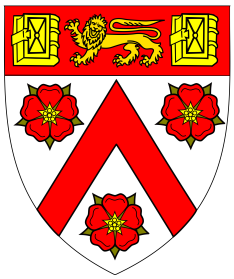
Data-Driven Language Understanding for Spoken Dialogue Systems



Nikola Mrkšić

Supervisor: Professor Steve Young

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
Doctor of Philosophy

Trinity College June 2018

To my mother, Živodarka Purić-Mrkšić

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and acknowledgements. Some of the material included in this thesis has been previously presented at international conferences and in peer-reviewed journals:

1. N. Mrkšić, D. Ó Séaghdha, B. Thomson, M. Gašić, P.-H. Su, D. Vandyke, T.-H. Wen and S. Young. *Multi-domain Dialog State Tracking using Recurrent Neural Networks.* In Proceedings of ACL 2015.

2. N. Mrkšić, D. Ó Séaghdha, B. Thomson, M. Gašić, L. Rojas-Barahona, P.-H. Su, D. Vandyke, T.-H. Wen and S. Young. *Counter-fitting Word Vectors to Linguistic Constraints.* In Proceedings of NAACL 2016.

3. N. Mrkšić, D. Ó Séaghdha, T.-H. Wen, B. Thomson and S. Young. *Neural Belief Tracker: Data-Driven Dialogue State Tracking.* In Proceedings of ACL 2017.

4. N. Mrkšić, I. Vulić, D. Ó Séaghdha, I. Leviant, R. Reichart, M. Gašić, A. Korhonen and S. Young. *Semantic Specialisation of Distributional Word Vector Spaces using Monolingual and Cross-Lingual Constraints.* Transactions of the Association for Computational Linguistics, Volume 5, 2017.

5. I. Vulić, N. Mrkšić, R. Reichart, D. Ó Séaghdha, S. Young and A. Korhonen. *Morph-Fitting: Fine-Tuning Word Vector Spaces using Simple Language-Specific Rules.* In Proceedings of ACL 2017.

6. N. Mrkšić and I. Vulić. *Fully Statistical Neural Belief Tracking.* In Proceedings of ACL 2018.

This dissertation contains fewer than 45,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 25 figures.

Nikola Mrkšić
June 2018

Acknowledgements

First and foremost, I would like to thank Steve Young, my PhD supervisor. Steve, thank you for the incredible three years, your support, all the inspiration, for trying to teach me how to be pragmatic, efficient, accountable and how to do good research. To Blaise Thomson, thank you for convincing me to join VocalIQ and do machine learning instead of investment banking. I would also like to thank Zoubin Ghahramani for showing me how fun machine learning can be and for introducing me to Blaise. I would also like to thank my examiners, Mirella Lapata and Phil Woodland, for their helpful comments and suggestions.

Next, I would like to thank Diarmuid Ó Séaghdha for being a great friend and mentor. Thanks for all the ideas, discussions, your rigorous research ethics, table football, maps of Cork and other forms of magical realism. To Ivan Vulić, thank you for sharing the adventures from Beijing to Haifa, as well as for our very productive research collaboration. I would also like to thank Roi Reichart for being a great collaborator, a passionate workaholic, as well as for the many Indian dinners.

To Wen Tsung-Hsien and Su Pei-Hao, thank you for all the inspiration, laughter, healthy competition, and the copious amounts of Chinese food. Thank you for choosing to start PolyAI with me - I look forward to seeing how far we can go.

The Dialogue Systems Group has been a great place to work. I thank Paweł for discussions of sixteenth century Poland and Iñigo for being a Basque separatist. I would also like to thank Milica, David, Stefan, Dongho, Lina, and especially Matt Henderson, whose work a large part of this thesis builds on. I would also like to thank my friends at Cambridge and elsewhere for keeping the world an interesting place: Stefan, Vladan, Ognjen, Dušan, Marko, Ribar, Bojke, Krki, Mare, Matko and many others. I would also like to thank my father Ljubomir for all his support, care and optimism.

I would also like to thank everyone at Trinity College, my second home, for the wonderful seven years at the College. In particular, I would like to thank Richard Serjeantson, Arthur Norman and Sean Holden for their support, as well as Trinity College itself for funding my undergraduate and postgraduate studies at Cambridge.

To Renata, my archbishop of Banterbury, pigeon, and best friend - thank you for all your love and support. Without you, these years would not have been so special.

Finally, this thesis is devoted to my mother, Živodarka Purić-Mrkšić, who made sure that I got here. Mama, thank you for all your sacrifices, patience, perseverance, love and care.

Abstract

Spoken dialogue systems provide a natural conversational interface to computer applications. In recent years, the substantial improvements in the performance of speech recognition engines have helped shift the research focus to the next component of the dialogue system pipeline: the one in charge of *language understanding*. The role of this module is to translate user inputs into accurate representations of the *user goal* in the form that can be used by the system to interact with the underlying application. The challenges include the modelling of linguistic variation, speech recognition errors and the effects of dialogue context. Recently, the focus of language understanding research has moved to making use of *word embeddings* induced from large textual corpora using unsupervised methods. The work presented in this thesis demonstrates how these methods can be adapted to overcome the limitations of language understanding pipelines currently used in spoken dialogue systems.

The thesis starts with a discussion of the pros and cons of language understanding models used in modern dialogue systems. Most models in use today are based on the *delexicalisation* paradigm, where exact string matching supplemented by a list of domain-specific rephrasings is used to recognise users' intents and update the system's internal *belief state*. This is followed by an attempt to use pretrained word vector collections to automatically induce domain-specific *semantic lexicons*, which are typically hand-crafted to handle lexical variation and account for a plethora of system failure modes. The results highlight the deficiencies of distributional word vectors which must be overcome to make them useful for downstream language understanding models.

The thesis next shifts focus to overcoming the language understanding models' dependency on semantic lexicons. To achieve that, the proposed *Neural Belief Tracking* (NBT) model forsakes the use of standard one-hot *n*-gram representations used in Natural Language Processing in favour of distributed representations of user utterances, dialogue context and domain ontologies. The NBT model makes use of external lexical knowledge embedded in semantically specialised word vectors, obviating the need for domain-specific semantic lexicons.

Subsequent work focuses on *semantic specialisation*, presenting an efficient method for injecting external lexical knowledge into word vector spaces. The proposed ATTRACT-REPEL algorithm boosts the semantic content of existing word vectors while simultaneously inducing high-quality *cross-lingual* word vector spaces. Finally, NBT models powered by specialised cross-lingual word vectors are used to train multilingual belief tracking models. These models operate across many languages at once, providing an efficient method for bootstrapping language understanding models for lower-resource languages with limited training data.

Contents

Chapter 1

Introduction

Since the early days of computing, the notion of *talking to machines* has captured human imagination and motivated generations of researchers. The applications and implications of an autonomous system capable of holding a conversation with a human being are numerous. They range from deep philosophical questions investigating the nature of consciousness and self-awareness, to the more practical applications such as large-scale automation of menial labour in call centres, technical support, and many other industries.

The Turing Test (1950) is one of the most influential criteria for *machine intelligence*, measuring the autonomous dialogue agent's ability to masquerade as a human being. In an attempt to pass the Turing test, Joseph Weizenbaum designed ELIZA (1966), a template-based dialogue agent designed to talk to users about their personal problems. Emulating the paradigm of Rogerian or *person-centred* psychotherapy, ELIZA aimed to provide an empathic environment for the patient to discover the answers to their problems through introspection, rather than by offering expert advice:

ELIZA: What is bothering you?
USER: I feel sad all the time.
ELIZA: Why do you feel sad all the time?
USER: I don't get along with my parents.
ELIZA: Tell me about your parents.

Weizenbaum designed ELIZA to show how shallow human-machine communication would be - the procedure consisted of less than 200 lines of code. However, when he let his personal assistant interact with ELIZA, she spent hours talking to the agent and revealing her most intimate problems. In fact, she believed there was a human psychotherapist on the other side of the interface, despite the fact that there was nothing genuinely intelligent about ELIZA apart from the skilfully written template rules which made it a good reflective listener.

