**Rand**
SALDO-ex	52.1	51.6	55.5	53.2	39.3
SweFN-ex	51.0	49.5	53.7	54.3	40.3
Blogs	49.8	58.0	70.0	72.4	40.8
Europarl	55.7	59.4	67.6	67.9	42.3
Novels	56.6	59.9	70.1	77.2	40.1
Wikipedia	60.4	59.6	69.5	76.8	41.2

Table 7.2: WSD accuracies on the development and test sets.

Regarding execution times, the tested models take a few hours to train and, once trained, are able to disambiguate over 8 000 instances per second, significantly surpassing the UKB model's times, which disambiguates approximately 8 instances per second. This is related to the fact that the complexity of our disambiguation mechanism is linear on the context vectors (see equation 7), while the UKB model's is dependent on the graph size.

7.4 Conclusion

In this paper we have presented a WSD method trained on a synthetic corpus composed of random walks over an LKB's graph. This method has been shown to be very efficient, disambiguating thousands of words per second. While the accuracy obtained by the method does not beat that of comparable approaches, it is several orders of magnitude faster while outperforming a random-sense baseline. As has been shown in the results, the way in which random walks are generated seems to have an

[8]We used version 2.0 of UKB, run in the *word-by-word* mode, using an unweighted graph based on the PD tree.

influence on the results; exploring alternative ways of generating training datasets might be a way of improving the model's results while retaining its efficiency.

Acknowledgments

This research was funded by the Swedish Research Council under grant 2013–4944.

8 Learning word sense embeddings from corpora and lexica

This chapter is a postprint version of the following publication:

Luis Nieto Piña and Richard Johansson 2017. Training word sense embed- dings with lexicon-based regularization. *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers).* Asian Federation of Natural Language Processing. Taipei, Taiwan.

Abstract

We propose to improve word sense embeddings by enriching an automatic corpus-based method with lexicographic data. Information from a lexicon is introduced into the learning algorithm's objective function through a regularizer. The incorporation of lexicographic data yields embeddings that are able to reflect expert-defined word senses, while retaining the robustness, high quality, and coverage of automatic corpus-based methods. These properties are observed in a manual inspection of the semantic clusters that different degrees of regularizer strength create in the vector space. Moreover, we evaluate the sense embeddings in two downstream applications: word sense disambiguation and semantic frame prediction, where they outperform simpler approaches. Our results show that a corpus-based model balanced with lexicographic data learns better representations and improve their performance in downstream tasks.

8.1 Introduction

Word embeddings, as a tool for representing the meaning of words based on the context in which they appear, have had a considerable impact on many of the traditional Natural Language Processing tasks in recent years. (Turian, Ratinov and Bengio 2010; Collobert et al. 2011; Socher et al. 2011; Glorot, Bordes and Bengio 2011) This form of semantic representation has come to replace in many instances traditional count-based vectors (Baroni, Dinu and Kruszewski 2014), as they yield high-quality semantic representations in a computationally efficient manner, which allows them to leverage information from large corpora.

Due to this success, some attention has been devoted to the question of whether their representational power can be refined to further advance the state of the art in those tasks that can benefit from semantic representations. One instance in which this could be realized concerns polysemous words, which has led to several attempts at representing word senses instead of simple word forms. Doing so would help avoid the situation in which several meanings of a word have to be conflated into just one embedding, typical of simple word embeddings.

Among the different approaches to learning word sense embeddings, a distinction can be made between those that make use of a semantic network (SN) and those that do not. Approaches in the latter group usually apply an unsupervised strategy for clustering instances of words based on the context formed by surrounding words. The resulting clusters are then used to represent the different meanings of a word. These representations characterize word usage in the training corpus rather than lexicographic senses, and run the risk of marginalizing under-represented word senses. Nonetheless, for well represented word senses, this strategy proves to be effective and adaptable to changes.

The alternative is to integrate an SN in the learning process. This kind of resource encodes a lexicon of word senses, connecting lexically and semantically related concepts, usually in the form of a graph. Methods that take this approach are able to work with lexicographic word senses as defined by experts, usually integrating them in different ways with corpus-learned embeddings. However, their completeness depends on the quality of the underlying SN.

In this paper, we present an approach that tries to achieve a balance between these two variants. We propose to make use of an SN for learning word sense embeddings by leveraging its signal through a regularizer function that is applied on top of a traditional objective function used to learn embeddings from corpora. In this manner, our model is able to merge these two opposed sources of data with the expectation that each one will balance the limitations of the other: flexible, high-quality embeddings learned from a corpus, with well defined separation between the expert-defined senses of any given polysemic word. The influence of each source of information can be regulated through a mix parameter.

As the corpus-based part of our model, we use a version of the Skip-gram (Mikolov et al. 2013b) model that is modified so that it is able to learn two distinct vocabularies: word senses and word forms as introduced by Nieto Piña and Johansson (2015). Regarding the SN data, we focus our attention on its underlying graph. We assume that neighboring nodes in such a graph correspond to semantically related concepts. Thus, given a word sense, a sequence of related word senses can be generated from its neighbors. A regularizer function can then be used to update their corresponding embeddings so that they become closer in the vector space. This has the benefit of creating clear separations between the different senses of polysemic words, precisely as they are described in the SN, even in the cases where this separation would not be clear from the data in a corpus.

We give an overview of related work in section 8.2, and our model is described in detail in section 8.3. The resulting word sense embeddings are evaluated in section 8.4 on two separate automated tasks: word sense disambiguation (WSD) and lexical frame prediction (LFP). The experiments used for evaluation allow us to investigate the influence of the lexicographic data on the embeddings by comparing different model parameterizations. We conclude with a discussion of our results in section 8.5.

8.2 Related work

The recent success of word embeddings as effective semantic representations across the broad spectrum of NLP tasks has led to an increased interest in developing embedding methods further in order to acquire finer-grained representations able to handle polysemy and homonymy. This effort can be divided into two approaches: those that tackle the problem as an unsupervised task, aiming to discover different usages of words in corpora, and those that make use of knowledge resources as a way of injecting linguistic knowledge into the models.

Among the earliest efforts in the former group is the work of Reisinger and Mooney (2010) and Huang et al. (2012), who propose to cluster occurrences of words based on their contexts to account for different meanings. With the advent of the Skip-gram model (Mikolov et al. 2013b) as an efficient way of training prediction-based word embedding models, much of the research into obtaining word sense representations revolved around it. Neelakantan et al. (2014) and Nieto Piña and Johansson (2015) make use of context-based word sense disambiguation (WSD) during corpus training to allow on-line learning of multiple senses of a word with modified versions of Skip-gram. Li and Jurafsky (2015) and Bartunov et al. (2016) apply stochastic processes to allow for representations of a variable number of senses per word to be learnt in unsupervised fashion from corpora.

The embeddings obtained using this approach tend to be word-usage oriented, rather than represent formally defined word senses. While this is descriptive of the texts in the corpus at hand, it can be problematic for generalization. For instance, word senses that are underrepresented or absent in the training corpus will not be assigned a functional embedding. On the other hand, due to the ability of these models to process large amounts of data, well-represented word senses will acquire meaningful representations.

The alternative approach to unsupervised methods is to include data from knowledge resources, usually graph-encoded semantic networks (SN) such as WordNet (Miller 1995). Chen, Liu and Sun (2014) and Iacobacci, Pilehvar and Navigli (2015) propose to make use of knowledge resources to produce a sense-annotated corpus, on which known techniques can then be applied to generate word sense embeddings. A usual way of circumventing the lack of sense-annotated corpora is to apply post-processing techniques onto pre-trained word embeddings as a way of leveraging lexical information to produce word sense embeddings. The following models share this method: Johansson and Nieto Piña (2015) formulate an optimization problem to derive multiple word sense representations from word embeddings, while Pilehvar and Collier (2016) and one of the models proposed by Jauhar, Dyer and Hovy (2015) use graph learning techniques to do so.

A characteristic of this approach is that these models can generate embeddings for a complete inventory of word senses. However, the dependence on manually crafted resources can potentially lead to incompleteness, in case of unlisted word senses, or to inflexibility in the face of changes in meaning, failing to account for new meanings of a word.

The model that we present in this article tries to preserve desirable characteristics from both approaches. On one side, the model learns word sense embeddings from a corpus using a predictive learning algorithm that is efficient, streamlined, and flexible with respect to being able to discriminate between different usages of a word from running text. This learning algorithm is based on the idea of adding an extra latent variable to the Skip-gram objective function to account for different senses of a word, that has been explored in previous work by Jauhar, Dyer and Hovy (2015) and Nieto Piña and Johansson (2015). On the other side, the learning process is guided by a regularizer function that introduces information from an SN, in an attempt to achieve a clear, complete, and fair division between the different senses of a word. Furthermore, from a technical point of view, the effect of the regularizer function is applied in parallel to the embedding learning process. This eliminates the need for a two-step training process or pre-trained word embeddings, and makes it possible to regulate the influence that each source of data (corpus and SN) has on the learning process.

8.3 Model description

8.3.1 Learning word sense embeddings

The Skip-gram word embedding model (Mikolov et al. 2013b) works on the premise of training the vector for a word w to be able to predict those context words c_i with which it appears often together in a large training corpus, according to the following objective function:

$$\sum_{i=1}^{n} \log p(c_i|w)$$

where $p(c_i|w)$ can be approximated using the softmax function, The model, thus, works by maintaining two separate vocabularies which represent word forms in their roles as *target* and *context* words. The resulting word embeddings (usually those vectors trained for the target word vocabulary) are able to store meaningful semantic information about the words they represent.

The original Skip-gram model is, however, limited to word forms in both its vocabularies. Nieto Piña and Johansson (2015) introduced a modification of this model in which the target vocabulary holds a variable number of vectors for each word form, intended to represent its different senses. The training objective of such a model now has the following shape:

$$\log p(s|w) + \sum_{i=1}^{n} \log p(c_i|s) \tag{8}$$

Thus the word sense embeddings are trained to maximize the log-probability of context words c_i given a word's sense s plus the log-probability of that sense given the word w. For our purposes, this prior is a constant, $p(s|w) = \frac{1}{n}$, as we do not have information on the probability of each sense of a given word.

This formulation requires a sense s of word w to be selected for each instance in which the objective function above is applied. This word sense disambiguation is applied on-line at training time and based on the target word's context: The sense s chosen to disambiguate an instance of w is the one whose embedding maximizes the dot product with the sum of the context words' embeddings.

$$\arg\max_{s} \frac{e^{s \sum_i c_i}}{\sum_s e^{s \sum_i c_i}} \tag{9}$$

This unsupervised model learns different usages of a word with minimal overhead computation on top of the original, word-based Skip-gram. The number of senses per word can be obtained from a lexicon or set to a fixed number.

8.3.2 Embedding a lexicon

In order to adapt the graph-structured nature of the data in an SN to be used in continuous representations, we propose to introduce it through a regularizer that can act upon the same embeddings trained by the unsupervised model described above.

Any given node s in a graph will have a set of neighbors n_i directly connected to it. In the graph underlying an SN, we assume n_i to be lexically or semantically similar to s. In this setting, a collection of sequences composed of word senses s and n_i can be collected by visiting all nodes in the SN's graph and collecting its immediate neighbors. Note that extracting such a collection of sequences from a semantic graph follows quite naturally, but in fact it could be generated from any other resource that relates concepts, such as a thesaurus, even if it is not encoded in a graph, as long as the relations it contains are relevant to the model being trained.

We propose to use a collection of sequences of related word senses to update their corresponding word sense vectors by pulling any two vectors closer together in their geometric space whenever they are encountered in a sequence. This action can be easily modeled by minimizing the following expression:

$$\sum_{i=1}^{k} ||s - n_i||^2 \tag{10}$$

for each sequence of word senses $(s, n_1, n_2, \ldots, n_k)$. By minimizing the distance in the vector space between vectors representing interconnected concepts according to the SN's organization, the vector model is effectively representing that organization in a way that geometrical distance correlates with lexical or semantical relatedness, a central concept in the word embedding literature.

8.3.3 Combined model

The two preceding sections describe the two parts of a combined model that is able to learn simultaneously from a corpus and an SN. This is achieved by training embeddings from a corpus with the objective described in equation 8, and complementing this procedure with lexicographic data by means of using equation 10 as a regularizer. The extent of the regularizer's influence on the model is adapted by a mix parameter $\rho \in [0,1]$: the higher the value of ρ, the more influence the SN data has on the model, and vice versa.

Thus, the objective function of our model is as follows:

$$\log p(s|w) + (1-\rho)\sum_{i=1}^{n} \log p(c_i|s) - \rho \sum_{j=1}^{m} ||s - n_j||^2$$

In practice, this objective is realized by alternating updates through each of the model's parts, the number of which is regulated by ρ. Updates on the corpus-based part are executed with Skip-gram with negative sampling (Mikolov et al. 2013b), adapted to work with a vocabulary of word senses as explained in section 8.3.1.

On top of the formulation of the lexicon-based part of the model given in the previous section we propose two variations on this model in order to explore the extent to which the SN data can be used to influence the combined model explained in the following section. The initial formulation of the model will be referenced as V0 in this paper.

In the first variation (henceforth V1) we propose to only apply equation 10 on word senses pertaining to polysemous words. If by using the SN we intend to learn clear separations between different senses of a word, it attends to reason to limit its application to those cases, while monosemous words can be sufficiently well trained by the usual corpus-based approach, and act as semantic anchors in the broader vector space.

The second variation (henceforth V2) deals with the specific architecture of the corpus-based training algorithm. As mentioned in the previous section, this model trains a target and a context vocabulary. We propose to use the regularizer to act not only on word sense vectors, but also on context (word form) vectors. By doing this we expect the context vocabulary to be ready for instances of different senses of a word, training context vectors to be potentially more effective in the disambiguation scheme introduced in equation 9. This variation introduces an extra term into equation 10,

$$\sum_{i=0}^{n} ||w(s) - w(n_i)||^2$$

where $w(x)$ is a mapping from a given sense x to its corresponding word form.

8.4 Experiments

8.4.1 Experimental setting

We trained the three variants of our model using different parameterizations of $\rho \in (0,1)$. Each of these instances learned target and context embeddings of 50 dimensions, using a window of size 5 on the corpus-based part of the training algorithm, for a total number of 5 iterations over a number of updates equal to the size of the training corpus.

Below we describe the lexicon and corpus used to train the sense embeddings.

8.4.1.1 SALDO: a Semantic Network of Swedish Word Senses

SALDO (Borin, Forsberg and Lönngren 2013) is the largest graph-structured semantic lexicon available for Swedish. The version used here contains roughly 125,000 concepts (word senses) organized into a single semantic network.