Following the initial fascination with the Turing test, the research community moved on from using smoke and mirrors to feign human-like intelligence. As it became clear that building open-domain dialogue agents capable of conversing about any topic would be no easy feat, the focus of research shifted to a bottom-up paradigm. Instead of trying to mimic humans in general conversation, *task-oriented* dialogue systems could help users accomplish specific, well-defined tasks. By limiting dialogue systems to narrow domains, system designers could anticipate user behaviour and implement the required functionality to a relatively high standard. Examples of existing applications include spoken interfaces for call centres, booking systems, in-car navigation, and many others.

With the onset of the Fourth Industrial Revolution[1] and the adoption of new technologies such as smart phones, watches, homes, cars and others, personal assistants such as Apple's Siri, Google Assistant and Amazon's Alexa are permeating into every aspect of human life. These assistants allow users to achieve a plethora of tasks using their voice: playing music or films, using email, scheduling meetings, setting the room temperature, and many others.

In recent years, the capabilities of virtual assistants have improved dramatically. The most striking improvements have been in speech recognition, where the systems' *word error rates* already achieved parity with human-level performance. However, virtual assistants still struggle to understand users' goals and react in a way which would accomplish their demands. Large technological companies "solve" this problem by employing hundreds of engineers to add countless hand-crafted rules which deal with various user actions. This leads to opaque systems which require more maintenance as they become increasingly complex. This business model is incredibly expensive, becoming untenable as the number of potential application domains grows larger. Moreover, the effort is in large part replicated whenever these systems are deployed for another language.

Data-Driven Language Understanding The main goal of this thesis is to address these problems by introducing an improved data-driven paradigm for performing language understanding for spoken dialogue systems. The first part of the thesis reviews existing language understanding methods, highlighting the design flaws which limit their applicability to more complicated dialogue domains. The second part of the thesis shows how recent advances in neural network research can be used to construct data-driven models which overcome the limitations of existing language understanding models used in spoken dialogue systems.

[1] According to Klaus Schwab of the World Economic Forum: "*This Fourth Industrial Revolution is, however, fundamentally different. It is characterized by a range of new technologies that are fusing the physical, digital and biological worlds, impacting all disciplines, economies and industries, and even challenging ideas about what it means to be human.*"

The second objective of this thesis is to show that vectorial representations of words, known as *word embeddings*, are an effective building block for data-driven language understanding models. In support of this argument, semantic knowledge from large-scale semantic lexicons such as WordNet (Miller, 1995) is injected into word embeddings and seamlessly used by the downstream neural networks to improve their language understanding capabilities. Subsequently, linguistic relations from multilingual dictionaries are used to induce cross-lingual word embeddings which allow the proposed language understanding models to operate across different languages at once, while requiring no language-specific input from system designers.

1.1 Thesis Outline

The topic of this thesis is *language understanding* for statistical spoken dialogue systems. These systems are usually modular, consisting of six components: **1)** speech recognition; **2)** spoken language understanding; **3)** dialogue state tracking; **4)** dialogue management; **5)** natural language generation; and **6)** text-to-speech synthesis. Chapter 2 gives an overview of spoken dialogue systems theory, as well as recent trends in the design of each system component. Special focus is given to the spoken language understanding (SLU) and dialogue state tracking (DST) components, which form the backbone of language understanding in modern dialogue systems. Brief summaries of subsequent chapters are given next.

Chapter 3: Word-Based Belief Tracking This chapter presents current state-of-the-art neural network approaches for language understanding in dialogue systems. Its focus is on *delexicalisation*, a technique which treats all user intents defined by the domain ontology using the same model parameters, relying on little more than exact matching and context to perform language understanding. This idea is then extended to train multi-domain belief tracking models, as well as to bootstrap models for new dialogue domains.

Chapter 4: Inducing Semantic Lexicons This chapter shows how pre-trained *word embeddings* can be used to induce semantic lexicons, which are lists of rephrasings that delexicalisation-based models use to deal with linguistic variation. The pre-trained word vectors are first calibrated to capture true semantic similarity, and then used to construct semantic lexicons which improve language understanding performance across two dialogue domains. The chapter concludes by discussing the limitations of delexicalisation-based models powered by domain-specific semantic lexicons.

Chapter 5: Neural Belief Tracker This chapter presents the Neural Belief Tracker (NBT), a language understanding model designed to move past the word-based delexicalisation paradigm and allow language understanding models to naturally scale to linguistically rich user input across disparate dialogue domains and different languages. To do this, the Neural Belief Tracker reasons purely over pre-trained word embeddings, learning to compose them into distributed representations of user utterances and dialogue context which are subsequently used to infer user goals. In doing that, the NBT models make use of the semantic content embedded in pre-trained word vectors, removing the dependency on exact matching and domain-specific semantic lexicons.

Chapter 6: Exploiting Lexical Resources This chapter focuses on *semantic specialisation* of word vectors, presenting a novel specialisation algorithm termed ATTRACT-REPEL. This method can inject constraints from mono- and cross-lingual resources into existing word vectors to create semantically specialised cross-lingual vector spaces. The presented evaluation shows that the method can make use of existing cross-lingual lexicons to construct high-quality vector spaces for a plethora of different languages, facilitating semantic transfer from high- to lower-resource ones. Its effectiveness is demonstrated by achieving state-of-the-art results on semantic similarity datasets across six languages.

Chapter 7: Belief Tracking across Languages Dialogue system research has traditionally focused on English, leaving the problem of deploying existing language understanding models to other languages relatively unexplored. This chapter investigates the important factors for applying the research presented in previous chapters to two new languages: Italian and German. The first part of the chapter studies the interplay between semantic specialisation of word embeddings and downstream language understanding performance. The second part of the chapter goes further, investigating the importance of modelling morphological phenomena for achieving robust performance in languages with complex morphology.

Chapter 8 concludes the thesis, giving an overview of the presented research and summarising the main contributions. It discusses the strengths and weaknesses of the presented data-driven language understanding paradigm in the context of real-world personal assistants. The last part of the chapter outlines potential directions for future work.

Chapter 2

Statistical Spoken Dialogue Systems

This chapter gives an overview of statistical spoken dialogue systems and their core components, with a particular focus on the role of the language understanding modules.

2.1 Modular Spoken Dialogue Systems

Typical spoken dialogue systems can be thought of as pipelines connecting their principal components (Figure 2.1). One iteration through this pipeline corresponds to one *dialogue turn*, which consists of two utterances: one by the user, the other by the dialogue system. The long-term research goal of work on end-to-end *statistical dialogue systems* is to design all the system components as statistical models with parameters estimated from data (Young, 2010b). Such design would allow all the system components to handle uncertainty in both their input and their output (Young et al., 2013). This chapter first introduces the notation required to understand the operation of each component. This is followed by a survey of the main research directions pertaining to each system module.

2.1.1 Task-Oriented Dialogue Systems

The work presented in this thesis investigates task-oriented dialogue systems.[1] Task-oriented systems are primarily designed to search and interact with large databases which contain information pertaining to a certain *dialogue domain*. Examples of dialogue domains include flight booking (Hemphill et al., 1990), restaurant search (Williams, 2012b) or tourist information (Henderson et al., 2014b).[2]

[1] This term is used interchangeably with *goal-oriented* dialogue systems.

[2] Alternative *chat-bot* style systems do not make use of task ontologies or the pipeline model. Instead, these models learn to generate/choose system responses based on previous dialogue turns (Kannan and Vinyals, 2017;

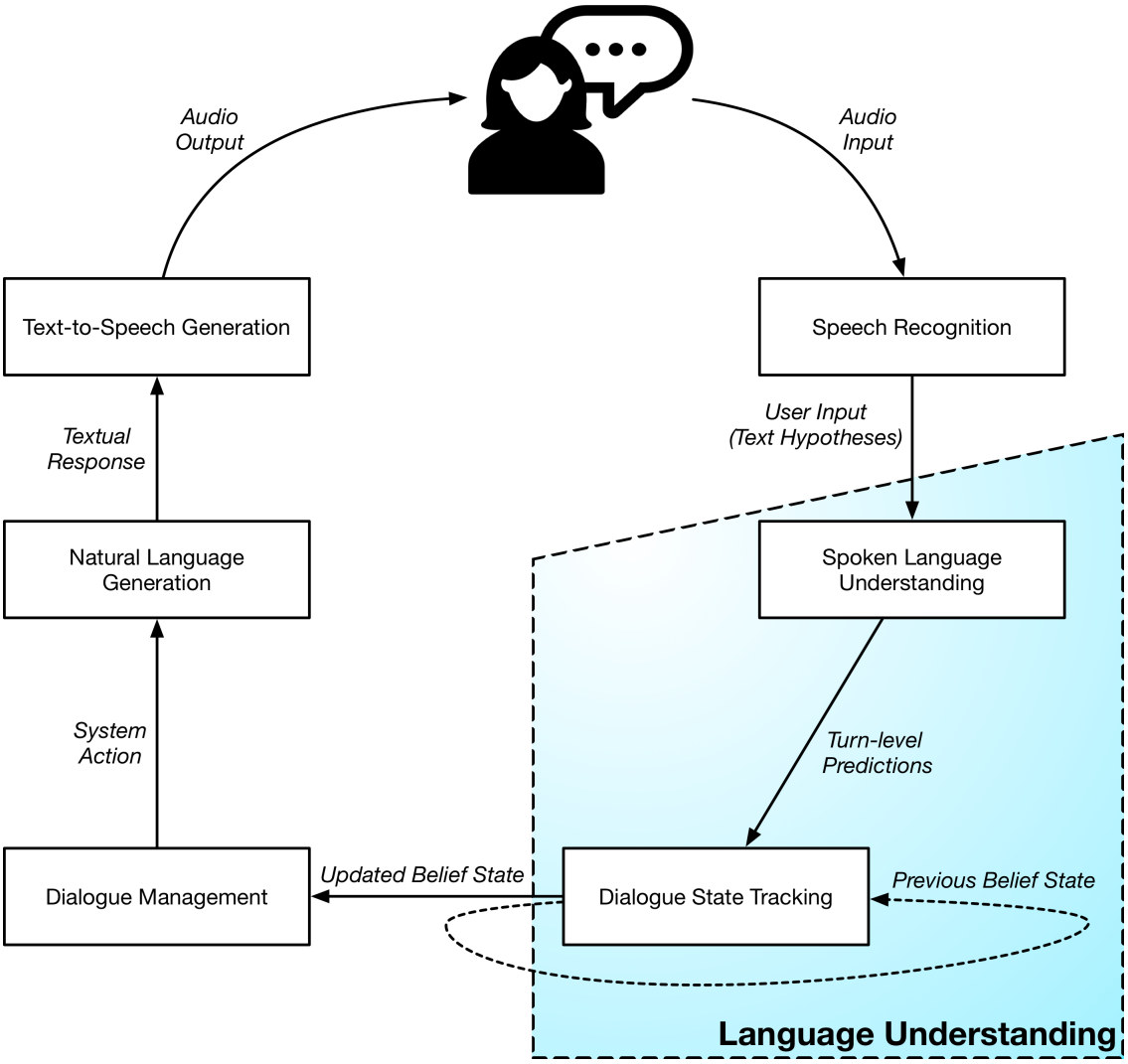


Figure 2.1 The pipeline of the core components in statistical spoken dialogue systems.

The slots of *slot-filling-based* dialogue systems fully define the dialogue domain, specifying what the system can talk about and all the tasks it can help the users with. The slots determine all the actions the system can take, the possible semantics of the user utterances, and the possible states of the dialogue (Henderson, 2015a).

2.1.2 Domain Ontologies

The *domain ontology* of a dialogue system contains the information required to model user goals expressed up to a given point of the conversation. This collection of slots is referred to as the *dialogue state*. For any dialogue domain, the ontology is made up of informable slots S_{inf} and the set of requestable slots S_{req}.[3] Figure 2.2 shows the ontology used by the belief tracker for the Cambridge Restaurants dialogue domain.

The informable slots represent the attributes of entities in a database which the user can use to constrain the search, whereas the requestable slots represent the entities' attributes which the users can ask about, but not necessarily use as search constraints. For example, in a restaurant search domain, users might be able to search for restaurants with a specific food type, but may only be able to ask about the address once the dialogue system comes up with a restaurant suggestion.

The ontology therefore consists of the set of requestable slots S_{req}, the set of informable slots S_{inf}, and a set of slot values V_s for each $s \in S_{inf}$. The language understanding component does not need access to the sets of values for requestable slots that are not informable, for example the *addresses* of all restaurants in the database. This is because the user cannot constrain the search using these values. Instead, one can only inquire about these values once a database entity has been chosen. The dialogue manager component handles requests for such attributes, as well as any other interaction with the database.

2.1.3 Dialogue Act Taxonomy

Dialogue acts serve to encode the semantics of both user utterances and system prompts. In slot-based systems, they consist of a *dialogue act type* and a set of *slot-value* pairs which represent the *dialogue act arguments* (Henderson, 2015a).

Dialogue act types represent the general action of the utterance. The systems investigated in this thesis rely on three basic types of actions, which correspond to informing

Lowe et al., 2015; Serban et al., 2016a,b; Vinyals and Le, 2015). This means these models cannot interact with databases or react to user queries different from those encountered in their training data.

[3] In all domains considered in this thesis, $S_{inf} \subseteq S_{req}$, though in general these two sets can be disjoint.

```
INFORMABLE SLOTS: {
    PRICE RANGE: [
        cheap,
        moderate,
        expensive
    ],
    AREA: [
        centre,
        north,
        west,
        south,
        east
    ],
    FOOD: [
        Afghan, African, Afternoon Tea, Asian, Australian,
        Austrian, Barbeque, Basque, Belgian, Bistro, Brazilian,
        British, Cantonese, Caribbean, Catalan, Chinese,
        Christmas, Corsican, Creative, Crossover, Cuban,
        Danish, Dutch, English, Eritrean, French, Fusion ...
    ],
    NAME: [
        Ali Baba, Anatolia, Ask, Backstreet Bistro,
        Bangkok City, Bedouin, Bloomsbury,
        Caffe Uno, Cambridge Lodge, Charlie Chan,
        Chiquito Restaurant Bar, City Stop Restaurant,
        Clowns Cafe, Cocum, Cote, Curry Garden,
        Curry King, Curry Prince, Curry Queen.....
    ]
}
REQUESTABLE SLOTS: [
    Postcode,
    Address,
    Area,
    Food,
    Phone,
    Price Range,
    Signature,
    Name
]
```

Figure 2.2 A subset of the domain ontology for the Cambridge Restaurants dialogue domain. This is the underlying ontology for most dialogue datasets used in evaluation across this thesis. The full ontology has 112 name values, 92 food values (including the special *dontcare* value), 4 price range values, 6 area values and 8 requestable values.