The sense nodes in the SALDO network are connected by edges that are defined in terms of semantic *descriptors*. A descriptor of a sense is another sense used to define its meaning. The most important descriptor is called the *primary* descriptor (PD), and since every sense in SALDO (except an abstract root sense) has a single unique PD, the PD subgraph of SALDO forms a tree. In most cases, the PD of a sense s is a hypernym or a synonym of s, but other types of semantic relations are also possible.

To exemplify, figure 8.1 shows a fragment of the PD tree. In the example, there are some cases where the PD edges correspond to hypernymy, such as *hard rock* being a type of *rock music*, which in turn is a type of *music*, but there are also other types of relations, such as *music* being defined in terms of *to sound*.

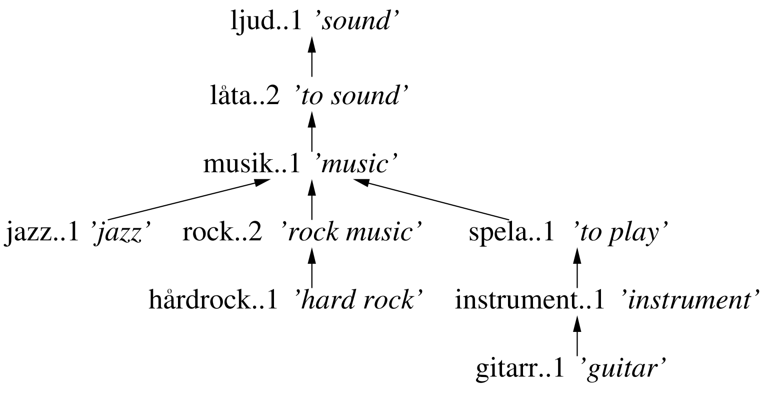


Figure 8.1: A fragment of the network in SALDO.

8.4.1.2 Training corpus

For training the embedding models, we created a mixed-genre corpus of approximately 1 billion words downloaded from Språkbanken, the Swedish language bank.[9] The texts were tokenized, part-of-speech-tagged and lemmatized. Compounds were segmented automatically and when a compound-word lemma was not listed as an entry in the SALDO lexicon, we used the compound parts instead. For instance, *hårdrock* 'hard rock' would occur as a single token in the corpus, while *rockstjärna* 'rock star' would be split into two separate tokens.

8.4.2 Qualitative inspection of word senses

By inspecting lists of nearest neighbors to a given embedding, some insight can be gained into how a model represents the meaning of the concept it represents. It is especially interesting in the case of polysemous words, where the neighbors of each of its senses can help judging how well it manages to separate their different meanings.

In table 8.1 we list nearest neighbors for each of the two senses of the Swedish word *rock*: 'coat' and 'rock music'. The neighboring concepts in the table are extracted from two separate vector models trained with

[9]`http://spraakbanken.gu.se`

rock-1 ‘coat’

ρ = 0.01	**ρ = 0.5**
syrtut ‘frock coat’	*syrtut* ‘frock coat’
Rhythm ‘rhythm music’	*kappa* ‘coat’
rockband ‘rock band’	*kåpa* ‘cowl’
Peepshows ‘peep shows’	*päls* ‘fur coat’
skaband ‘ska band’	*mudd* ‘cuff’

rock-2 ‘rock music’

ρ = 0.01	**ρ = 0.5**
hårdrock ‘hard rock music’	*punk* ‘punk music’
pop ‘pop music’	*rappa* ‘to rap’
punk ‘punk music’	*rap* ‘rap music’
jazza ‘to jazz’	*pop* ‘pop music’
dödsmetall ‘death metal music’	*jam* ‘music jam’

Table 8.1: Nearest neighbors for the two senses of *rock* ‘coat’ and ‘ rock music’ for different ρ.

different parameterizations for the mix parameter ρ: The first, $\rho = 0.01$, has little influence from the lexicon and thus is similar to a corpus-only approach; the second, $\rho = 0.5$, allows for more information from the lexicon to influence the embeddings. In our corpus, the music sense is overrepresented; this can be seen in the table, where both senses trained with $\rho = 0.01$ have most of their nearest neighbors semantically related to music. The model that is more influenced by the lexicon with $\rho = 0.5$ is, however, able to learn two distinct senses. Note how the music sense is not negatively affected by this change: many of its nearest neighbors are the same in both models, and all of them keep the music-related topic in common.

It is also interesting to filter these lists of nearest neighbors to limit them to unlisted words; i.e., words that are not present in the lexicon and appear only in the corpus. This provides an observation of how well those embeddings that are trained by both parts of the model are integrated with those others whose training is based only on the corpus. Table 8.2 contains such lists of unlisted items for the two senses of *rock* on two models with different parameterization. It presents a similar behavior to the previous experiment: In a model with low influence from the lexicon, the representations of both senses tend towards that of the overrepresented one; when more influence from the lexicon is allowed, a clear separation of the two senses into their expected meanings is observed.

rock-1 'coat'

ρ = 0.01	**ρ = 0.5**
Rhythm 'rhythm music'	*jesussandaler* 'Jesus sandals'
Peepshows 'peep shows'	*tubsockar* 'tube socks'
skabandk 'ska band'	*blåjeans* 'blue jeans'
Punkrock 'punk rock'	*snowjoggers* 'snow joggers'
sleaze 'to sleaze'	*midjekort* 'doublet jacket'

rock-2 'rock music'

ρ = 0.01	**ρ = 0.5**
nu-metal 'nu metal'	*metal* 'metal music'
goth ' goth music'	*rnb* 'RnB music'
psytrance ' psytrance music'	*indie* 'indie music'
boogierock 'boogie rock'	*dubstep* 'dubstep music'
synthband 'synth music band'	*goth* 'goth music'

Table 8.2: Nearest unlisted neighbors for the two senses of *rock* 'coat' and 'rock music' for different ρ.

8.4.3 Word sense disambiguation

We trained and evaluated several parameterizations of our model on a Swedish language word sense disambiguation (WSD) task. The aim of this task is to select a sense of an instance of a polysemous word in context. For this purpose, we use a disambiguation mechanism similar to the one introduced in section 8.3.1. Given an ambiguous word in context, a score is calculated for each of its possible senses by applying the expression in equation 9; however, to correct for skewed sense distributions, we replaced the uniform prior with a power-law prior $P(s_k|w) \propto k^{-2}$, where k is the numerical identifier of the sense. The highest scoring sense is then selected to disambiguate that instance of the word.

As baselines for this experiment, we used random sense and first sense[10] selection. Additionally, we show the results achieved by a disambiguation system, UKB, based on Personalized PageRank (Agirre and Soroa 2009), and which was trained on the PD tree from SALDO. The implementation of this model makes no assumptions on the underlying graph and thus it is easily adaptable to work with any kind of SN. Our

[10]No frequency information is available for SALDO's sense inventory and the senses are not ordered by frequency. The senses are ordered by lexicographers so that the lower-numbered senses are more "central" or "primitive", which often but not always correlates with the sense frequency.

models were all parameterized with $\rho = 0.9$ based on the results obtained on the SweFN dataset. All evaluated systems including the baselines are unsupervised: none of them has used a sense-annotated corpus during training.

8.4.3.1 Sense-annotated Datasets

We evaluated the WSD systems on eleven different datasets, which to our knowledge are all sense-annotated datasets that exist for Swedish. The datasets consist of instances, where each instance is a sentence where a single target word has been selected for disambiguation.

Two datasets consist of *lexicographical examples* (Lex-Ex): the *SALDO examples* (SALDO-ex) and *Swedish FrameNet examples* (SweFN-ex). The latter of these is annotated in terms of semantic frames, but there is a deterministic mapping from frames to SALDO senses.

Two additional datasets are taken from the Senseval-2 Swedish lexical sample task (Kokkinakis, Järborg and Cederholm 2001). It uses a different sense inventory, which we mapped manually to SALDO senses. The lexical sample originally consisted of instances for 40 lemmas, out of which we removed 7 lemmas because they were unambiguous in SALDO. Since we are using an unsupervised experimental setup, we report results not only on the designated test set but also on the training set.

The other datasets come from the *Koala* annotation project (Johansson et al. 2016). The latest version consists of seven different corpora, each sampled from text in a separate domain: blogs, novels, Wikipedia, European Parliament proceedings, political news, newsletters from a government agency, and government press releases. Unlike the two lexicographical example sets and the Senseval-2 lexical sample, in which the instances have been selected by lexicographers to be prototypical and to have a good coverage of the sense variation, the instances in the Koala corpora are annotated 'as is' in running text.

The sentences in all datasets were tokenized, compound-split, and lemmatized, and for each target word we automatically determined the set of possible senses, given its context and inflection. We only considered senses of content words: nouns, verbs, adjectives, and adverbs. Multi-word targets were not included, and we removed all instances where only one sense was available.[11]

[11]In addition, to facilitate a comparison to the UKB system as a baseline, we removed a small number of instances that could not be lemmatized unambiguously.

Test set	**Subset**	**Size**	**RND**	**S1**	**UKB**	**V0**	**V1**	**V2**
Lex-Ex	Average	2,365	39.86	53.90	54.76	61.23	**61.26**	58.34
	SweFN-Ex	1,197	40.43	54.80	54.64	60.90	**61.90**	59.06
	SALDO-Ex	1,168	39.29	53.00	54.88	**61.56**	60.62	57.62
Senseval	Average	8,237	35.90	50.36	44.37	**54.29**	52.95	53.61
	Train	6,995	35.98	50.48	45.43	**54.40**	53.57	53.11
	Test	1,242	35.83	50.24	43.32	**54.19**	52.33	54.11
Koala	Average	11,167	41.83	69.50	67.17	65.17	**73.49**	68.59
	Blogs	2,222	41.86	**71.02**	66.70	60.98	67.78	64.27
	Europarl	1,838	41.80	66.16	65.61	61.26	**71.60**	68.28
	Novels	2,446	41.04	72.85	67.46	67.95	**73.47**	71.30
	Wikipedia	2,444	42.50	75.98	67.59	73.65	**76.68**	73.98
	Political news	1,082	40.60	69.41	69.04	67.47	**75.69**	69.59
	Newsletters	280	42.04	63.57	65.00	58.93	**73.21**	64.29
	Press releases	855	42.99	67.49	68.77	65.96	**76.02**	68.42
Total		21,769	40.40	63.18	60.77	62.48	**67.53**	64.00

Table 8.3: WSD accuracy on baselines, UKB, and the three variants of our model ($\rho = 0.9$) on all test sets.

8.4.3.2 Disambiguation results

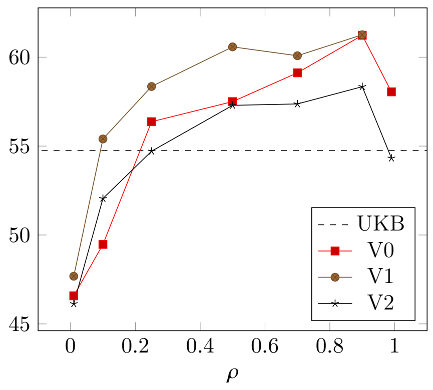
Table 8.3 shows disambiguation accuracies for our models on the datasets described above, along with the scores achieved by our baselines and the UKB model. The results of each variant of our model were obtained with a parameterization of $\rho = 0.9$, which was chosen as the best scoring value on the Swe-FN subset used as validation set. The model which only applies the regularizer to polysemous words (V1) dominates most highest scores, overtaken in some instances by V0 and in one by the first sense baseline. Note how the general magnitudes of the scores within each type of dataset underline their different characteristics explained above.

Additionally, for the sake of making a more detailed analysis of the influence of the parameter ρ that dominates the extent of the lexicon's influence on the model, figure 8.2 shows the average performance of our models on each dataset for a wide range of values for ρ. There is a clear pattern across all models and datasets by which a greater input from the SN translates into a better performance in WSD. These figures also confirm the superior performance of the variant V1 of our model seen in table 8.3.

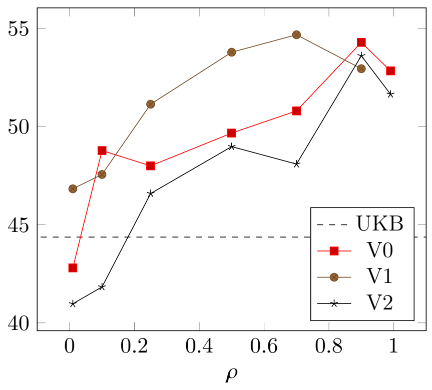
8.4.4 Frame prediction

In our second evaluation, we investigated how well the sense vector models learned by the different training algorithms correspond to semantic classes defined by the Swedish FrameNet (Friberg Heppin and Gronostaj Toporowska 2012). In a frame-semantic model of lexical meaning (Fillmore and Baker 2009), the meaning of words is defined by associating them with broad semantic classes called *frames*; for instance, the word *falafel* would belong to the frame FOOD. Important classes of frames include those corresponding to objects and people, mainly populated by nouns, such as FOOD or PEOPLE_BY_AGE; verb-dominated frames corresponding to events, such as IMPACT, STATEMENT, or INGESTION; and frames dominated by adjectives, often referring to relations, qualities, and states, e.g. ORIGIN or EMOTION_DIRECTED.

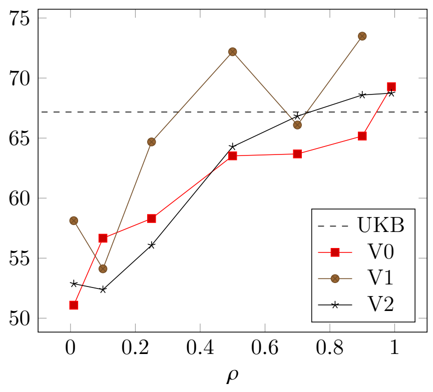
In case a word has more than one sense, it may belong to more than one frame. In the Swedish FrameNet, unlike its English counterpart, these senses are explicitly defined using SALDO (see section 8.4.1.1): for instance, for the highly polysemous noun *slag*, its first sense ('type') belongs to the frame TYPE, the second ('hit') to IMPACT, the third ('battle') to HOSTILE_ENCOUNTER, etc.



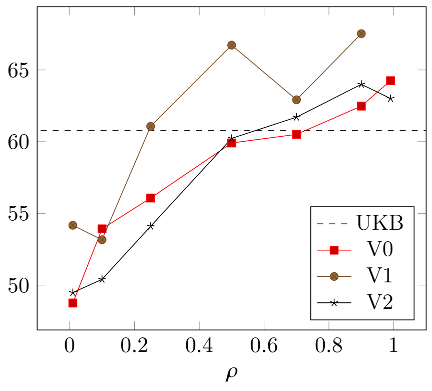
(a) Lex-Ex.