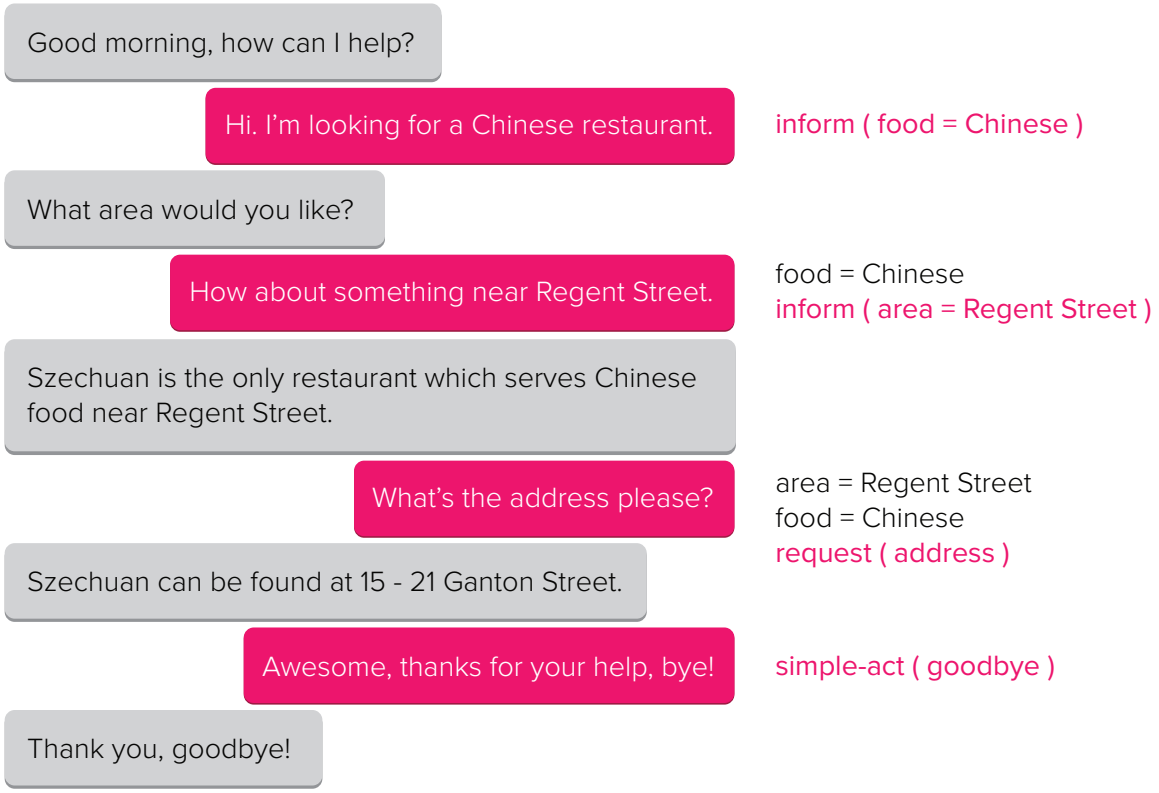


Figure 2.3 User goals in a dialogue are represented as sets of constraints expressed by slot-value pairs. The pink dialogue acts show the current SLU output, and the black text shows the slot value constraints the model remembers from previous dialogue turns.

constraints, such as inform(*food*=*Serbian*), requesting information, such as request(*address*), and performing simple dialogue acts such as saying *hi*, *thank you* and *goodbye*.

The dialogue state at any point of the dialogue consists of a number of dialogue acts (with the corresponding semantic information given by slot value pairs). Figure 2.3 gives an example conversation with the correct dialogue state annotation after each dialogue turn.

2.1.4 Uncertainty in Dialogue

Dialogue systems can be thought of in terms of analogies with computer networks, which also facilitate noisy communication between two end-nodes (Kurose and Ross, 2002). In our case, the two end-nodes are the user and the dialogue system. There are three communication layers in the dialogue system, each corresponding to a different level of abstraction. As shown in Figure 2.1, these layers are:

1. **Physical Speech**: sound waveforms taken as input by the automated speech recognition (ASR) module and produced as output by the text-to-speech (TTS) module.

2. **Natural Language**: textual representation of user queries produced by the speech recogniser and passed to the language understanding module, which subsumes Spoken Language Understanding (SLU) and Dialogue State Tracking (DST). Natural language in the form of text is also produced as output by the Natural Language Generator (NLG), which uses system acts produced by the Dialogue Manager (DM) as input.

3. **Dialogue Acts**: domain-specific meaning representations which provide a formal language for expressing user goals and the produced system responses. This formalisation allows the system to interface between the user's intents and goals on one side, and the external database which contains the required information on the other.

As in computer networking, moving to higher levels of abstraction introduces additional uncertainty into the representation of user goals. There are two main sources of uncertainty in dialogue systems: **a)** the uncertainty introduced by imperfect speech recognition, which gets progressively worse as dialogue systems are deployed in noisy environments (e.g., cars or public transport); and **b)** the noise introduced by the language understanding modules, which may have difficulty handling linguistic variation or interpreting contextual feedback.

The language understanding component, which is the main focus of this thesis, is instrumental in dealing with and modelling both kinds of noise. We next discuss the speech recogniser, spoken language understanding and the dialogue state tracking components, focusing on the kinds of uncertainty introduced by each component.

2.2 Automatic Speech Recognition

The automatic speech recognition (ASR) module is the first component in the pipeline architecture of spoken dialogue systems. The role of this module is to transform the recorded waveform representations of user utterances into textual output.

Over the last two decades, most state-of-the-art ASR systems relied on Hidden Markov Models (HMMs). These sequential statistical models provide an efficient framework for estimating the most probable word sequence for the provided acoustic input (Gales and Young, 2007). In recent years, these have been superseded by models based on deep neural networks, both for estimating HMM state probabilities (Bourlard and Morgan, 1993; Dahl et al., 2012; Hinton et al., 2012) and for end-to-end speech recognition using Recurrent Neural Networks (RNNs) (Deng et al., 2013; Graves and Jaitly, 2014; Miao et al., 2015).

The advent of deep learning in speech recognition has led to substantial improvements in word error rates (WER). In fact, some systems already claim to achieve human parity on well-known speech recognition datasets such as Switchboard Hub5 2000.[4] However, spoken dialogue systems are often deployed in very noisy environments (e.g., cars, public transport, loud workplaces), where the word error rates are significantly higher, ranging from 8.9% to 23.4% (Alam et al., 2015). Coupled with the fact that non-English ASR systems are still lagging behind English (Ghoshal et al., 2013), it is clear that handling ASR noise remains a pertinent challenge for the construction of spoken dialogue systems.

The speech recognition component of the dialogue system can be bootstrapped using publicly available third-party toolkits. Notable examples include the HTK toolkit (Young et al., 2002), CMUSphinx (Walker et al., 2004) and more recently Kaldi (Povey et al., 2011). In the Dialog State Tracking Challenge 2, speech recognition Word Error Rates ranged from 22.4% to 40.4%, highlighting the importance of handling noisy speech input for the design of robust spoken dialogue systems (Henderson et al., 2014a).

2.2.1 Forms of ASR Output

ASR models assign posterior probabilities to words in an utterance given its recorded acoustics (Jurafsky and Martin, 2009). The ASR output is often inaccurate, as it is affected by factors like outside noise, variation between different users, imperfections of the equipment used to record the waveforms, and many others.

Different forms of ASR system output vary in how well they approximate the full posterior distributions over the ASR hypotheses. Popular output encodings include:

- **N-best lists**: this form of output approximates the full posterior distribution by returning the top N most probable hypotheses with their respective probabilities. The top N hypotheses are typically very similar, with the low-scoring words unlikely to feature in any of them.

- **Word lattices** are directed acyclic graphs with weighted edges encoding the scores of recognised words (Murveit et al., 1993). Each path through the word lattice encodes an utterance hypothesis (and can be used to compute its score).

- **Word Confusion Networks** are restricted forms of word lattices: all paths from the start to the end node must pass through all the nodes (Mangu et al., 2000). The

[4] As of February 2018, software companies like IBM (5.5%), Microsoft (5.1%) and Capio (5.0%) have reported conversational word error rates on par or better than humans (5.9%) on the Switchboard Hub5 2000 evaluation dataset (Han et al., 2018; Saon et al., 2017; Xiong et al., 2017).

constrained structure of the confusion network allows for efficient computation of posterior probabilities for any given word sequence.

The more informative the summary of the posterior distribution over the words in an utterance is, the better the subsequent language understanding models can represent the uncertainty about the possible user utterances. For instance, Henderson et al. (2012a) showed that the use of full posterior distributions over ASR hypotheses (encoded using word confusion networks) results in significant performance gains for the task of *semantic decoding* (converting ASR output to turn-level dialogue act representations).

The work presented in this thesis is limited in scope to ASR systems which produce N-best lists as output. Combining predictions for this form of output is straightforward: given the prediction for each hypothesis, a linear combination of N predictions weighted by the ASR confidence scores provides the final output. The performance of the proposed methods would certainly benefit from modelling uncertainty using richer ASR output. However, the primary focus of the presented work is modelling aspects of uncertainty introduced by the next pipeline component, when textual representations of natural language are converted into ontology-defined dialogue acts.

2.3 Spoken Language Understanding

At the highest level of communication, the user and the dialogue system are conveying intents, exchanging information and suggesting actions to each other. These intents are represented using *dialogue acts*, which use an abstract language (defined by the ontology) to describe the current application domain (see Section 2.1.3).

The role of the Spoken Language Understanding (SLU) module, also known as the *semantic decoder*, is to convert ASR output into dialogue acts. The subsequent Dialogue State Tracking (DST) module uses these dialogue acts to *update* the system's *belief state*, which is a probability distribution over all possible states of the dialogue. This distribution is then used by the *dialogue manager* to choose an appropriate system response.

Linguistic Uncertainty Translating the ASR output into abstract meaning representations defined by the domain ontology propagates uncertainty introduced by the speech recogniser. On top of that, the semantic decoding step introduces additional sources of uncertainty. The first is related to the difficulty of interpreting natural language, that is by linguistic phenomena difficult to model. Rephrasing is one example of these: the SLU model should learn that an utterance such as "*How about Sushi*" should map to FOOD=*Japanese*. Another form of

uncertainty is due to modelling context - if the user asks "*Would you like Turkish food?*" and the user responds with "*Why not?*", the semantic decoder should output FOOD=*Turkish*.

2.3.1 Overview of SLU Approaches

Historically, rule-based methods such as template matching and grammar-based methods have been used for semantic decoding (Ward and Issar, 1994; Young and Proctor, 1989). Such rules are hard to design, requiring a substantial amount of user testing before achieving satisfactory performance in any given dialogue domain.

Unlike rule-based methods, data-driven ones can process long-range dependencies by learning to parse spontaneous conversational input. By learning from annotated data, statistical models can learn to overcome erroneous speech recognition by recognising frequently occurring ASR errors. Proposed data-driven methods make use of a plethora of techniques, including Inductive Logic Programming (Zelle and Mooney, 1996a), Generative Probabilistic Models (He and Young, 2005), Weighted Finite State Transducers (Jurčíček et al., 2009), Support Vector Machines (Mairesse et al., 2009), Combinatorial Categorial Grammars (Kwiatkowski et al., 2011; Zettlemoyer and Collins, 2007), and many others.

SLU for Spoken Dialogue Systems In the recent Dialogue State Tracking Challenges (DSTC) (Williams et al., 2016), some systems used the output of template-based matching systems such as Phoenix (Wang, 1994), which was provided as the baseline SLU model. However, more robust and accurate statistical SLU systems have been used in spoken dialogue system design. Many of these approaches train independent binary models that decide whether each slot-value pair was expressed in the user utterance. Given enough data, these models can learn which lexical features are good indicators for a given value and can capture elements of paraphrasing (Mairesse et al., 2009). This line of work later shifted focus to robust handling of rich ASR output (Henderson et al., 2012b; Tur et al., 2013).

SLU as Sequence Labelling SLU has also been treated as a *sequence labelling* problem, where each word in an utterance is labelled according to its role in the user's intent; standard labelling models such as Conditional Random Fields (CRFs) or Recurrent Neural Networks (RNNs) can then be used (Celikyilmaz and Hakkani-Tur, 2015; Liu and Lane, 2016a,b; Mesnil et al., 2015; Peng et al., 2015; Raymond and Ricardi, 2007; Vu et al., 2016; Yao et al., 2014; Zhang and Wang, 2016, i.a.). Other approaches adopt a more complex modelling structure inspired by semantic parsing (Saleh et al., 2014; Vlachos and Clark, 2014).

Most SLU methods have very substantial resource requirements. Models in the first group are very data-hungry, as they learn independent parameters for each slot-value pair. The second group of models relies on fine-grained, high-quality manual annotation at the word level, which hinders scaling to larger, more realistic application domains.

2.4 Dialogue State Tracking

Models for probabilistic dialogue state tracking, or *belief tracking*, were introduced as components of spoken dialogue systems in order to better handle noisy speech recognition and other sources of uncertainty in understanding a user's goals (Bohus and Rudnicky, 2006; Williams and Young, 2007; Young et al., 2010). Modern dialogue management policies can learn to use a tracker's distribution over intents to decide whether to execute an action or request clarification from the user.

The Dialogue State Tracking Challenge (DSTC) shared tasks have spurred research on this problem and established standard evaluation paradigms. The first challenge provided a corpus of dialogues in the Pittsburgh *bus timetable information* domain (Williams et al., 2013). The second challenge used a restaurant search domain and incorporated user *goal changes* which were not present in the first challenge (Henderson et al., 2014a). The third challenge evaluated the models' domain adaptation capabilities: a new *tourist information* domain was used for evaluation, with the DSTC 2 data used for training along with a small number of dialogues from the new domain (Henderson et al., 2014b).[5] All dialogue corpora used for evaluation in this thesis follow the shared DSTC 2/3 format, presented next.

Dialogue State Definition For slot-based systems, the dialogue state is represented by the list of constraints that the user has expressed up to that point in time. In the Dialogue State Tracking Challenges 2 and 3 (Henderson et al., 2014a,b), the task is defined by an *ontology* that enumerates the goals a user can specify and the attributes of entities that the user can request information about. In this setting, the dialogue state consists of:

1. Goal constraints for each of the informable slots $s \in S_{inf}$. Each goal constraint is either a value $v \in V_s$, or one of the special values: *dontcare* or *none-specified*. Examples of such constraints in the restaurant search domain, following a user utterance such as *"How about a Greek place somewhere north"* would be FOOD=*Greek*, AREA=*north*.

[5] More recently, Dialogue State Tracking Challenges 4 and 5 have taken place. However, the focus of these challenges has shifted to modelling *human-human* conversation. DSTC 4 used Skype conversations between tour guides and tourists (Kim et al., 2016c), while DSTC 5 focused on cross-lingual DST, using English dialogues for training and Chinese ones for evaluation (Kim et al., 2016d).

2. A subset of the requested slots $S' \subseteq S_{req}$, indicating which of the requested slots' values the user would like to know about. In the restaurant search domain, following a user utterance such as *"What are the address and phone number"*, the set of requested slots would be given by: {*address, phone*}. Requested slots represent questions about previously suggested entities, and as such do not require belief tracking *across turns*.

3. The dialogue search method, which can be: **1)** *by constraints*, if the user is specifying target entity attributes; **2)** *by alternatives*, if the user is asking for alternative venues which meet the given constraints; **3)** *by name*, if the user is asking for a specific venue; and **4)** *finished*, which means the user wants to end the conversation.

In traditional (modular) spoken dialogue systems, at each dialogue turn, the SLU component extracts new (noisy) dialogue acts which represents the constraints expressed in the new user utterance. The DST component uses the new dialogue acts to update the belief state, which is the *probability distribution* over the possible dialogue states described above. Figure 2.3 in Section 2.1 showed a sample dialogue with user utterances annotated with the correct dialogue state. The role of the DST model is to infer the best possible estimate of the dialogue state, given the history of user and system utterances in the current dialogue.

Propagating Uncertainty The belief state update incorporates all the sources of uncertainty into the updated belief state. The uncertainty is propagated from the previous dialogue turns (by using the previous belief state), as well as from the output of the ASR and the SLU components operating on the current user input. The newly inferred belief state is then passed on to the POMDP-based dialogue manager (discussed in Section 2.5), which can use the full distribution to learn optimal policies for operating under varying levels of uncertainty.

Hand-Crafted Dialogue State Trackers

The early approaches to performing dialogue state tracking were based on *hand-crafted* systems which used only the top SLU/ASR hypothesis to map the existing dialogue state to a new dialogue state (Williams et al., 2016). An early example of this approach is the Information State Update model, which used hand-crafted rules to maintain the *information state* (Larsson and Traum, 2000). This was one of the first attempts to formalise the structure of the input needed to perform dialogue management. Another example of hand-crafted systems was MIT Jupiter (Zue et al., 2000), a weather information system which relied on hand-made rules to update the system variables.

One advantage of hand-crafted dialogue state trackers is that they require no dialogue data to tune their internal parameters. Moreover, they allow system designers to directly encode

specific desirable behaviour which could override specific system failure modes caused by imperfect ASR or SLU modules. However, a major drawback of early hand-crafted systems was their inability to consider (and maintain) multiple hypotheses for the dialogue state, and therefore model the uncertainty stemming from the preceding system components.

To overcome these issues, more recent rule-based approaches predict an updated dialogue state for each of the hypotheses provided by the ASR/SLU pipeline, and then use another hand-crafted rule to combine these estimates into a single belief state (Sun et al., 2014; Wang and Lemon, 2013). However, these approaches still do not learn from available dialogue data; in fact, their performance does not benefit in any way from the collected real-world dialogues. As a result, these systems require careful fine-tuning whenever the ASR/SLU components is modified, or whenever the system is deployed for a new dialogue domain. To overcome these issues, Sun et al. (2016) use *recurrent polynomial networks* to incorporate prior knowledge in the form of hand-crafted rules into a hybrid statistical framework. However, the performance of this approach still hinges on the quality of the initial rule-based system and the reliability of the SLU module, limiting its capacity for generalising to new dialogue domains.