(b) Senseval.



(c) Koala.



(d) All test sets.

Figure 8.2: Average WSD accuracies on all instances of each dataset for different values of ρ on the three variants of our model.

In the evaluation, we trained classifiers to determine whether a SALDO sense, represented as a sense vector, belongs to a given frame or not. To train the classifiers, we selected the 546 frames from the Swedish FrameNet for which at least 5 entries were available. In total we had 28,842 verb, noun, adjective, and adverb entries, which we split into training (67% of the entries in each frame) and test sets (33%). For each frame, we used LIBLINEAR (Fan et al. 2008) to train a linear support vector machine, using the vectors of the senses associated with that frame as positive training instances, and all other senses listed in FrameNet as negative instances.

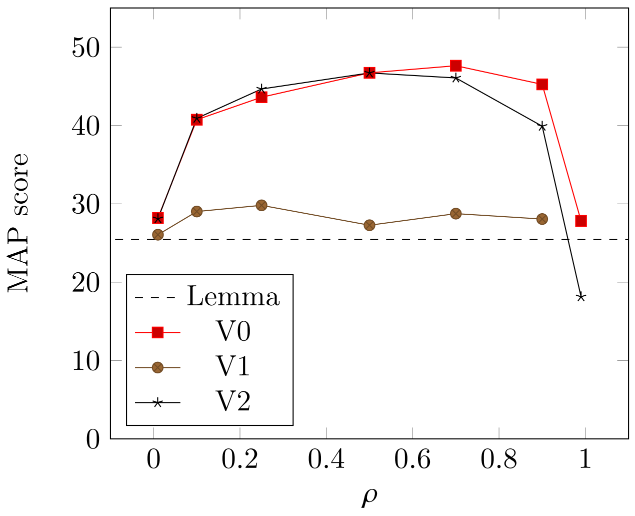


Figure 8.3: MAP scores for the frame prediction classifiers for the different types of models.

8.4.4.1 Evaluation Results

At test time, for each frame we applied the SVM scoring function of its classifier to each sense in the test set. The ranking induced by this score was evaluated using the Average Precision (AP) metric commonly used to evaluate rankers; the goal of this ranking step is to score the senses belonging to the frame higher than those that do not. We computed the Mean Averaged Precision (MAP) score by macro-averaging the AP scores over the set of frames.

Figure 8.3 shows the MAP scores of frame predictors based on different sense vector models. We compared the three training algorithms described in section 8.3 for different values of the regularization strength parameter ρ. As a baseline, we included a model that does not distinguish between different senses: it represents a SALDO sense with the word vector of its lemma.

As the figure shows, almost all sense-aware vector models outperformed the model that just used lemma vectors. The result shows tendencies that are different from what we saw in the WSD experiments. The best MAP scores were achieved with mid-range values of ρ, so it seems that this task requires embeddings that strike a balance between representing the lexicon structure faithfully and representing the cooccurrence patterns in the corpus. A model with very light influence of the lexicon was hardly better than just using lemma embeddings, and unlike what we saw for the WSD task we see a strong dropoff when increasing ρ.

In addition, the tendencies here differ from the WSD results in that the training algorithm that only applies the lexicon-based regularizer to polysemous words (V1) gives lower scores than the other two approaches. We believe that this is because it is crucial in this task that sense vectors are clustered into coherent groups, which makes it more useful to move sense vectors closer to their neighbors even when they are monosemous; this as opposed to the WSD task, where it is more useful to leave the monosemous sense vectors in place as "anchors" for the senses of polysemous words. The context-regularized training algorithm (V2) gives no improvement over the original approach (V0), which is expected since context vectors are not used in this task.

Frame	**Lemma**	**V0**	**V1**
ANIMALS	0.73	**0.86**	0.76
FOOD	0.72	**0.84**	0.77
REMOVING	0.20	**0.50**	0.22
MAKE_NOISE	0.40	**0.62**	0.46
ORIGIN	0.90	**0.90**	0.89
COLOR	0.73	**0.88**	0.80
FREQUENCY	0.40	**0.43**	0.35
TIME_VECTOR	0.40	**0.52**	0.27

Table 8.4: Frame prediction AP scores for selected frames dominated by nouns, verbs, adjectives, and adverbs, respectively.

To get a more detailed picture of the strengths and weaknesses of the models in this task, we selected eight frames: two frames dominated by nouns, two for verbs, two for adjectives, two for adverbs. Table 8.4 shows the AP scores for these frames of the lemma-vector baseline, the initial approach (V0), and the version that only regularizes senses of polysemous words (V1). All lexicon-aware models used a ρ value of 0.7. Almost across the board, the V0 method gives very strong improvements. The exception is the frame ORIGIN, which contains adjectives of ethnicity and nationality (*Mexican*, *African*, etc); this set of adjectives is already quite coherently clustered by a simple word vector model and is not substantially improved by any lexicon-based approach.

8.5 Conclusion

In this article we have introduced a family of word sense embedding models that are able to leverage information from two concurrent sources of information: a semantic network and a corpus. Our hypothesis was that by combining them, the robustness and coverage of embeddings trained on a large corpus could achieve a more balanced and linguistically informed representation of the senses of polysemic words. This point has been proved in the evaluation of our models on Swedish language tasks.

A manual inspection of the word sense representation through their nearest neighbors exemplified it in section 8.4.2. Indeed, an increased influence from the SN causes a clearer distinction between different senses of a word, even in the case where one of them is underrepresented in the corpus.

A WSD experiment was carried out on a variety of sense-annotated datasets. Our model consistently outperformed random and first sense baselines, as well as a comparable graph-based WSD system trained on a Swedish SN, which underlines the fact that the strength of our model resides in a combination of lexicon- and corpus-learning.

This is further confirmed in the evaluation of our model on a frame prediction task: A well balanced combination of lexicon and corpus data produces word sense embeddings that outperform common word embeddings when used to predict their semantic frame membership. Furthermore, this superiority is uniform across common frames dominated by different parts of speech.

An analysis of different values of our model's mix parameter ρ showed the value of using lexicographic information in conjunction with corpus data. Especially on WSD, larger values of ρ (i.e., more influence from the SN) generally lead to improved results.

In conclusion, we have shown that automatic word sense representation benefits greatly from using a semantic network in addition to the usual corpus-learning. The combination of these sources of information yields robust, high-quality, and balanced embeddings that excel in downstream tasks where accurate representation of word meaning is crucial. Given these findings, we intend to continue exploring more refined ways in which data from a semantic network can be leveraged to increase sense-awareness in embedding models.

Acknowledgments

This research was funded by the Swedish Research Council under grant 2013–4944. The Koala corpus was developed in a project funded by Riksbankens Jubilemsfond, grant number In13-0320:1.

9 Automatically linking lexica with word sense embeddings

This chapter is a postprint version of the following publication:

Luis Nieto Piña and Richard Johansson 2018. Automatically Linking Lexical Resources with Word Sense Embedding Models. *Proceedings of the third workshop on semantic deep learning (SemDeep-3), COLING 2018,* 23–29. Association for Computational Linguistics. Santa Fe, USA.

Abstract

Automatically learnt word sense embeddings are developed as an attempt to refine the capabilities of coarse word embeddings. The word sense representations obtained this way are, however, sensitive to underlying corpora and parameterizations, and they might be difficult to relate to word senses as formally defined by linguists. We propose to tackle this problem by devising a mechanism to establish links between word sense embeddings and lexical resources created by experts. We evaluate the applicability of these links in a task to retrieve instances of Swedish word senses not present in the lexicon.

9.1 Introduction

Word embeddings have boosted performance in many Natural Language Processing applications in recent years (Collobert et al. 2011; Socher et al. 2011). By providing an effective way of representing the meaning of words, embeddings facilitate computations in models and pipelines that need to analyze semantic aspects of language.

Based on their success, an effort has been concentrated in improving embedding models, from devising more computationally effective models to extending them to cover other semantic units beyond words, such as multi-word expressions (Yu and Dredze 2015) or word senses (Neelakantan et al. 2014). Being able to represent word senses solves the problem of conflating several meanings of one polysemic word into a single embedding (Li and Jurafsky 2015). Furthermore, having complete and accurate word sense representations brings embedding models closer to a range of existing, expert-curated resources such as lexica. Bridging the gap between these two worlds arguably opens a road to new methods that could benefit well-established, widely used resources (Faruqui et al. 2015; Speer, Chin and Havasi 2017).

This is the focus of the work we present in this article. We propose an automatic way of creating a mapping between entries in a lexicon and word sense representations learned from a corpus. By having an identification between a manual inventory of word senses and word sense embeddings, the lexicon obtains capabilities by which its entries can be manipulated as mathematical objects (vectors), while the vector space model receives support from a linguistic resource. Furthermore, by analyzing the disagreements between the lexicon and the embedding model, we can acquire insight into the shortcomings of their respective coverage. For instance, unlinked lexicon entries evidence those instances that the vector model is unable to learn, while unlinked word sense embeddings may suggest new usages of words found in the corpora. Being able to locate these cases opens the way towards improving lexica and embedding models. Automatic discovery of novel senses has been shown as a feasible and productive endeavor (Lau et al. 2012). In our evaluation we provide some insight into these situations: we use a mapping to calculate the probability that a word in a sentence from a corpus is an instance of an unlisted sense (i.e., not present in the lexicon.)

This mapping process, and some of its potential applications, are explained in detail in the following sections. Section 9.2 contains a description of the mapping mechanism. Section 9.3 goes on to evaluate the performance of this mapping on finding instances of unlisted senses. We present our conclusions in section 9.4.

9.2 Model

9.2.1 Lexicon

A lexicon which lists word senses and provides relations between them is required; for instance, a resource that encodes these relations in a graph architecture, such as WordNet (Miller 1998).

We need to retrieve word senses related to any given target word sense in order to obtain a set of *neighbors* which put the target sense in context. This context will be used to compare it with sets of neighbors extracted from a word sense vector space for senses of the same lemma, and thus find the best match to establish a link.

In our experiments on Swedish data we use SALDO (Borin, Forsberg and Lönngren 2013) as our lexicon. It is encoded as a network: nodes encode word senses, which are connected by edges defined by semantic *descriptors*: the meaning of a sense is defined by one or more senses, its descriptors. Among others, each sense has one *primary descriptor* (PD) and, in turn, it may be the primary descriptor of one or more senses. E.g., the PD of the musical sense of *rock* would be *music*, which would be the PD of *hard rock*. The PD network of SALDO traces back to a *root node* (which does not have a PD) and, thus, it has a tree topology. In general, senses with more abstract meanings are closer to the root, while more concrete ones are located closer to the leaves.

9.2.2 Word sense embeddings

Word sense embeddings allow us to create an analogy between semantic relatedness among words and geometric distance between their representations in a vector space. In particular, a word sense embedding model assigns multiple representations to any given word, each of which is related to a distinct word sense. For our purposes, we make use of Adaptive Skip-gram (Bartunov et al. 2016), an adaptation of the hierarchical softmax Skip-gram (Mikolov et al. 2013b).

The hierarchical softmax model is described by its training objective: to maximize the probability of context words v given a target word w and the model's parameters θ:

$$p(v|w,\theta) = \prod_{n\in \mathrm{path}(v)} \frac{1}{1 - e^{-\mathrm{ch}(n) x_w^\top y_n}}, \tag{11}$$

where x_w are the input representations of the target words w, and the original output representations of context words y_n are associated with nodes in a binary tree which has all possible vocabulary words v as leaves; θ is the set of these representations as weights of the vector model. In this context, path(v) are the nodes n in the path from the tree's root to the leaf v, identified by ch(n) being -1 or 1 depending on whether n is a right or left child.

The Adaptive Skip-gram (AdaGram) model expands this objective to account for multiple word senses to be represented in the input vocabulary X. It does so by introducing a latent variable z that determines a concrete sense k of word w. The output vocabulary Y remains unchanged. This model uses a Dirichlet process (*stick-breaking representation*) to automatically determine the number of senses per word. They define a prior over the multiple senses of a word as follows:

$$p(z = k|w, \boldsymbol{\beta}) = \beta_{wk} \prod_{r=1}^{k-1} (1 - \beta_{wr}),$$

$$p(\beta_{wk}|\alpha) = \mathrm{Beta}(\beta_{wk}|1, \alpha),$$

where $\boldsymbol{\beta}$ is a set of samples β from the Beta distribution and α is a hyperparameter. By combining this prior with a pre-specified maximum number of prototypes and a threshold probability, the model is able to automatically determine the number of word senses any given word is expected to have. The objective function that defines the AdaGram model is defined as follows:

$$p(Y, Z, \boldsymbol{\beta}|X, \alpha, \theta) = \prod_{w=1}^{V} \prod_{k=1}^{\infty} p(\beta_{wk}|\alpha) \prod_{i=1}^{N} \left[p(z_i|x_i, \boldsymbol{\beta}) \prod_{j=1}^{C} p(y_{ij}|z_i, x_i, \theta) \right], \tag{12}$$

where V is the word vocabulary size, N is the size of the training corpus, C is the size of the context window, and $p(\beta_{wk}|\alpha)$ is the prior over multiple senses obtained via the Dirichlet process described above. The granularity in the distinction between word senses is controlled by α. A trained model produces representations for word senses in a D-dimensional vector space where those with similar meanings are closer together than those with dissimilar ones.

For the purposes of this work, we trained a word sense embedding model on a Swedish corpus (cf. section 9.3.1) using the default parameterization of AdaGram: 100-dimension vectors, maximum 5 prototypes per word, $\alpha = 0.1$, and one training epoch. (See AdaGram's documentation for a complete list.)

9.2.3 Lexicon-embedding mapping

Our goal is to establish a mapping between lexicographic word senses and embeddings that represent the same meanings. The approach we take is to generate a set of related word senses for each sense of any

Lexicon	**Vector space**
rock-1 'jacket'	*rock-b* (clothing)
rock-2 'rock music'	*rock-a*, *rock-c*, *rock-d*, *rock-e* (music)

Table 9.1: Example mapping for *rock*.

given word, both from the lexicon (via relations encoded in its graph's edges) and from the vector space (via cosine distance), to measure their relatedness and define mappings.

To obtain sets of neighbors from the lexicon, given any target word sense, any senses directly connected by an edge to the target's node is selected as a neighbor. In the vector space, nearest neighbors based on cosine distance are selected.

In order to measure relatedness using geometric operations we need to assign *provisional* embeddings to lexicographic neighbors in order to measure their distance to vector space neighbors. We do this by using AdaGram's disambiguation tool: given a target word and a set of context words, it calculates the posterior probability of each sense of the target word given the context words according to the hierarchical softmax model (equation 11). The word in context is disambiguated by selecting the sense with the highest posterior probability. (In our case, for any given sense, the rest of senses in the set act as context.)

We expect some lexicon-defined senses to not be present in the vector space and some word senses captured by the embedding model to not be listed in the lexicon. Additionally, the AdaGram model may create two or more senses for one word which relate to the same lexicographic sense. We address this by making the mapping 1-to-N to allow a lexicon sense to be linked to more than one sense embedding if necessary. Additionally, in order to make it possible for lexicon senses to be left unlinked, we propose to use a *false sense embedding* that acts as an attractor for those lexicon senses with weak links to real embeddings.

The mapping mechanism is shown in algorithm 3. For each word in the vocabulary, a set of neighbors is generated for each of its senses in the lexicon and in the vector space. The vector representations of these neighbors are averaged, since averaging embeddings has been proven an efficient approach to representing multi-word semantic units (Mikolov et al. 2013b; Kenter, Borisov and de Rijke 2016). A probability matrix $p \in [0,1]^{n \times m}$ is calculated by applying the softmax function to pairs of

Algorithm 3: Mapping algorithm.

```
for all words $w$ do
    /* Lexicon neighbors */
    $n \leftarrow$ #lexical senses of $w$
    $s_i \leftarrow i$th lexical sense of $w$, $i \in [1, n]$
    for all $s_i$ do
        $a_i \leftarrow$ set of neighbors of $s_i$
        for all neighbor $k$ in $a_i$ do
            $v(a_{ik}) \leftarrow$ embedding for $a_{ik}$
        end for
    end for
    /* Vector space neighbors */
    $T \leftarrow$ max #senses per word
    $m \leftarrow$ #nearest neighbors (NN) per sense
    $z_j \leftarrow j$th sense embedding of $w$, $j \in [1, T]$
    for all $z_j$ do
        $b_{jl} \leftarrow l$th NN of $b_j$, $l \in [1, m]$
    end for
    /* Average neighbors */
    for all $i$, $j$ do
        $v(a_i) \leftarrow$ avg. vector over $k$
    end for
    for all $j$ do
        $v(b_j) \leftarrow$ avg. vector over $l$
    end for
    /* Mapping probabilities */
    for all $i$, $j$ do
        $p_{ij} \leftarrow \mathrm{softmax}(v(a_i) \cdot v(b_j))$
        $p_{iN+1} \leftarrow \mathrm{softmax}(\overrightarrow{0})$
    end for
    /* Mapping */
    for $j \in [1, T]$ do
        if prior$(s_j) > \rho$ then
            $r \leftarrow \mathrm{indmax}_i(p_{ij})$
            if $r \neq N + 1$ then
                map $s_r$ to $z_j$
            end if
        end if
    end for
end for
```

vectors. An extra column is added with scores generated by the softmax on zero-valued vectors to account for the false sense. A row in p represents the probability that each sense in the vector model corresponds to that row's lexicon sense, from which the maximum is obtained to establish a link. (A threshold ρ exists to avoid low-probability links.)

An example of the linking performed by this mechanism is shown on table 9.1. According to the lexicon SALDO, the noun *rock* has two senses in Swedish: (1) 'jacket' and (2) 'rock music'. The word sense embedding model finds five different meanings for *rock*; upon inspection of their nearest neighbors, which give an indication of the closest senses to any particular meaning, four of them relate to the music sense (linked to *rock-2*) and one to items of clothing (linked to *rock-1*). As can be seen in table 9.1, the clothing-related meaning is linked to the sense meaning 'jacket', while the four music-related ones is linked to the sense meaning 'rock music'.

9.3 Evaluation

9.3.1 Training corpus

For training the AdaGram model, we created a mixed-genre corpus of approximately 1 billion words of contemporary Swedish downloaded from Språkbanken, the Swedish language bank.[12] The texts were processed using a standard preprocessing chain including tokenization, part-of-speech-tagging and lemmatization. Compounds were segmented automatically and when the lemma of a compound word was not listed in SALDO, we used the lemmas of the compound parts instead. For instance, *golfboll* 'golf ball' would occur as a single lemma in the corpus, while *pingisboll* 'ping-pong ball' would be split into two separate lemmas.

9.3.2 Benchmark dataset

For evaluation, we annotated a benchmark dataset. We selected five target lemmas for which we knew that the corpus contains occurrences of word senses that are unlisted in the lexicon. In addition, we selected four target lemmas that do not strictly have new word senses, but that tend to be confused with tokenization artifacts, named entities, or foreign words.

[12]`http://spraakbanken.gu.se`

Table 9.2 shows the selected lemma, an overview of the senses that are listed in the lexicon, as well as the most important unlisted senses.

Lemma	**Listed**	**Main unlisted**
fet	‘fat’; ‘thick’	‘cool’; emphasis
klient	‘client’ (customer)	‘client’ (application)
klubb	‘club’	(night) ‘club’
pirat	‘pirate’	‘pirate’ (e.g. music)
sida	‘side’; ‘page’	‘web page’
fil	‘file’; ‘row’; ‘lane’; ‘yogurt’	e.g. in *fil. dr.* ‘PhD’
get	‘goat’	foreign words
mus	‘mouse’; ‘pussy’	a cosmetic brand
sur	‘sour’; ‘grumpy’; ‘soaked’	foreign words

Table 9.2: Selected target lemmas.

For each of these nine target lemmas, we selected 1,000 occurrences randomly from the corpus. Two annotators went through the selected occurrences and determined which of them are instances of the senses present in the lexicon, and which of them are unlisted senses. A small number of occurrences were discarded because they were difficult for the annotators to understand.

9.3.3 Experimental settings

Given a mapping between lexicographic word senses and automatically discovered senses in a corpus, sentences from the benchmark dataset can be scored by their probability of containing an out-of-lexicon instance. A score is calculated using the linking probability between lexicographic sense y_j and vector model sense x_i, $P(y_j|x_i)$, and the probability of the AdaGram sense x_i in the context of the sentence ctx, $P(x_i|ctx)$:

$$P(y_j|ctx) = \sum_{i=1}^{T} P(y_j|x_i)P(x_i|ctx),$$

thus obtaining the probability of a particular lexicographic sense y_j given context ctx. The sum of all $p(y_j|ctx)$, $j \in [1,T]$, yields the probability that an instance contains one of the listed word senses. Our score, then, is calculated as the inverse probability:

$$\text{score} = 1 - \sum_{j=1}^{n} P(y_j|ctx). \tag{13}$$

The human annotations of the sentences are interpreted as the gold standard, and the scores as our model's classification output that can be used to rank the sentences from most to least probably containing an out-of-lexicon instance of a target word. We evaluate this classification using the Area Under the Receiver Operating Characteristic Curve (AUC). For reference, recall that the expected AUC of a random ranking is 0.5.

The potential of the automated mapping between a lexicon and an embedding model to help retrieve instances of word senses unlisted in the lexicon gives us a certain measure of the quality of this mapping. The recovery of instances of unlisted senses can only succeed if the mapping has successfully identified listed lexicon senses in the embedding model, leaving unlisted ones unlinked. On the other hand, failure to recover instances of unlisted senses can expose weaknesses in the automated mapping. (See section 9.3.4 for an analysis of unsuccessful cases.)

9.3.4 Results

We apply the scoring and evaluation process explained above on the sets of sentences for each of the words selected for the benchmark dataset. Table 9.3 summarizes the results obtained. We observe a clear difference

Word	***n***	**AUC**
fet	976	0.76
klient	959	0.02
klubb	985	0.74
pirat	907	0.53
sida	985	0.69
Sub-avg.		0.55

(a) Unlisted senses.

Word	***n***	**AUC**
fil	982	0.87
get	954	0.98
mus	972	0.83
sur	967	0.96
Sub-avg.		0.91
Total avg.		0.73

(b) Spurious senses.

Table 9.3: AUC scores.

between the two types of lemmas: the model's performance is notably higher when the lemmas may be confused with tokenization artifacts, named entities or foreign words (table 9.3b) than when the lemmas have meanings not listed in the lexicon (table 9.3a). This is arguably due to the ability of the word sense embedding model to isolate spurious meanings in the vector space since these occurrences tend to be distributionally quite distinct from the standard uses. However, note that also for the case of unlisted meanings, the model generally improves over a random baseline.

The exception to this is the case of *klient*, where the listed sense is ‘customer’ and the new sense is ‘client application’. Here, the ranking is upside down, as can be seen from the very low AUC value. The cause of this issue can be traced to the inability of the mapping algorithm to successfully link the only lexicographic sense to its corresponding vector space embedding. The neighbors of the AdaGram sense corresponding to the ‘customer’ sense tend to be legal terms, which are not connected to this sense in the lexicon. (Arguably, the legal use of ‘client’ could be seen as a separate sense.) Compare this to the case of *pirat*, which performs roughly at random (AUC ≈ 0.5), suggesting simply that the AdaGram model has not picked up the distinction between the senses, which makes it hard to solve the problem by simply changing the link, as would be the case with *klient*.

A straightforward way to address those cases in which the mapping fails to correctly establish a link would be to refine the way in which the word senses are represented at the time of linking. Our approach is to do this by averaging the embeddings of nearest neighbors, due to the simplicity and usual robustness of this operation. However, more complex approaches to combining embeddings have been demonstrated using, for example, weighted averaging (Arora, Liang and Ma 2017), which could be adapted for our needs in this work.

9.4 Conclusion

We have presented an approach to automatically link lexicographic word senses with word sense embeddings by retrieving sets of senses related to the different meanings of a lemma and measuring the similarity between their vector representations. We argue that potential applications of such a system resides on those embeddings that cannot be linked to a lexicographic sense, as this could serve to suggest potential new entries to the lexicon, or to filter unlisted instances from a corpus. To illustrate this point, the evaluation of our system has been focused on its ability to retrieve instances assumed to belong to unlisted senses. Our system is able to identify such instances in many cases, as its performance in terms of AUC is above that of a random baseline. The performance is specially high in cases where the target lemma can be confused in the corpus with spurious meanings such as foreign words. In summary, our results indicate that establishing links between existing resources and embedding models has potential applications in NLP tasks which require formal lexicographic knowledge. In the future, we propose to examine ways to improve the current mapping system with improved neighbor representation approaches, as well as to investigate further possible uses, such as improving existing resources with data obtained from corpora.

Acknowledgements

This research was funded by the Swedish Research Council under grant 2013–4944.

10 Inter-resource lexical-semantic mapping

This chapter is a postprint version of the following publication:

Lars Borin, Luis Nieto Piña and Richard Johansson 2015. Here be dragons? The perils and promises of inter-resource lexical-semantic mapping. *Proceedings of the workshop on semantic resources and semantic annotation for natural language processing and the digital humanities at NODALIDA 2015,* 1–11. Vilnius, Lithuania.

Abstract

Lexical-semantic knowledges sources are a stock item in the language technologist's toolbox, having proved their practical worth in many and diverse natural language processing (NLP) applications.

In linguistics, lexical semantics comes in many flavors, but in the NLP world, wordnets reign more or less supreme. There has been some promising work utilizing Roget-style thesauri instead, but wider experimentation is hampered by the limited availability of such resources.

The work presented here is a first step in the direction of creating a freely available Roget-style lexical resource for modern Swedish. Here, we explore methods for automatic disambiguation of inter-resource mappings with the longer-term goal of utilizing similar techniques for automatic enrichment of lexical-semantic resources.

10.1 Introduction

10.1.1 The uniformity of lexical semantic resources for NLP

Lexical-semantic knowledges sources are a stock item in the language technologist's toolbox, having proved their practical worth in many and diverse natural language processing (NLP) applications.

Although lexical semantics and the closely related field of lexical typology have long been large and well-researched branches of linguistics (see, e.g., Cruse 1986; Goddard 2001; Murphy 2003; Vanhove 2008), the lexical-semantic knowledge source of choice for NLP applications is WordNet (Fellbaum 1998a), a resource which arguably has been built largely in isolation from the linguistic mainstream and which thus is somewhat disconnected from it.

However, the English-language Princeton WordNet (PWN) and most wordnets for other languages are freely available, often broad-coverage lexical resources, which goes a long way toward explaining their popularity and wide usage in NLP as due at least in part to a kind of streetlight effect.

For this reason, we should certainly endeavor to explore other kinds of lexical-semantic resources as components in NLP applications. This is easier said than done, however. The PWN is a manually built resource, and efforts aiming at automatic creation of similar resources for other languages on the basis of PWN, such as Universal WordNet (de Melo and Weikum 2009) or BabelNet (Navigli and Ponzetto 2012), although certainly useful and laudable, by their very nature will simply reproduce the WordNet structure, although for a different language or languages. Of course, the same goes for the respectable number of manually constructed wordnets for other languages.[13]

Manually built alternatives to wordnets are afflicted by being for some other language than English (e.g., SALDO: Borin, Forsberg and Lönngren 2013) or by not being freely available – see the next section – or possibly both.

10.1.2 Roget's *Thesaurus* and NLP

While wordnets completely dominate the NLP field, outside it the most well-known lexical-semantic resource for English is without doubt Roget's *Thesaurus* (also alternately referred to as "Roget" below; Roget 1852; Hüllen 2004), which appeared in its first edition in 1852 and has since been published in a large number of editions all over the English-speaking world. Although – perhaps unjustifiedly – not as well-known in NLP as the PWN, the digital version of Roget offers a valuable complement to PWN (Jarmasz and Szpakowicz 2004), which has seen a fair

[13]See the *Global WordNet Association* website: `http://globalwordnet.org`.

amount of use in NLP (e.g., Morris and Hirst 1991; Jobbins and Evett 1995, 1998; Wilks 1998; Kennedy and Szpakowicz 2008).

It has been proposed in the literature that Roget-style thesauruses could provide an alternative source of lexical-semantic information, which can be used both to attack other kinds of NLP tasks than a wordnet, and even work better for some of the same tasks, e.g., *lexical cohesion*, *synonym identification*, *pseudo-word-sense disambiguation*, and *analogy problems* Morris and Hirst (1991); Jarmasz and Szpakowicz (2004); Kennedy and Szpakowicz (2008, 2014).