The deficiencies of hand-crafted approaches have spurred research on *data-driven* DST methods, which fall into two camps: *generative* and *discriminative*.

2.4.1 Generative Dialogue State Trackers

Historically, the most popular data-driven methods were based on generative Bayesian networks. These methods model dialogue as a dynamic Bayesian network: the (true) dialogue state s_t (at time t) is treated as a hidden random variable (Henderson, 2015b). The system action a_t and the observed user action o_t are the observed variables. At each dialogue turn, Bayesian inference is used to obtain an updated estimate of the dialogue state s_{t+1}. The dialogue state itself is one of $s \in S$, where S is the Cartesian product of all possible slot values, requestable slots and the search methods. Adopting the notation of Williams and Young (2007), let b' denote the updated belief state (for time $t+1$), and let b denote the belief state at time t. The belief state update can then be expressed as:

$$
\begin{aligned}
b'(s_{t+1}) &= P(s_{t+1} \mid o_{t+1}, a_t, b) && (2.1)\\
&= \frac{P(o_{t+1} \mid s_{t+1}, a_t, b) P(s_{t+1} \mid a_t, b)}{P(o_{t+1} \mid a_t, b)} && (2.2)\\
&= \frac{P(o_{t+1} \mid s_{t+1}, a_t) \sum_{s \in S} P(s_{t+1} \mid a_t, b, s) P(s \mid a_t, b)}{P(o_{t+1} \mid a_t, b)} && (2.3)\\
&= \frac{P(o_{t+1} \mid s_{t+1}, a_t) \sum_{s \in S} P(s_{t+1} \mid a_t, s) b(s)}{P(o_{t+1} \mid a_t, b)} && (2.4)
\end{aligned}
$$

In these equations, $P(o_{t+1} \mid s_{t+1}, a_t)$ is the probability that the ASR/SLU pipeline produces the output o_{t+1} given an underlying (true) state s_{t+1} and following system act a_t. $P(s_{t+1} \mid a_t, s)$ is the probability of moving to state s_{t+1} from state s (again, following system act a_t). These equations show that the belief state at time $t+1$ provides a *sufficient statistic* of dialogue history given the previous system and user actions: it is Markovian with respect to $a_t, o_t, a_{t-1}, o_{t-1}, \ldots, a_1, o_1$. Consequently, the belief state at time $t+1$ encompasses the full dialogue history and as such provides a complete representation that subsequent planning (dialogue management) algorithms need to choose the next system action a_{t+1}.

Equations 2.1 - 2.5 summarise the approach proposed by Williams and Young (2007). Over the last ten years, many different factorisations of the Bayesian Network and its hidden dialogue state have been proposed. For instance, DeVault and Stone (2007) use a Bayesian Network which models both the observed user action and the underlying intention separately, using separate terms to capture *common-sense* relationships between intentions, actions and likely dialogue states. A common feature of these approaches is that the system designer encodes his knowledge of conversational dynamics into the employed Bayesian network. Its parameters are then learned from data, using approaches such as Expectation Maximisation (Syed and Williams, 2008) or Expectation Propagation (Thomson et al., 2010).

The basic form of the update equation is quadratic in the number of dialogue states. As such, it makes the naïve version of this approach infeasible for systems with a large number of dialogue states (i.e., real-world domain ontologies with many slots). This issue is exacerbated by the fact that such systems are deployed in real-time applications. The approaches proposed to overcome this drawback fall into two camps: **1)** those that introduce additional factorisations which further limit the kinds of behaviour that the Bayesian network can model (Bui et al., 2009; Thomson and Young, 2010; Williams and Young, 2007); and **2)** those that maintain a list (i.e., *a beam*) of the most likely dialogue states at each point of the conversation. Notable examples of the latter approach include the Hidden Information State model (Gašić and Young, 2011; Young, 2010a; Young et al., 2007), Mixture Model POMDPs (Henderson and Lemon, 2008), Probabilistic Ontology Trees (Mehta et al., 2010), and Dynamic Probabilistic Ontology Trees (Raux and Ma, 2011). Lee and Kim (2016) recently proposed a generative model which uses Long-Short Term Memory (Hochreiter and Schmidhuber, 1997) networks to model dialogue history, extending their previous approach which used a weighted softmax function to predict the dialogue state (Lee et al., 2014).

Despite the substantial body of work on making generative DST models tractable, these models still cannot scale to consider large numbers of (potentially) useful features from ASR, SLU and dialogue history. This is because all dependencies between features are modelled explicitly, which requires infeasible amounts of dialogue data (Williams et al., 2016). On

the other hand, the tractable variants of these models make independence assumptions or approximations which affect learning adversely. One instance of such assumptions is that the errors are drawn independently from a uniform distribution, which is clearly false. In the case of ASR, a request for *Serbian food* is more likely to be confused with *Siberian* or *Syrian*, rather than with requests for *Czech*, *Romanian* or *Taiwanese* food. These limitations of generative models led to increased interest in *discriminative* DST models.

2.4.2 Discriminative Dialogue State Trackers

Unlike generative models, discriminative models directly estimate the conditional probability $P(s_t \mid \mathbf{f})$, where $\mathbf{f}$ is the set of (arbitrary) features representing ASR, SLU and dialogue history. The relation between these features need not be specified by the system designer: instead, they are learned directly from *labelled* dialogue data. This allows discriminative models to consider very large sets of potentially correlated features.

Until recently, very few discriminative models were suggested, with the notable exception of the maximum entropy model proposed by Bohus and Rudnicky (2006). This method used a *generalized linear model* to determine which of the many hand-crafted features are most relevant for the belief state update. The Dialogue State Tracking Challenges saw a variety of novel discriminative approaches. These can be broadly split into models which treat dialogue state tracking as a turn-by-turn *classification* problem, and those that model dialogue as a *sequential process*.

DST as Turn-by-Turn Classification

Dialogue State Tracking can be framed as a classification problem, where the task of the classifier is to estimate $P(s_t \mid \mathbf{f}_1, \ldots, \mathbf{f}_t)$, i.e., the most probable dialogue state at time t given the collection of SLU, ASR and system act features up until the current dialogue turn (Henderson, 2015c). The main appeal of this *static classification* approach is that any classification paradigm can be applied once the sequence of observations $\mathbf{f}_1, \ldots, \mathbf{f}_t$ is turned into a fixed-length summary representation. Given this succinct summary of the preceding dialogue turns, a plethora of approaches can be used to predict the new dialogue state. These include maximum entropy models (Lee and Eskenazi, 2013; Metallinou et al., 2013; Williams, 2012a, 2013), neural networks (Casanueva et al., 2016a,b; Henderson et al., 2013) or web-style ranking (Williams, 2014), which achieved the best accuracy in the second Dialogue State Tracking Challenge.

Summary Feature Functions The central research question of approaches based on static classification is that of specifying feature functions which map the sequence of actions and observations $\mathbf{f}_1, \ldots, \mathbf{f}_t$ into a single fixed-length representation for the current dialogue turn. For each slot S in the dialogue ontology, this representation usually consists of: **1)** a *general* set of features $\mathbf{f}_t^*$ which aims to represent dialogue history up to turn t; and **2)** a collection of slot-value dependent features $\mathbf{f}_t^v$ for all slot values $v \in S$. As an example of the former, Metallinou et al. (2013) use manually-specified general features which include:

- the number of distinct SLU predictions up to turn t;
- the entropy of probabilities assigned by the SLU module up to turn t;
- the size and entropy of the SLU confidence scores at turn t;
- the mean and variance of the length of the SLU N-best list across all previous turns;
- the posterior probabilities of the ASR hypotheses.

As an example of slot-value specific features $\mathbf{f}_t^v$, which capture information which the model can use to decide whether a specific value $v \in S$ is part of the belief state (i.e. that $s_t = v$), Henderson et al. (2013) use features which include:

- SLU score: the probability that $s_t = v$, assigned by the SLU module;
- Rank score: $\frac{1}{r}$, where r is the rank of $s_t = v$ in the SLU N-best list, or 0 otherwise;
- Affirm score: the probability assigned to an *affirm* action if the system just asked the user whether $s_t = v$, or 0 otherwise;
- Negate score: the probability assigned to a *negate* action if the system just asked the user whether $s_t \neq v$, or 0 otherwise.