An obstacle to the wider use of Roget in NLP applications is its limited availability. The only free digital version is the 1911 American edition available through Project Gutenberg.[14] This version is obviously not well suited for processing modern texts. Szpakowicz and his colleagues at the University of Ottawa have conducted a number of experiments with a modern (from 1987) edition of Roget (e.g., Jarmasz and Szpakowicz 2004; Kennedy and Szpakowicz 2008), but as far as we can tell, this dataset is not generally available, due to copyright restrictions. The work reported by Kennedy and Szpakowicz (2014) represents an effort to remedy this situation, utilizing corpus-based measures of semantic relatedness for adding new entries to both the 1911 and 1987 editions of Roget.

In order to investigate systematically the strengths and weaknesses of diverse lexical-semantic resources when applied to different classes of NLP tasks, we would need access to resources that are otherwise comparable, e.g., with respect to language, vocabulary and domain coverage. The resources should also ideally be freely available, in order to ensure reproducibility as well as to stimulate their widest possible application to a broad range of NLP problems. Unfortunately, this situation is rarely encountered in practice; for English, the experiments contrasting WordNet and Roget have indicated that these resources are indeed complementary. It would be desirable to replicate these findings, e.g., for other languages and also using lexical-semantic resources with different structures (WordNet and Roget being two out of a large number of possibilities).

This is certainly a central motivation for the work presented here, the ultimate goal of which is to develop automatic methods for producing or considerably facilitating the production of a Swedish counterpart of Roget with a large and up-to-date vocabulary coverage. This is not to be done by translation, as in previous work by de Melo and Weikum

[14]See `http://www.gutenberg.org/ebooks/22` and Cassidy 2000.

(2008) and Borin, Allwood and de Melo (2014b). Instead, an existing but largely outdated Roget-style thesaurus will provide the scaffolding, where new word senses can be inserted with the help of two different kinds of semantic relatedness measures:

1. One such measure is corpus-based, similar to the experiments conducted by Kennedy and Szpakowicz (2014), described above.
2. The other measure utilizes an existing lexical-semantic resource (SALDO: Borin, Forsberg and Lönngren 2013).

In the latter case, we also have a more theoretical aim with our work. SALDO was originally conceived as an "associative thesaurus" (Lönngren 1998), and even though its organization in many respects differs significantly from that of Roget, there are also some commonalities. Hence, our hypothesis is that the structure of SALDO will yield a good semantic relatedness measure for the task at hand. SALDO is described in section 10.2.2 below.

10.2 The datasets

10.2.1 Bring's Swedish thesaurus

Sven Casper Bring (1842–1931) was the originator of the first and so far only adaptation of Roget's *Thesaurus* to Swedish, which appeared in 1930 under the title *Svenskt Ordförråd ordnat i begreppsklasser* 'Swedish vocabulary arranged in conceptual classes' (referred to as "Bring" or "Bring's thesaurus" below). The work itself consists of two parts: (1) a conceptually organized list of Roget categories; and (2) an alphabetically ordered lemma index.

In addition, there is a brief preface by S. C. Bring, which we reproduce here in its entirety:[15]

> This wordlist has been modelled on P. M. Roget's "Thesaurus of English Words and Phrases". This kind of wordlist can be seen as a synonym dictionary of sorts. But each conceptual class comprises not only synonyms, but words of all kinds which are habitually used in discoursing on the kind of topics which could be subsumed under the class label concept, understood in a wide sense.

[15]This English translation comes from the Bring resource page at Språkbanken: `http://spraakbanken.gu.se/eng/resource/bring`.

> Regarding Roget's classification system, there are arguably a number of classes which ought to be merged or split. But this classification seems to have established itself solidly through many editions of Roget's work as well as German copies of it. It should also be considered an advantage that the same classification is used in such dictionaries for different languages.
>
> Uppsala in September 1930.
>
> *S. C. Bring*

Like in Roget, the vocabulary included in Bring is divided into slightly over 1,000 "conceptual classes". A "conceptual class" corresponds to what is usually referred to as a "head" in the literature on Roget. Each conceptual class consists of a list of words (lemmas), subdivided first into nouns, verbs and others (mainly adjectives, adverbs and phrases), and finally into paragraphs. In the paragraphs, the distance – expressed as difference in list position – between words provides a rough measure of their semantic distance.

Bring thus forms a hierarchical structure with four levels:

(1) conceptual class (Roget "head")
(2) part of speech
(3) paragraph
(4) lemma (word sense)

This stands in contrast to Roget, where the formal structure defines a nine-level hierarchy (Jarmasz and Szpakowicz 2001, 2004):

(1) class
(2) section
(3) subsection
(4) category, or head group
(5) head (Bring "conceptual class")
(6) part of speech
(7) paragraph
(8) semicolon group
(9) lemma (word sense)

Since most of the Bring classes have corresponding heads in Roget, it should be straightforward to add the levels above Roget heads/Bring classes to Bring if needed. There are some indications in the literature that this additional structure can in fact be useful for calculating semantic similarity (Jarmasz and Szpakowicz 2004).

Bring's thesaurus has recently been made available in two digital versions by Språkbanken (the Swedish Language Bank) at the University of Gothenburg, both versions under a Creative Commons Attribution License:

Bring (v. 1): A digital version of the full contents of the original 1930 book version (148,846 entries).[16]

Blingbring (v. 0.1), a version of Bring where obsolete items have been removed and the remaining entries have been provided with word sense identifiers from SALDO (see section 10.2.2), providing links to most of Språkbanken's other lexical resources. This version contains 126,911 entries.[17]

The linking to SALDO senses in the current Blingbring version (v 0.1) has not involved a disambiguation step. Rather, it has been made by matching lemma-POS combinations from the two resources. For this reason, Blingbring includes slightly over 21,000 ambiguous entries (out of approximately 127,000 in total), or about 4,800 ambiguous word sense assignments (out of about 43,000 unique lemma-POS combinations).

The aim of the experiments described below has been to assess the feasibility of disambiguating these ambiguous linkages automatically, and specifically also to evaluate SALDO as a possible knowledge source for accomplishing this disambiguation. The longer-term goal of this work is to develop good methods for adding modern vocabulary automatically to Bring from, e.g., SALDO, thereby hopefully producing a modern Swedish Roget-style resource for the NLP community.

10.2.2 SALDO

SALDO (Borin, Forsberg and Lönngren 2013) is a large (137K entries and 2M wordforms) morphological and lexical-semantic lexicon for modern Swedish, freely available (under a Creative Commons Attribution license).[18]

As a lexical-semantic resource, SALDO is organized very differently from a wordnet (Borin and Forsberg 2009). As mentioned above, it was initially conceived as an "associative thesaurus". Since it has been extended following the principles laid down initially by Lönngren (1998), this characterization should still be valid, even though it has grown tremendously over the last decade.

If the fundamental organizing principle of PWN is the idea of full synonyms in a taxonomic concept hierarchy, the basic linguistic idea underlying SALDO is instead that, semantically speaking, the whole vocabulary of a language can be described as having a center – or core –

[16]`http://spraakbanken.gu.se/eng/resource/bring`
[17]`http://spraakbanken.gu.se/eng/resource/blingbring`
[18]`http://spraakbanken.gu.se/eng/resource/saldo`

and (consequently) a periphery. The notion of *core vocabulary* is familiar from several linguistic subdisciplines (Borin 2012). In SALDO this idea is consistently applied down to the level of individual word senses, as we will now describe.

The basic lexical-semantic organizational principle of SALDO is hierarchical. Every entry in SALDO – representing a word sense – is supplied with one or more semantic descriptors, which are themselves also entries in the dictionary. All entries in SALDO are actually occurring words or conventionalized or lexicalized multi-word units of the language. No attempt is made to fill perceived gaps in the lexical network using definition-like paraphrases, as is sometimes done in PWN (Fellbaum 1998b: 5f). A further difference as compared to PWN (and Roget-style thesuruses) is that SALDO aims to provide a lexical-semantic description of *all* the words of the language, including the closed-class items (prepositions, subjunctions, interjections, etc.), and also including many proper nouns.

One of the semantic descriptors in SALDO, called *primary*, is obligatory. The primary descriptor is the entry which better than any other entry fulfills two requirements: (1) it is a semantic neighbor of the entry to be desribed and (2) it is more central than it. However, there is no requirement that the primary descriptor is of the same part of speech as the entry itself. Thus, the primary descriptor of *kniv* 'knife (n)' is *skära* 'cut (v)', and that of *lager* 'layer (n)' is *på* 'on (p)'.

Through the primary descriptors SALDO is a single tree, rooted by assigning an artifical top sense (called PRIM) as primary descriptor to the 41 topmost word senses.

That two words are semantic neighbors means that there is a direct semantic relationship between them (such as synonymy, hyponymy, meronymy, argument-predicate relationship, etc.). As could be seen from the examples given above, SALDO includes not only open-class words, but also pronouns, prepositions, conjunctions etc. In such cases closeness must sometimes be determined with respect to function or syntagmatic connections, rather than ("word-semantic") content.

Centrality is determined by means of several criteria: frequency, stylistic value, word formation, and traditional lexical-semantic relations all combine to determine which of two semantically neighboring words is to be considered more central.

For more details of the organization of SALDO and the linguistic motivation underlying it, see Borin, Forsberg and Lönngren 2013.

Like Roget, SALDO has a kind of topical structure, which – again like Roget, but different from a wordnet – includes and connects lexical items of different parts of speech, but its topology is characterized by a much deeper hierarchy than that found in Roget. There are no direct correspondences in SALDO to the lexical-semantic relations making up a wordnet (minimally synonymy and – part-of-speech internal – hyponymy).

Given the (claimed) thesaural character of SALDO, we would expect a SALDO-based semantic similarity measure to work well for disambiguating the ambiguous Blingbring entries, and not be inferior to a corpus-based or wordnet-based measure. There is no sufficiently large Swedish wordnet at present, so for now we must restrict ourselves to a comparison of a corpus-based and a SALDO-based method.

The experiments described below were conducted using SALDO v. 2.3 as available for downloading on Språkbanken's website.

10.3 Automatic disambiguation of ambiguous Bring entries

We now turn to the question of automatically linking the Bring and SALDO lexicons: many entries in Bring have more than one sense in SALDO, and we present a number of methods to automatically rank SALDO senses by how well they fit into a particular Bring class. Specifically, since entries in Bring are not specified in terms of a sense, this allows us to predict the SALDO sense that is most appropriate for a given Bring entry. For instance, the lexicon lists the noun *broms* as belonging to Bring class 366, which contains a large number of terms related to animals. SALDO defines two senses for this word: *broms-1* 'brake' and *broms-2* 'horsefly', but it is only the second sense that should be listed in this Bring class.

In this work we consider the task of selecting a SALDO sense for a Bring entry, but we imagine that the methods proposed here can be applied in other scenarios as well. For instance, it is possible that they could allow us to predict the Bring class for a word that is *not* listed in Bring, but we leave this task for future investigation. The methods are related to those presented by Johansson (2014) for automatically suggesting FrameNet frames for SALDO entries.

We first describe how we use the SALDO network and cooccurrence statistics from corpora to represent the meaning of SALDO entries. These meaning representations are then used to carry out the disambiguation. We investigate two distinct ways to use the representations for disambiguating: (1) by selecting a *prototype* (centroid) for each class, and then selecting the SALDO sense that is most similar to the prototype; (2) by using the existing Bring entries as training instances for a *classifier* that assigns a Bring class to a SALDO entry, and then ranking the SALDO senses by the probability output by the classifier when considering each sense for a Bring class.

10.3.1 Representing the meaning of a SALDO entry

To be able to connect a SALDO entry to a Bring class, we must represent its *meaning* in some structured way, in order to relate it to other entries with a similar meaning. There are two broad approaches to representing word meaning in NLP work: representations based on the structure of a formal knowledge representation (in our case the SALDO network), and those derived from co-occurrence statistics in corpora (*distributional* representations). In this work, we explore both options.

10.3.1.1 Word senses in Bring and in SALDO

But even if we restrict ourselves to how they are conceived in the linguistic literature, word senses are finicky creatures. They are obviously language-dependent, strongly so if we are to believe, e.g., Goddard (2001). Furthermore, there seems to be a strong element of tradition – or ideology – informing assumptions about how word senses contribute to the interpretation of complex linguistic items, such as productive derivations, compounds and incorporating constructions, as well as phrases and clauses. This in turn determines the granularity – the degree of polysemy – posited for lexical entries.

One thing that seems to be assumed about Roget – and which if true consequently ought to hold for Bring as well – is that multiple occurrences of the same lemma (with the same part of speech) represent different word senses (e.g., Kwong 1998; Nastase and Szpakowicz 2001). This is consistent with a “splitting” approach to polysemy, similar to that exhibited by PWN and more generally by an Anglo-Saxon lexicographical tradition.

However, this is not borne out by the Bring–SALDO linking. First, there are many unambiguous – in the sense of having been assigned only one SALDO word sense – Bring lemma-POS combinations that appear in multiple Bring classes. Second, during the practical disambiguation work conducted in order to prepare the evaluation dataset for the experiments described below, the typical case was not – as would have been expected if the above assumption were correct – that ambiguous items occurring in several Bring classes would receive different word sense assignments. On the contrary, this turned out to be very much a minor phenomenon.

A "word sense" is not a well-defined notion (Kilgarriff 1997; Hanks 2000; Erk 2010; Hanks 2013), and it may well be simply that this is what we are seeing here. Specifically, the Swedish lexicographical tradition to which SALDO belongs reflects a "lumping" view on word sense discrimination. If we aspire to link resources such as Roget, Bring, SALDO, etc. between languages, issues such as this need to be resolved one way or another, so there is clearly need for more research here.

10.3.1.2 Lexicon-based representation

In a structure-based meaning representation, the meaning of a concept is defined by its relative position in the SALDO network. How do we operationalize this position as a practical meaning representation that can be used to compute similarity of meaning or exemplify meaning for a machine learning algorithm? It seems clear that the way this operationalization is carried out has implications for the ability of automatic systems to generalize from the set of SALDO entries associated with a Bring class, in order to reason about new entries.

When using a semantic network, the meaning of a word sense s is defined by how it is related to other word senses; in SALDO, the immediate neighborhood of s consists of a primary descriptor and possibly a set of secondary descriptors, and the meaning of s can be further analyzed by following primary and secondary edges in the SALDO graph. In this work, we follow the approach by Johansson (2014) and let the lexicon-based meaning representation $\phi(s)$ of a SALDO entry s be defined in terms of the transitive closure of the primary descriptor relation. That is, it consists of all SALDO entries observed when traversing the SALDO graph by following primary descriptor edges from s to the SALDO root entry (excluding the root itself). For instance, the meaning of the fourth sense of *fil* 'file (n)' would be represented as the set

ϕ(*fil-4*) = { *fil-4* '(computer) file (n)', *datorminne-1* 'computer memory (n)', *datalagring-1* 'data storage (n)', *lagring-1* 'storage (n)', *lagra-1* 'store (v)', *lager-2* 'stock/store (n)', *förråd-1* 'store (n)', *förvara-1* 'store/keep (v)', *ha-1* 'have (v)' }.

Computationally, these sets are implemented as high-dimensional sparse vectors, which we normalize to unit length. Although in this work we do not explicitly use the notion of similarity functions, we note that the cosine similarity applied to this representation gives rise to a network-based measure similar in spirit to that proposed by Wu and Palmer (1994b):

$$\mathrm{sim}(s_1, s_2) = \frac{|\phi(s_1) \cap \phi(s_2)|}{\sqrt{|\phi(s_1)|} \cdot \sqrt{|\phi(s_2)|}}$$

10.3.1.3 Corpus-based representation

Corpus-based meaning representations rely on the distributional hypothesis, which assumes that words occurring in a similar set of contexts are also similar in meaning (Harris 1954). This intuition has been realized in a very large number of algorithms and implementations (Turney and Pantel 2010), and the result of applying such a model is typically that word meaning is modeled *geometrically* by representing co-occurrence statistics in a vector space: this makes it straightforward to define similarity and distance measures using standard vector-space metrics, e.g. the Euclidean distance or the cosine similarity. In this work, we applied the skip-gram model by Mikolov et al. (2013a), which considers co-occurrences of each word in the corpus with other words in a small window; this model has proven competitive in many evaluations, including the frame prediction task described by Johansson (2014).

Since our goal is to select a word sense defined by SALDO, but corpus-based meaning representation methods typically do not distinguish between senses, we applied the postprocessing algorithm developed by Johansson and Nieto Piña (2015) to convert vectors produced by the skip-gram model into new vectors representing SALDO senses. For instance, this allows us to say that for the Swedish noun *fil*, the third sense defined in SALDO (‘sour milk’) is geometrically close to *milk* and *yoghurt* while the fourth sense (‘computer file’) is close to *program* and *memory.* This algorithm decomposes vector-based word meaning representations into a convex combination of several components, each representing a sense defined by a semantic network such as SALDO. The vector representations of senses are selected so that they minimize the geometric distances to their neighbors in the SALDO graph. The authors showed that the decomposed representations can be used for predicting FrameNet frames for a SALDO sense.

10.3.2 Disambiguating by comparing to a prototype

The fact that corpus-based representations for SALDO senses are located in a real-valued vector space allows us to generate a prototype for a certain Bring conceptual class by means of averaging the sense vectors belonging to a that class in Bring. This prototype is in the same vector space as the sense representations, so we are able to measure distances between sense vectors and prototypes and determine which sense is closer to the concept embodied in the class prototype.

Thus, our first method for disambiguating links between Bring items and SALDO senses works as follows. For each class j, a prototype c_j is calculated by averaging those sense vectors v_i that are unambiguously linked to a Bring item b_i from class j:

$$c_j = \frac{1}{n} \sum_{b_i \in j} v_i$$

where n is the number of unambiguous links in class j.

Then, for an ambiguous link between a Bring item b_k in class j and its set of possible vectors $\{v_{kl}\}$, the distance from each vector to the class centroid c_j is measured, and the closest one is selected as the representation of the SALDO sense linked to b_k:

$$\arg\min_{l} \mathrm{d}(c_j, v_{kl})$$

where d is a distance function. In our case we have chosen to use *cosine distance*, which is commonly applied on the kind of representations obtained from the Skip-gram model (Mikolov et al. 2013a) to compute similarity between representations.

10.3.3 Disambiguating with classifiers

Statistical classifiers offer a wide range of options to learn the distribution of labeled data, which afterwards can be used to label unseen data instances. They are not constrained to work with data in a geometric space, as opposed to the method explained in the previous section. Thus, we can apply classifiers on lexicon-based representations as well.

In our case, we are not interested so much in classifying new instances as in assessing the confidence of such classifications. Consequently, in our ambiguous data we have a set of instances that can possibly be linked to a Bring entry whose class is known to us. Therefore, we would like to ascertain how confident a classifier is when assigning these instances to their corresponding class, and base our decision to disambiguate the link on this information.

For this task we use the Python library Scikit-learn (Pedregosa et al. 2011), a general machine learning package which offers a variety of statistical classifiers. Specifically, we work with a logistic regression method (instantiated with the library's default values, except the inverse regularization strength, set to 100), which classifies instances based on the probability that they belong to each possible class.

The classifier is trained on the set of SALDO sense vectors unambiguously linked to Bring items and their conceptual class information. Once trained, it can be given a set of SALDO sense representations $\{v_{kl}\}$ ambiguously assigned to one Bring entry b_k in class j and, instead of simply classifying them, output their probabilities $\{p_{jl}\}$ of belonging to class j. We then only have to select the sense with the highest probability to disambiguate the link:

$$\arg\max_{l} p_{jl}$$

10.4 Experiments

10.4.1 Evaluation data preparation

The Blingbring data was downloaded from Språkbanken's website and a sample of ambiguous Bring–SALDO linkages was selected for manual disambiguation.

An initial sample was drawn from this data set according to the following principles:[19]

- The sampling unit was the class+part of speech-combination, i.e., *nouns in class 12*, *verbs in class 784*, etc.
- This unit had to contain at least 100 lemmas (actual range: 100–569 lemmas),
- out of which at least 1 must be unambiguous (actual range: 56–478 unambiguous lemmas),
- and at least 4 had to be ambiguous.
- From the ambiguous lemmas, 4 were randomly selected (using the Python function random-sample).

[19]These should be seen as first-approximation heuristic principles, and not based on any more detailed analysis of the data. We expect that further experiments will provide better data on which to base such decisions.

The goal was to produce an evaluation set of approximately 1,000 items, and this procedure yielded 1,008 entries to be disambiguated. The disambiguation was carried out by the first author. In practice, it deviated from the initial procedure and proceeded more opportunistically, since reference often had to be made to the main dataset in order to determine the correct SALDO word sense. On these occasions, it was often convenient to (a) either disambiguate additional items in the same Bring class; and/or (b) disambiguate the same items throughout the entire dataset.

In the end, 1,368 entries were disambiguated for the experiments, out of which about 500 came out of the original sample. The degree of ambiguity in this gold standard data is shown in the second column of table 10.1, while the third column shows the degree of ambiguity in the full Blingbring dataset containing 44,615 unique lemma-POS combinations.

# senses/ entry	**GS data: # entries**	**Blingbring: # entries**
1	–	39,275
2	888	4,006
3	266	873
4	122	286
5	56	102
6	18	31
7	10	18
8	7	10
9	1	3
10	–	6
11	–	5

Table 10.1: Word-sense ambiguity in the gold standard data and in Blingbring

On the other hand, unambiguous entries in Blingbring linking one Bring item to one SALDO sense are isolated to serve as training data. As mentioned above in section 10.3.1.1, the structure of Bring's thesaurus makes it possible for a word to appear in more than one conceptual class; if the SALDO sense related to those two or more instances is the same, we may have a training instance that spans more than just one class. Initially, it may seem reasonable to exclude such instances from the training data, as their presence may be problematic for the definition of a class. But this phenomenon is quite ubiquitous: 72.6% of the senses unambiguously associated with a Bring entry in Blingbring appear in

Method	**Accuracy**
Random baseline	0.4238
Corpus-based, incl. overlap	0.5731
Corpus-based, no overlap	0.5651

Table 10.2: Disambiguation accuracy using a similarity measure.

more than one class. For this reason, we define two different training sets, one that includes *overlap* among the classes and one that does not, and conduct conduct experiments separately on each of them.

10.4.2 Prototype-based disambiguation

In this section we give the results obtained with the method described in section 10.3.2. This experiment is performed using corpus-based representations only, as lexicon-based ones lack a geometrical interpretation, on which the cosine similarity measure used is based.

Table 10.2 lists the accuracy of the method on our evaluation set. Two results are given corresponding to the training set containing or not instances that span several classes. The accuracy of a random baseline is also given as a reference. Both of the approaches have an accuracy well above the random baseline with an improvement of over 0.14 points, and we observe that there is practically no difference between them, although the approach in which instances overlapping classes are included in the training data performs slightly better.

In table 10.3 we present for this last case a breakdown of the accuracy into the parts of speech that Bring classes list: *nouns*, *verbs* and *others*.[20] The table also lists the proportions of these classes in the data. No significant difference can be appreciated between the diverse types of words, although nouns fare slightly better than the other two cases.

10.4.3 Classification-based disambiguation

The results of applying the method introduced in section 10.3.3 are given here. In this experiment we also consider lexicon-based data besides the corpus-based representations.

[20]As explained in section 10.2.1, the tag *others* encompasses mainly adjectives, adverbs and phrases, and unfortunately there is not enough information in Bring to separate these classes and give a more fine-grained analysis.

PoS	**Proportion**	**Accuracy**
Noun	54.8%	0.5819
Verb	21.3%	0.5538
Others	23.2%	0.5485

Table 10.3: Disambiguation accuracy by Part-of-Speech using a similarity measure. Overlapping instances included in the training set.

Table 10.4 lists the accuracies obtained in each instance: corpus-based or lexicon-based data, using either overlapping instances or not. The random baseline accuracy is also shown for reference.

In this case, we observe a greater improvement over the baseline than in the previous experiment with an increase in accuracy of 0.23 between the best cases in each experiment. There is also a considerable difference between the two types of data: the best case using lexicon-based representations provides an accuracy improvement of 0.12 over the best result obtained with corpus-based data. Contrary to the experience of the previous experiment, there is a substantial difference between the presence or absence of overlapping instances in the training data: the accuracy increases by 0.03 in the case of corpus-based data when overlapping instances are used, and by 0.13 in the case of lexicon-based data. This behaviour may seem counter-intuitive, since using training instances that belong to more than one class should dilute the boundaries between those classes. It should be noted here, however, that, given a new instance, the main task assigned in our problem to the classifier is not to decide to which class the instance belongs (as this information is already known), but to output the membership probability for a certain class, so that we are able to compare with those of other instances. Thus, the boundaries between classes matter less to us than the amount of training data that allows the classifier to learn the definition of each class separately.

Table 10.5 presents an accuracy breakdown for the highest scoring approach in the previous results (i.e., including overlap) using each type of data. These results also differ from the ones in the previous experiments, as we observe a marked difference between parts of speech: using corpus-based representations, nouns obtain the highest accuracy with 0.10 points over the other two classes, while using lexicon based data favours verbs, although closely followed by nouns.

Method	**Accuracy**
Random baseline	0.4238
Corpus-based, incl. overlap	0.6879
Corpus-based, no overlap	0.6572
Lexicon-based, incl. overlap	0.7836
Lexicon-based, no overlap	0.6499

Table 10.4: Disambiguation accuracy using a classifier.

Corpus-based

PoS	**Accuracy**
Noun	0.7372
Verb	0.6308
Others	0.5825

Lexicon-based

PoS	**Accuracy**
Noun	0.7885
Verb	0.8154
Others	0.7282

Table 10.5: Disambiguation accuracy by Part-of-Speech using a classifier. Overlapping instances included in the training data.

10.5 Conclusions and future work

Summing up the main results, (1) both the corpus-based and the lexicon-based methods resulted in a significantly higher disambiguation accuracy compared to the random baseline; (2) contrary to intuition, using overlapping instances yielded better accuracy than using only non-overlapping items, which we attribute to the increased amount of training data in the former case; and (3) the hypothesis that the SALDO-based method would yield a better result was supported by the experiments.

The results of the lexicon-based method are already good enough overall that it will be possible to use it as a preprocessing step in order to speed up the disambiguation of the remaining ambiguous entries considerably. The results could also be analyzed in more detail in order to find out whether there are special cases that could be automatically identified where the accuracy may be even higher.

For instance, it would be useful to see whether the structure of the thesaurus can be used in a more sophisticated way. In this work we have only considered the top-level Bring class when selecting among the alternative SALDO senses for an ambiguous Bring entry, but as described in section 10.2.1, the thesaurus is organized hierarchically, and closely related terms are placed near each other on the page.

In future work, we would like to investigate to what extent the methods that we have proposed here can be generalized to other Bring-related tasks. In particular, it would be useful to propose a Bring class for words in SALDO that are not listed in Bring, for instance because the word did not exist when the Bring lexicon was compiled. This would make a new and very useful lexical-semantic resource available for use in sophisticated Swedish NLP applications.

Acknowledgements

This research was funded by the Swedish Research Council under the grants *Towards a knowledge-based culturomics* (dnr 2012-5738), *Distributional methods to represent the meaning of frames and constructions* (dnr 2013-4944), and *Swedish FrameNet++* (dnr 2010-6013). We also acknowledge the University of Gothenburg for its support of the Centre for Language Technology and Språkbanken.

References

Abadi, Martín, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard et al. 2016. Tensorflow: A system for large-scale machine learning. *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, 265–283.

Agirre, Eneko and Aitor Soroa 2009. Personalizing pagerank for word sense disambiguation. *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, 33–41. Association for Computational Linguistics.

Allén, Sture 1981. The lemma-lexeme model of the swedish lexical database. *Empirical Semantics. Bochum*, pp. 376–387.

Alsuhaibani, Mohammed, Danushka Bollegala, Takanori Maehara and Ken-ichi Kawarabayashi 2018. Jointly learning word embeddings using a corpus and a knowledge base. *PLOS ONE* 13 (3): 1–26 (03).

Amrami, Asaf and Yoav Goldberg 2018. Word sense induction with neural biLM and symmetric patterns. *Proceedings of the 2018 conference on empirical methods in natural language processing*, 4860–4867. Association for Computational Linguistics.

Arora, Sanjeev, Yingyu Liang and Tengyu Ma 2017. A simple but tough-to-beat baseline for sentence embeddings. *Proceedings of the International Conference on Learning Representations (ICLR).*

Artstein, Ron and Massimo Poesio 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics* 34 (4): 555–596.

Athiwaratkun, Ben, Andrew Gordon Wilson and Anima Anandkumar 2018. Probabilistic fasttext for multi-sense word embeddings. *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers)*, pp. 1–11.

Atkins, Beryl TS 1992. Tools for computer-aided corpus lexicography: the hector project. *Acta Linguistica Hungarica* 41 (1/4): 5–71.

Bagga, Amit and Breck Baldwin 1998. Algorithms for scoring coreference chains. *Proceedings of the 1st International Conference on Language Resources and Evaluation*, 563–566. Granada, Spain.