To deal with the fact that there are different feature vectors for each value $v \in S$, static classifiers typically tie the subsequent parameters which interact with each feature vector $\mathbf{f}_t^v$. This facilitates value-independent behaviour which allows these models to deal with unseen data. For example, if S is the AREA slot and v=*centre*, and if the SLU probability for AREA=*centre* is high, the DST module should recognise that the user is asking for a venue in the city centre, even though it may not have seen this example during training.[6]

[6]Note that this approach leaves the responsibility for deciding that AREA=*centre* to the SLU module. This may be simple when an exact match with the given value is provided (e.g., if the user asks for *a place in the centre*), but less so when the user asks for something *central* or *downtown*.

Dialogue as a Sequential Process

In contrast to the previous group of approaches, this group of methods casts Dialogue State Tracking as a *sequence labelling* problem. Ren et al. (2014a,b) use the previous belief state as a feature in the framework of *discriminative Markov Models*. Alternatively, linear Conditional Random Fields (Lafferty et al., 2001) can be used to model the conditional distribution $P(s_t, s_{t-1}, \ldots, s_1 \mid \mathbf{f}_1, \ldots \mathbf{f}_t)$ (Lee, 2013; Ma and Fosler-Lussier, 2014; Ren et al., 2013). This conditional distribution over the sequence of states can then be marginalised to obtain the belief state estimate for the latest turn.

Both kinds of approaches use features akin to those used by the static classifiers, differing in that the value-specific features $\mathbf{f}^v_{t_i}$ for $t_i = 1 \ldots t$ are all considered when estimating the belief state at time t (rather than just $\mathbf{f}^v_t$, as in the case of static classifiers). As noted by Henderson (2015c), all DST approaches based on these methods proposed in the literature assume discrete features. This means that many of the (hand-crafted) SLU/ASR features must be quantised, exacerbating the DST models' dependence on manual feature engineering.

Delexicalisation-Based DST Models To overcome these issues, Henderson et al. (2013; 2014c; 2014d) have proposed approaches based on deep and later Recurrent Neural Networks (Hochreiter and Schmidhuber, 1997; Hopfield, 1982; Jain and Medsker, 1999). Their models operate over continuous feature vectors representing the current user utterance and previous dialogue history. Such *delexicalisation-based* DST models have two major advantages over past approaches: **1)** they eliminate the need for feature engineering on the DST front; and **2)** they can entirely remove the dependence on (domain-specific) SLU modules.[7]

Initial word-based DST models used non-sequential models (such as deep neural networks) to produce continuous user utterance representations which were then used by sequential DST models to update the belief state. Approaches such as that of Žilka and Jurčíček (2015) and Vodolán et al. (2017) further advance this paradigm by using Long Short-Term Memory (LSTM) networks (Hochreiter and Schmidhuber, 1997) over user utterances to produce continuous user utterance representations which better model word order and long-term dependencies. Another research direction is that proposed by (Liu and Perez, 2017; Perez and Liu, 2017), who frame dialogue state tracking as a question answering problem and use *End-to-End Memory Networks* (Sukhbaatar et al., 2015; Weston et al., 2014) to train DST models with more advanced reasoning capabilities.

[7] Henderson et al. (2014d) refer to their model as the *word-based* dialogue state tracker, as the model relies on finding exact matches of words in the user utterance with the respective ontology values. For notational clarity, this family of models is referred to as *delexicalisation-based* throughout this thesis.

Joint Learning of SLU and DST Another way to categorise prior discriminative approaches is by their reliance (or otherwise) on a separate SLU module for interpreting user utterances.[8] Traditionally, SLU and DST components were separate, with the DST module operating on SLU output to update the belief state. Models based on the word-based DST paradigm integrate SLU and DST into a single component which uses the ASR output to directly update the belief state without an intermediate semantic decoding step. Combining the two components enables joint learning, allowing both components to make use of the information expressed in previous dialogue turns. The next chapter presents this state-of-the-art model in more detail. The following sections give an overview of the dialogue management and natural language generation components.

Relation to Semantic Parsing The tasks of spoken language understanding and dialogue state tracking in many ways resemble *semantic parsing*, the task of converting natural language into computer-executable formal *meaning representations* for domain-specific applications (Kate and Wong, 2010). Well-known semantic parsing tasks include the Air Traffic Information Service (ATIS), Robocup Coach Language (CLang), and Geoquery, a Database Query Application (Chen et al., 2003; Hemphill et al., 1990; Zelle and Mooney, 1996b). Similarly to Dialogue State Tracking models, semantic parsers are typically domain-specific, with the Meaning Representation Language (MRL) designed by application creators. Since semantic parsers provide natural language interfaces for computational systems, one could consider the SLU/DST pipelines of task-based dialogue systems an instance of semantic parsing models. Contrary to most semantic parsers, the role of the language understanding pipeline in dialogue systems is not to produce a full meaning representation of a particular sentence, but rather the full representation of the user's intent, expressed over a number of dialogue turns while interacting with the dialogue system. The dialogue act slot-filling paradigm used in DST resembles the meaning representations of early semantic parsing tasks such as the Air Traffic Information Service. However, the dialogue act formalism is considerably simpler than that of more recent semantic parsing tasks, which rely on the strongly typed syntax of grammar formalisms such as Combinatory Categorical Grammars (CCGs) (Reddy et al., 2016) or dependency structures such as Stanford Dependencies (de Marneffe and Manning, 2008) and Universal Dependencies (Nivre et al., 2016).

[8] The best-performing models in DSTC 2 all used both raw ASR output and the output of (potentially more than one) SLU decoders (Williams, 2014; Williams et al., 2016). This does not mean that those models are immune to the drawbacks identified here for the two model categories; in fact, they share the drawbacks of both.

2.5 Dialogue Management

The role of the *dialogue manager* is to choose an appropriate system response following the latest user utterance. To do this, the dialogue manager uses the dialogue system's internal estimate of the dialogue state (the belief state) to produce the *system action*, which is the dialogue act representation of the dialogue system's response to the user utterance.

The evolution of research on dialogue management and its relation to the DST component resembles the evolution of DST with respect to the SLU module. Historically, dialogue managers kept a single (top) hypothesis for the dialogue state and used it to choose the next system action. These approaches were based on form filling systems (Goddeau et al., 1996), flowcharts (Lucas, 2000; McTear, 2002; Sutton et al., 1996), and logical inference (Larsson and Traum, 2000). A common trait of these approaches is that system behaviour was not learned from dialogue data or from user interactions. Instead, the system designers manually specified the desired system behaviour, which limited the applicability of early systems to complex dialogue domains with many possible system actions.

The next generation of dialogue managers used Markov Decision Processes (MDPs) to allow the dialogue policy to be learned from data using reinforcement learning (Biermann and Long, 1996; Levin and Pieraccini, 1995; Singh et al., 1999). The learned *dialogue policy* $\pi : s \rightarrow a$ maps the current dialogue state hypothesis s to one of the system actions $a \in A$, where A stands for the set of potential system actions.

The MDP-based approaches allow the system to learn from user interactions. However, the learned policy uses only the top dialogue state hypothesis to choose the next system action. This means that its performance does not benefit from modelling the uncertainty introduced by the previous components of the dialogue system pipeline. An early method which incorporated (some of) the uncertainty information was that of Bohus and Rudnicky (2005), who add additional features to represent uncertainty about the top DST hypothesis. Similar to the early DST approaches, this approach is heavily hand-crafted, with supplementary features dependent on specific properties of preceding ASR, SLU and DST modules.

2.5.1 Partially Observable Markov Decision Processes

Over the past decade, the statistical dialogue modelling literature has been increasingly focused on modelling dialogue as a Partially Observable Markov Decision Process (POMDP) (Roy et al., 2000; Thomson and Young, 2010; Williams and Young, 2007; Young et al., 2010). This approach combines statistical approaches to learning and handling uncertainty, making the system more robust to errors in ASR and SLU decoding, as well as allowing the dialogue policy to be optimised using reinforcement learning.