Baker, Collin F, Charles J Fillmore and John B Lowe 1998. The berkeley framenet project. *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, 86–90. Association for Computational Linguistics.

Bansal, Mohit, Kevin Gimpel and Karen Livescu 2014. Tailoring continuous word representations for dependency parsing. *Proceedings of the 52nd annual meeting of the association for computational linguistics (volume 2: Short papers)*, Volume 2, 809–815.

Baroni, Marco, Georgiana Dinu and Germán Kruszewski 2014. Don't count, predict! A systematic comparison of context-counting vs. context-predicting semantic vectors. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Volume 1.

Baroni, Marco, Brian Murphy, Eduard Barbu and Massimo Poesio 2010. Strudel: A corpus-based semantic model based on properties and types. *Cognitive science* 34 (2): 222–254.

Bartunov, Sergey, Dmitry Kondrashkin, Anton Osokin and Dmitry Vetrov 2016. Breaking sticks and ambiguities with adaptive skip-gram. *Artificial Intelligence and Statistics*, 130–138.

Bengio, Yoshua, Réjean Ducharme, Pascal Vincent and Christian Jauvin 2003. A neural probabilistic language model. *Journal of machine learning research* 3 (Feb): 1137–1155.

Bojanowski, Piotr, Edouard Grave, Armand Joulin and Tomas Mikolov 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics* 5: 135–146.

Bordes, Antoine, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston and Oksana Yakhnenko 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 2787–2795.

Borin, Lars 2012. Core vocabulary: A useful but mystical concept in some kinds of linguistics. Diana Santos, Krister Lindén and Wanjiku Ng'ang'a (eds), *Shall we play the Festschrift game? Essays on the occasion of Lauri Carlson's 60th birthday*, 53–65. Berlin: Springer.

Borin, Lars, Jens Allwood and Gerard de Melo 2014a. Bring vs. mtroget: Evaluating automatic thesaurus translation. *Proceedings of lrec 2014, may 26-31, 2014 reykjavik, iceland.* European Language Resources Association.

Borin, Lars, Jens Allwood and Gerard de Melo 2014b. Bring vs. MTRoget: Evaluating automatic thesaurus translation. *Proceedings of LREC 2014*, 2115–2121. Reykjavík: ELRA.

Borin, Lars and Markus Forsberg 2009. All in the family: A comparison of SALDO and WordNet. *Proceedings of the nodalida 2009 workshop on wordnets and other lexical semantic resources.* Odense.

Borin, Lars, Markus Forsberg, Martin Hammarstedt, Dan Rosén, Roland Schäfer and Anne Schumacher 2016. Sparv: Språkbanken's corpus annotation pipeline infrastructure. *The sixth Swedish Language Technology Conference (SLTC), Umeå University*, 17–18.

Borin, Lars, Markus Forsberg and Lennart Lönngren 2013. SALDO: a touch of yin to WordNet's yang. *Language Resources and Evaluation* 47: 1191–1211.

Borin, Lars, Markus Forsberg and Johan Roxendal 2012. Korp-the corpus infrastructure of Språkbanken. *LREC*, 474–478.

Borin, Lars, Luis Nieto Piña and Richard Johansson 2015. Here be dragons? the perils and promises of inter-resource lexical-semantic mapping. *Proceedings of the workshop on Semantic resources and semantic annotation for Natural Language Processing and the Digital Humanities at NODALIDA 2015*, 1–11. Vilnius, Lithuania: Northern European Association for Language Technology.

Brin, Sergey and Lawrence Page 1998. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems* 30 (1-7): 107–117.

Brody, Samuel and Mirella Lapata 2009. Bayesian word sense induction. *Proceedings of the 12th conference of the european chapter of the association for computational linguistics*, 103–111. Association for Computational Linguistics.

Camacho-Collados, José and Roberto Navigli 2016. Find the word that does not belong: A framework for an intrinsic evaluation of word vector representations. *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*, 43–50.

Camacho-Collados, José, Mohammad Taher Pilehvar, Nigel Collier and Roberto Navigli 2017. Semeval-2017 task 2: Multilingual and cross-lingual semantic word similarity. *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, 15–26. Association for Computational Linguistics, Vancouver, Canada.

Cassidy, Patrick 2000. An investigation of the semantic relations in the Roget's Thesaurus: Preliminary results. *Proceedings of CICLing 2000*, 181–204.

Cer, Daniel, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio and Lucia Specia 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *SemEval-2017 - Proceedings of the 11th International Workshop on Semantic Evaluations*, pp. 1–14.

Chen, Xinxiong, Zhiyuan Liu and Maosong Sun 2014. A unified model for word sense representation and disambiguation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1025–1035.

Cisco 2017. The zettabyte era–trends and analysis. *Cisco white paper.*

Cohen, Jacob 1960. A coefficient of agreement for nominal scales. *Educational and psychological measurement* 20 (1): 37–46.

Collobert, Ronan and Jason Weston 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. *Proceedings of the 25th International Conference on Machine Learning*, 160–167. ACM.

Collobert, Ronan, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu and Pavel Kuksa 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research* 12: 2493–2537.

Cortes, Corinna and Vladimir Vapnik 1995. Support-vector networks. *Machine learning* 20 (3): 273–297.

Cruse, D. Alan 1986. *Lexical semantics.* Cambridge: Cambridge University Press.

Cuba Gyllensten, Amaru and Magnus Sahlgren 2015. Navigating the semantic horizon using relative neighborhood graphs. *Proceedings of the 2015 conference on empirical methods in natural language processing*, 2451–2460.

Deerwester, Scott, Susan T Dumais, George W Furnas, Thomas K Landauer and Richard Harshman 1990. Indexing by latent semantic analysis. *Journal of the American society for information science* 41 (6): 391–407.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee and Kristina Toutanova 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, vol. abs/1810.04805.

Edmonds, Philip and Scott Cotton 2001. Senseval-2: overview. *The Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems*, 1–5. Association for Computational Linguistics.

Erk, Katrin 2010. What is word meaning, really? (And how can distributional models help us describe it?). *Proceedings of the 2010* *workshop on geometrical models of natural language semantics*, 17–26. Uppsala: ACL.

Fan, Rong-En, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang and Chih-Jen Lin 2008. LIBLINEAR: A library for large linear classification. *Journal of machine learning research* 9 (Aug): 1871–1874.

Fang, Wei, Jianwen Zhang, Dilin Wang, Zheng Chen and Ming Li 2016. Entity disambiguation by knowledge and text jointly embedding. *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, 260–269.

Faruqui, Manaal, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy and Noah A Smith 2015. Retrofitting word vectors to semantic lexicons. *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1606–1615.

Fellbaum, Christiane (ed.) 1998a. *WordNet: An electronic lexical database.* Cambridge, Mass.: MIT Press.

Fellbaum, Christiane 1998b. Introduction. Christiane Fellbaum (ed.), *WordNet: An electronic lexical database*, 1–19. Cambridge, Mass.: MIT Press.

Fillmore, Charles J. and Collin Baker 2009. A frames approach to semantic analysis. B. Heine and H. Narrog (eds), *The oxford handbook of linguistic analysis*, 313–340. Oxford: OUP.

Finkelstein, Lev, Evgeniy Gabrilovich, Yossi Matias, Ehud Rivlin, Zach Solan, Gadi Wolfman and Eytan Ruppin 2002. Placing search in context: The concept revisited. *ACM Transactions on information systems* 20 (1): 116–131.

Firth, John R 1957. A synopsis of linguistic theory, 1930-1955. *Studies in linguistic analysis.*

Friberg Heppin, Karin and Maria Gronostaj Toporowska 2012. The rocky road towards a swedish framenet-creating swefn. *LREC*, 256–261.

Gellerstam, Martin 1999. Lexin-lexikon för invandrare. *rapport nr.: Lexiconordica 6.*

Gers, Felix A and E Schmidhuber 2001. Lstm recurrent networks learn simple context-free and context-sensitive languages. *IEEE Transactions on Neural Networks* 12 (6): 1333–1340.

Ghanimifard, Mehdi and Richard Johansson 2015. Enriching word sense embeddings with translational context. *Proceedings of the international conference recent advances in natural language processing*, 208–215.

Gladkova, Anna, Aleksandr Drozd and Satoshi Matsuoka 2016. Analogy-based detection of morphological and semantic relations with word embeddings: what works and what doesn't. *Proceedings of the NAACL Student Research Workshop*, 8–15.

Glorot, Xavier, Antoine Bordes and Yoshua Bengio 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 513–520.

Goddard, Cliff 2001. Lexico-semantic universals: A critical overview. *Linguistic Typology* 5: 1–65.

Goikoetxea, Josu, Aitor Soroa and Eneko Agirre 2015. Random walks and neural network language models on knowledge bases. *Proceedings of the 2015 conference of the north american chapter of the association for computational linguistics: Human language technologies*, 1434–1439.

Goldberg, Yoav and Omer Levy 2014. word2vec explained: deriving Mikolov et al.'s negative-sampling word-embedding method. *arXiv preprint arXiv:1402.3722.*

Gong, Yihong and Xin Liu 2001. Generic text summarization using relevance measure and latent semantic analysis. *Proceedings of the 24th annual international acm sigir conference on research and development in information retrieval*, 19–25. ACM.

Gutmann, Michael and Aapo Hyvärinen 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 297–304.

Guðmundsdóttir, Björk, Sigurjón Birgir Sigurðsson and Lars von Trier 2000. I've seen it all. *Selmasongs: music from the motion picture soundtrack Dancer in the dark.*

Halácsy, Péter, András Kornai and Csaba Oravecz 2007. HunPos: an open source trigram tagger. *Proceedings of the 45th annual meeting of the acl on interactive poster and demonstration sessions*, 209–212. Association for Computational Linguistics.

Hamilton, William L., Jure Leskovec and Dan Jurafsky 2016. Diachronic word embeddings reveal statistical laws of semantic change. *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: Long papers)*, 1489–1501. Association for Computational Linguistics.

Hanks, Patrick 2000. Do word meanings exist? *Computers and the Humanities* 34 (1–2): 205–215.

Hanks, Patrick 2013. *Lexical analysis. Norms and exploitations.* Cambridge, Massachusetts: MIT Press.

Harris, Zellig S. 1954. Distributional structure. *Word* 10 (2-3): 146–162.

Hazman, Maryam, Samhaa R El-Beltagy and Ahmed Rafea 2011. A survey of ontology learning approaches. *International Journal of Computer Applications* 22 (9): 36–43.

Hilbert, Martin and Priscila López 2011. The world's technological capacity to store, communicate, and compute information. *Science*, p. 1200970.

Hill, Felix, KyungHyun Cho, Anna Korhonen and Yoshua Bengio 2016. Learning to understand phrases by embedding the dictionary. *Transactions of the Association for Computational Linguistics* 4: 17–30.

Hill, Felix, Roi Reichart and Anna Korhonen 2015. Simlex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics* 41 (4): 665–695.

Hinton, Geoffrey E, James L McClelland, David E Rumelhart et al. 1984. *Distributed representations.* Carnegie-Mellon University Pittsburgh, PA.

Huang, Eric H, Richard Socher, Christopher D Manning and Andrew Y Ng 2012. Improving word representations via global context and multiple word prototypes. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, 873–882. Association for Computational Linguistics.

Hüllen, Werner 2004. *A history of Roget's Thesaurus: Origins, development, and design.* Oxford: Oxford University Press.

Iacobacci, Ignacio, Mohammad Taher Pilehvar and Roberto Navigli 2015. SENSEMBED: Learning sense embeddings forword and relational similarity. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL-IJCNLP 2015*, Volume 1, 95–105. Association for Computational Linguistics (ACL).

Jarmasz, Mario and Stan Szpakowicz 2001. The design and implementation of an electronic lexical knowledge base. *Proceedings the 14th biennial conference of the Canadian society for computational studies of intelligence (AI 2001)*, 325–333.

Jarmasz, Mario and Stan Szpakowicz 2004. *Roget's Thesaurus* and semantic similarity. Nicolas Nicolov, Kalina Bontcheva, Galia Angelova and Ruslan Mitkov (eds), *Recent advances in natural language processing III. Selected papers from RANLP 2003*, 111–120. Amsterdam: John Benjamins.

Jauhar, Sujay Kumar, Chris Dyer and Eduard Hovy 2015. Ontologically grounded multi-sense representation learning for semantic vector space models. *Proceedings of the 2015 conference of the north american chapter of the association for computational linguistics: Human language technologies*, 683–693.

Jobbins, Amanda C. and Lindsay J. Evett 1995. Automatic identification of cohesion in texts: Exploiting the lexical organization of Roget's Thesaurus. *Proceedings of Rocling VIII*, 111–125. Taipei.

Jobbins, Amanda C. and Lindsay J. Evett 1998. Text segmentation using reiteration and collocation. *Proceedings of the 36th ACL and 17th COLING, Volume 1*, 614–618. Montreal: ACL.

Johansson, Richard 2014. Automatic expansion of the Swedish FrameNet lexicon. *Constructions and Frames* 6 (1): 92–113.

Johansson, Richard, Yvonne Adesam, Gerlof Bouma and Karin Hedberg 2016. A multi-domain corpus of Swedish word sense annotation. *Proceedings of the language resources and evaluation conference (lrec)*, 3019–3022. Portorož, Slovenia.

Johansson, Richard and Luis Nieto Piña 2015. Combining relational and distributional knowledge for word sense disambiguation. *Proceedings of the 20th Nordic Conference of Computational Linguistics*, 69–78. Vilnius, Lithuania.

Johansson, Richard and Luis Nieto Piña 2015. Embedding a semantic network in a word space. *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1428–1433. Denver, Colorado: Association for Computational Linguistics.

Johnson, Mark 2009. How the statistical revolution changes (computational) linguistics. *Proceedings of the eacl 2009 workshop on the interaction between linguistics and computational linguistics: Virtuous, vicious or vacuous?*, 3–11. Association for Computational Linguistics.

Jolliffe, IT 1986. Principal component analysis. *Springer Series in Statistics, Berlin: Springer, 1986.*

Jurgens, David and Ioannis Klapaftis 2013. SemEval-2013 task 13: Word sense induction for graded and non-graded senses. *Second Joint Con-*

*ference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, 290–299. Atlanta, United States.

Kågebäck, Mikael, Fredrik Johansson, Richard Johansson and Devdatt Dubhashi 2015. Neural context embeddings for automatic discovery of word senses. *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, 25–32.

Kennedy, Alistair and Stan Szpakowicz 2008. Evaluating *Roget's* thesauri. *Proceedings of acl-08: Hlt*, 416–424. Columbus, Ohio: ACL.

Kennedy, Alistair and Stan Szpakowicz 2014. Evaluation of automatic updates of *Roget's Thesaurus*. *Journal of Language Modelling* 2 (2): 1–49.

Kenter, Tom, Alexey Borisov and Maarten de Rijke 2016. Siamese cbow: Optimizing word embeddings for sentence representations. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Volume 1, 941–951.

Kiela, Douwe, Felix Hill and Stephen Clark 2015. Specializing word embeddings for similarity or relatedness. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2044–2048.

Kilgarriff, Adam 1997. I don't believe in word senses. *Computers and the Humanities* 31 (2): 91–113.

Kilgarriff, Adam and Joseph Rosenzweig 2000. Framework and results for english senseval. *Computers and the Humanities* 34 (1-2): 15–48.

Koehn, Phillip 2005. Europarl: A parallel corpus for statistical machine translation. *MT summit*, 79–86.

Kokkinakis, Dimitrios, Jerker Järborg and Yvonne Cederholm 2001. Senseval-2: the Swedish framework. *The Proceedings of the Second International Workshop on Evaluating Word Sense Disambiguation Systems*, 45–48. Association for Computational Linguistics.

Kwong, Oi Yee 1998. Aligning WordNet with additional lexical resources. *Workshop on usage of WordNet in natural language processing systems at COLING-ACL'98*, 73–79. Montréal: ACL.

Landauer, Thomas K and Susan T Dumais 1997. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review* 104 (2): 211.

Lau, Jey Han, Paul Cook, Diana McCarthy, David Newman and Timothy Baldwin 2012. Word sense induction for novel sense detection.

Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics, 591–601. Association for Computational Linguistics.

Lazaridou, Angeliki, Marco Baroni et al. 2015. Combining language and vision with a multimodal skip-gram model. *Proceedings of the 2015 conference of the north american chapter of the association for computational linguistics: Human language technologies*, 153–163.

LeCun, Yann, Yoshua Bengio and Geoffrey Hinton 2015. Deep learning. *nature* 521 (7553): 436.

Lenci, Alessandro 2008. Distributional semantics in linguistic and cognitive research. *Italian Journal of Linguistics* 20 (1): 1–31.

Levy, Omer and Yoav Goldberg 2014a. Neural word embedding as implicit matrix factorization. *Advances in neural information processing systems*, 2177–2185.

Levy, Omer and Yoav Goldberg 2014b. Dependency-based word embeddings. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Volume 2, 302–308. Baltimore, Maryland: Association for Computational Linguistics.

Li, Jiwei and Dan Jurafsky 2015. Do multi-sense embeddings improve natural language understanding? *Empirical Methods in Natural Language Processing (EMNLP)*, 1722–1732.

Lund, Kevin and Curt Burgess 1996. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior research methods, instruments, & computers* 28 (2): 203–208.

Lönngren, Lennart 1998. A Swedish associative thesaurus. *Euralex '98 proceedings, Vol. 2*, 467–474.

Maas, Andrew L, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng and Christopher Potts 2011. Learning word vectors for sentiment analysis. *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies-volume 1*, 142–150. Association for Computational Linguistics.

Manandhar, Suresh, Ioannis Klapaftis, Dmitriy Dligach and Sameer Pradhan 2010. Semeval-2010 task 14: Word sense induction & disambiguation. *Proceedings of the 5th International Workshop on Semantic Evaluation*, 63–68. Uppsala, Sweden.

Markoff, John 2005. Behind artificial intelligence, a squadron of bright real people. *The New York Times*, October.

Melamud, Oren, David McClosky, Siddharth Patwardhan and Mohit

Bansal 2016. The role of context types and dimensionality in learning word embeddings. *Proceedings of naacl-hlt*, 1030–1040.

de Melo, Gerard and Gerhard Weikum 2008. Mapping Roget's Thesaurus and WordNet to French. *Proceedings of LREC 2008*. Marrakech: ELRA.

de Melo, Gerard and Gerhard Weikum 2009. Towards a universal wordnet by learning from combined evidence. *Proceedings of the 18th ACM conference on information and knowledge management (CIKM 2009)*, 513–522. New York: ACM.

Mikolov, Tomas, Kai Chen, Greg Corrado and Jeffrey Dean 2013a. Efficient estimation of word representations in vector space. *Proceedings of International Conference on Learning Representations (ICLR)*, vol. abs/1301.3781.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado and Jeff Dean 2013b. Distributed representations of words and phrases and their compositionality. *Advances in Neural Information Processing Systems*, 3111–3119.

Miller, George 1998. *Wordnet: An electronic lexical database.* MIT press.

Miller, George A 1995. WordNet: a lexical database for English. *Communications of the ACM* 38 (11): 39–41.

Miller, George A, Martin Chodorow, Shari Landes, Claudia Leacock and Robert G Thomas 1994. Using a semantic concordance for sense identification. *Proceedings of the workshop on Human Language Technology*, 240–243. Association for Computational Linguistics.

Mitra, Sunny, Ritwik Mitra, Martin Riedl, Chris Biemann, Animesh Mukherjee and Pawan Goyal 2014. That's sick dude!: Automatic identification of word sense change across different timescales. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pp. 1020–1029.

Mnih, A and YW Teh 2012. A fast and simple algorithm for training neural probabilistic language models. *Proceedings of the 29th International Conference on Machine Learning, ICML 2012*, Volume 2, 1751–1758.

Mnih, Andriy and Koray Kavukcuoglu 2013. Learning word embeddings efficiently with noise-contrastive estimation. *Advances in Neural Information Processing Systems*, 2265–2273.

Morin, Frederic and Yoshua Bengio 2005. Hierarchical probabilistic neural network language model. *Proceedings of the International*

Conference on Artificial Intelligence and Statistics (AISTATS), Volume 5, 246–252. Citeseer.

Moro, Andrea and Roberto Navigli 2015. Semeval-2015 task 13: Multilingual all-words sense disambiguation and entity linking. *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, 288–297.

Morris, Jane and Graeme Hirst 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics* 17 (1): 21–48.

Murphy, M. Lynne 2003. *Semantic relations and the lexicon.* Cambridge: Cambridge University Press.

Nastase, Vivi and Stan Szpakowicz 2001. Word-sense disambiguation in Roget's Thesaurus using WordNet. *Workshop on WordNet and other lexical resources at NAACL.* Pittsburgh: ACL.

Navigli, Roberto 2009. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)* 41 (2): 10.

Navigli, Roberto and Mirella Lapata 2007. Graph connectivity measures for unsupervised word sense disambiguation. *Proceedings of the 20th international joint conference on Artifical intelligence*, 1683–1688. Morgan Kaufmann Publishers Inc.

Navigli, Roberto and Simone Paolo Ponzetto 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence* 193: 217–250.

Neelakantan, Arvind, Jeevan Shankar, Alexandre Passos and Andrew McCallum 2014. Efficient non-parametric estimation of multiple embeddings per word in vector space. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1059–1069. Doha, Qatar: Association for Computational Linguistics.

Nieto Piña, Luis and Richard Johansson 2015. A simple and efficient method to generate word sense representations. *Proceedings of the International Conference Recent Advances in Natural Language Processing*, 465–472. Hissar, Bulgaria.

Nieto Piña, Luis and Richard Johansson 2016. Embedding senses for efficient graph-based word sense disambiguation. *Proceedings of TextGraphs-10: the Workshop on Graph-based Methods for Natural Language Processing, NAACL-HLT 2016*, 1–5.

Nieto Piña, Luis and Richard Johansson 2017. Training word sense embeddings with lexicon-based regularization. *Proceedings of the*

Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers). Asian Federation of Natural Language Processing.

Nieto Piña, Luis and Richard Johansson 2018. Automatically linking lexical resources with word sense embedding models. *Proceedings of the Third Workshop on Semantic Deep Learning (SemDeep-3), COLING 2018*, 23–29. Association for Computational Linguistics.

Nieto Piña, Luis and Richard Johansson 2016. Benchmarking word sense disambiguation systems for swedish. *The sixth swedish language technology conference (SLTC), Umeå university.*

Padó, Sebastian and Mirella Lapata 2007. Dependency-based construction of semantic space models. *Computational Linguistics* 33 (2): 161–199.

Palmer, Martha, Hoa Trang Dang and Christiane Fellbaum 2007. Making fine-grained and coarse-grained sense distinctions, both manually and automatically. *Natural Language Engineering* 13 (2): 137–163.

Pantel, Patrick and Dekang Lin 2002. Discovering word senses from text. *Proceedings of the eighth acm sigkdd international conference on knowledge discovery and data mining*, 613–619. ACM.

Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake VanderPlas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot and Edouard Duchesnay 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12: 2825–2830.

Pelevina, Maria, Nikolay Arefiev, Chris Biemann and Alexander Panchenko 2016. Making sense of word embeddings. *Proceedings of the 1st Workshop on Representation Learning for NLP*, 174–183.

Pennington, Jeffrey, Richard Socher and Christopher Manning 2014. Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.

Peters, Matthew, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee and Luke Zettlemoyer 2018. Deep contextualized word representations. *Proceedings of the 2018 conference of the north american chapter of the association for computational linguistics: Human language technologies, volume 1 (long papers)*, 2227–2237. Association for Computational Linguistics.

Pilehvar, Mohammad Taher and Nigel Collier 2016. De-conflated semantic representations. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 1680–1690. Austin, Texas: Association for Computational Linguistics.

Pino, Juan and Maxine Eskenazi 2009. An application of latent semantic analysis to word sense discrimination for words with related and unrelated meanings. *Proceedings of the fourth workshop on innovative use of nlp for building educational applications*, 43–46. Association for Computational Linguistics.

Pitman, Jim 1995. Exchangeable and partially exchangeable random partitions. *Probability theory and related fields* 102 (2): 145–158.

Raganato, Alessandro, Jose Camacho-Collados and Roberto Navigli 2017. Word sense disambiguation: A unified evaluation framework and empirical comparison. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, Volume 1, 99–110.

Reichert, Richard, John Olney and James Paris 1969. Two dictionary transcripts and programs for processing them. Volume I. The encoding scheme, parsent and conix. Technical Report, Research report, System Development Corp., Santa Monica, CA.

Reisinger, Joseph and Raymond J Mooney 2010. Multi-prototype vector-space models of word meaning. *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 109–117. Association for Computational Linguistics.

Roget, Mark Peter 1852. *Thesaurus of English words and phrases.* London: Longman.

Rosenberg, Andrew and Julia Hirschberg 2007. V-measure: A conditional entropy-based external cluster evaluation measure. *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 410–420. Prague, Czech Republic.

Sahlgren, Magnus 2005. An introduction to random indexing. *Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering.*

Sahlgren, Magnus 2008. The distributional hypothesis. *Italian Journal of Disability Studies* 20: 33–53.

Schmidhuber, Jürgen 2015. Deep learning in neural networks: An overview. *Neural networks* 61: 85–117.

Schütze, Hinrich 1993. Word space. *Advances in neural information processing systems*, 895–902.

Schütze, Hinrich 1998. Automatic word sense discrimination. *Computational linguistics* 24 (1): 97–123.

Singhal, Amit 2012. Introducing the knowledge graph: things, not strings. `https://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html`. Accessed: 2018-12-25.

Smith, Raoul N and Edward Maxwell 1973. An English dictionary for computerized syntactic and semantic processing systems. *Proceedings of the 5th conference on Computational linguistics-Volume 1*, 303–322. Association for Computational Linguistics.

Socher, Richard, John Bauer, Christopher D Manning et al. 2013. Parsing with compositional vector grammars. *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Volume 1, 455–465.

Socher, Richard, Eric H Huang, Jeffrey Pennin, Christopher D Manning and Andrew Y Ng 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. *Advances in Neural Information Processing Systems*, 801–809.

Speer, Robert, Joshua Chin and Catherine Havasi 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. *AAAI*, 4444–4451.

Taghipour, Kaveh and Hwee Tou Ng 2015. One million sense-tagged instances for word sense disambiguation and induction. *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, 338–344. Beijing, China: Association for Computational Linguistics.

Tahmasebi, Nina and Thomas Risse 2017. Finding individual word sense changes and their delay in appearance. *Proceedings of recent advances in natural language processing 2017. varna, bulgaria 2–8 september, 2017 / edited by galia angelova, kalina bontcheva, ruslan mitkov, ivelina nikolova, irina temnikova.*

Tissier, Julien, Christophe Gravier and Amaury Habrard 2017. Dict2vec: Learning word embeddings using lexical dictionaries. *Conference on Empirical Methods in Natural Language Processing (EMNLP 2017)*, 254–263.

Turian, Joseph, Lev Ratinov and Yoshua Bengio 2010. Word representations: a simple and general method for semi-supervised learning. *Proceedings of the 48th Annual Meeting of the Association for Com-*

putational Linguistics, 384–394. Association for Computational Linguistics.

Turney, Peter D. and Patrick Pantel 2010. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research* 37: 141–188.

Vanhove, Martine (ed.) 2008. *From polysemy to semantic change: Towards a typology of lexical semantic associations.* Amsterdam: Jon Benjamins.

Véronis, Jean 2004. Hyperlex: lexical cartography for information retrieval. *Computer Speech & Language* 18 (3): 223–252.

Widdows, Dominic and Beate Dorow 2002. A graph model for unsupervised lexical acquisition. *Proceedings of the 19th international conference on computational linguistics-volume 1*, 1–7. Association for Computational Linguistics.

Wilks, Yorick 1998. Language processing and the thesaurus. *Proceedings national language research institute.* Tokyo. Also appeared as Technical report CS–97–13, University of Sheffield, Department of Computer Science.

Wu, Zhibiao and Martha Palmer 1994a. Verb semantics and lexical selection. *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, 133–138. Las Cruces, United States.

Wu, Zhibiao and Martha Palmer 1994b. Verb semantics and lexical selection. *Proceedings of the 32nd annual meeting of the association for computational linguistics*, 133–138. Las Cruces, New Mexico, USA.

Yaghoobzadeh, Yadollah and Hinrich Schütze 2016. Intrinsic subspace evaluation of word embedding representations. *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: Long papers)*, Volume 1, 236–246.

Yu, Mo and Mark Dredze 2014. Improving lexical embeddings with semantic knowledge. *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, Volume 2, 545–550.

Yu, Mo and Mark Dredze 2015. Learning composition models for phrase embeddings. *Transactions of the Association for Computational Linguistics* 3: 227–242.

Zhao, Ying and George Karypis 2001. Criterion functions for document clustering: Experiments and analysis. Technical Report TR 01-040, Department of Computer Science, University of Minnesota.