A key distinction of POMDP-based dialogue management models is that the (true) dialogue state s is treated as a hidden variable. Instead of the top DST hypothesis, the estimated distribution over dialogue states $b(s_t)$, i.e., the full *belief state* at turn t, is used to choose the next system action. To learn the policy, a reward must be assigned to each dialogue turn; it is usually assumed to be a function of the current dialogue state and the subsequent system action (written as $r(s_t, a_t)$).[9] Learning the policy involves maximising the *expected cumulative reward*.[10] For a dialogue of T turns, this is given by:

$$E\left(R(s_{1:T}, a_{1:T})\right) = E\left(\sum_{t=1}^{T} r(s_t, a_t)\right) \tag{2.5}$$

Several assumptions about the environment are made to facilitate this (otherwise intractable) computation (Thomson, 2009). An important one is that belief state transitions are assumed to be Markovian (i.e. to depend only on the previous belief state and system action). The policy $\pi(b_t, a_t)$ is considered stochastic, defining the probability of taking action a_t in belief state b_t. The expected future reward can then be expressed as:

$$V^{\pi}(b_t) = \sum_{a} \pi(b_t, a) r(b_t, a) + \sum_{a} \int_{b_{t+1}} \pi(b_t, a) P(b_{t+1} \mid b_t, a) V^{\pi}(b_{t+1}) \tag{2.6}$$

Estimating the Q-function, which represents the the expected future reward if action a is taken in state s, is one of the key problems in reinforcement learning. In a POMDP, the state is not observable, so the state-action Q-function does not suffice to estimate the expected future reward. Instead, the Q-function must map from a belief state and an action to the expected future reward:

$$Q^{\pi}(b_t, a) = r(b_t, a) + \int_{b_{t+1}} P(b_{t+1} \mid b_t, a) V^{\pi}(b_{t+1}) \tag{2.7}$$

Learning to estimate the Q-function usually requires hundreds of thousands of dialogues. One line of work addressing this problem relied on building user simulators which could automatically generate large dialogue corpora (Jurčíček et al., 2011). Alternatively, Gaussian Processes can be used to model the Q-function, allowing online policy learning with as few as several hundred dialogues (Casanueva et al., 2015; Gašić et al., 2013, 2010; Su et al.,

[9] There is a substantial body of work on defining/learning appropriate reward functions, which typically try to reward successful dialogue completion and favour concise dialogues which achieve user goals quickly.

[10] It is important to note that this cumulative reward deviates from the standard *discounted* return, where a discounting factor is used to assign less value to high-reward actions that happen in the distant future. Dialogue management is an *episodic* task, which means it is guaranteed to complete in a finite number of steps. This eliminates the need for using discounting factors to ensure that the conversation eventually terminates.

2016b). Recently, Su et al. (2017) showed that neural network models can be used in place of Gaussian Processes, achieving comparable performance in low-data scenarios.

Neural Networks for Dialogue Management More recently, neural network approaches to dialogue management have gained popularity. For instance, Su et al. (2016a) combine a supervised learning approach for learning the policy from offline dialogues with an online reinforcement learning method for fine-tuning neural network performance in interaction with real users. Williams et al. (2017) extend this line of work, combining supervised and reinforcement learning to train *Hybrid Code Networks* which incorporate external domain-specific knowledge in the form of software and system-action templates.

2.6 Response Generation

This section gives a short overview of the two final components of the modular dialogue system pipeline: the *natural language generator* (NLG), which transforms the system action produced by the dialogue manager into natural language, and the *text-to-speech* (TTS) generator, which creates the audio representation of the NLG output.

Natural Language Generation

The natural language generator (NLG) takes the system acts produced by the dialogue manager and converts them into natural language. For instance, given a system action *inform(*ADDRESS=*Portugal Street 8,* PARKING=*True)*, the NLG could produce output such as *"The address is Portugal Street 8, and parking is available."* The NLG methods used in dialogue systems broadly fall into two categories:

1. Template-based NLG The simplest method for bootstrapping NLG for dialogue is using *hand-crafted templates* for the given dialogue domain (Cheyer and Guzzoni, 2007; Reiter and Dale, 2000; Van Deemter et al., 2005). In a restaurant search domain, one such template could be: POSTCODE=$X \rightarrow$ *The postcode is X*, where X is the postcode of the target entity. A more complex template would be FOOD=*X,* NAME=*Y,* TYPE=$Z \rightarrow$ *Y is a Z serving X food* (e.g., "The Maypole is a pub serving Italian food"). The key weakness of such approaches is that different values sometimes require different contextualisations (e.g., "there *is* one venue which *meets* your criteria" versus "there *are* two *venues* which *meet* your criteria"). This is especially problematic for morphologically rich languages where, for example, the suffixes of words could depend on word gender or declension.

2. Data-Driven NLG Learnable methods for performing NLG in dialogue systems include the supervised learning approaches of Walker et al. (2001) and Stent et al. (2004), the reinforcement learning ones of Rieser and Lemon (2010), and the *factored language models* of Mairesse et al. (2007; 2014). More recently, Wen et al. (2015a; 2016; 2015b) showed how to train statistical language generators based on different deep neural network structures.

Text-to-Speech

The Text-to-Speech (TTS) engine is the final component of the modular dialogue system pipeline, taking the output of the NLG component and inducing its audio representation. Early approaches to this problem were based on *unit selection*, concatenating pre-recorded speech segments into audio output representing the given word sequence (Black and Lenzo, 2001; Clark et al., 2004; Taylor et al., 1998).

The unit selection approach has since been superseded by data-driven methods, which produce more naturally-sounding output by modelling sentence- and dialogue-level context. Similar to automated speech recognition, the first generation of statistical TTS was based on Hidden Markov Models (Tsiakoulis et al., 2014; Zen et al., 2007), with more recent systems moving on to recurrent neural networks (Zen et al., 2014, 2013).

Chapter 3

Word-Based Belief Tracking: The Pros and Cons of Delexicalisation

The first part of this chapter presents current state-of-the-art neural network approaches for belief tracking. These methods are based on *delexicalisation*, a technique which treats all user intents defined by the domain ontology using the same model parameters, relying on little more than exact matching and context to perform belief tracking.

The second part of the chapter shows how this idea can be extended to train multi-domain belief tracking models, as well as to bootstrap models for new dialogue domains. A substantial part of this chapter is based on work published in Mrkšić et al. (2015).

3.1 Word-Based Dialogue State Tracking

Belief tracking models capture users' goals given their utterances. Goals are represented as sets of constraints expressed by *slot-value* mappings such as FOOD=*chinese* or WIFI=*available*. The set of slots S and the set of values V_s for each slot make up the *ontology* for an application domain. The ontology is domain-dependent and expresses the properties of database entities which users can ask about in the given dialogue domain.

The word-based framework for belief tracking was introduced by Henderson et al. (2014c; 2014d). Following each user utterance, this model outputs a distribution over all goal slot-value pairs defined by the ontology. To model dialogue state transitions, the model uses a single-hidden-layer recurrent neural network, maintaining a *memory* vector that stores internal information about dialogue context. The input for each user utterance consists of the ASR hypotheses, the last system action, the current memory vector and the previous belief state. Rather than using a semantic decoder to convert this input into a meaning

representation, the system uses the turn input to extract a large number of word n-gram features (see lexical features in Figure 3.1). These features capture some of the dialogue dynamics but are not ideal for sharing information across different slots and domains.

Delexicalised n-gram features To overcome this problem, the word-based model relies on another set of features, collected by replacing all references to slot names and values with two generic symbols (SLOT and VALUE) and extracting *delexicalised* n-gram features (see Figure 3.1). Lexical n-grams such as [want cheap price] and [want Chinese food] map to the same delexicalised feature, represented by [want VALUE SLOT]. Such features facilitate transfer learning between slots and allow the system to operate on unseen values or even entirely new slots. As an example, [want available internet] would be delexicalised to [want VALUE SLOT] as well, a useful feature even if there is no training data available for the INTERNET slot. The delexicalised model learns the belief state update corresponding to this feature from its occurrences across the other slots and domains. Subsequently, it can apply the learned behaviour to unseen slots, or even to entirely new domains.

RNN-based Belief State Update The system maintains a separate belief state for each slot s, represented by the distribution $\mathbf{p}_s$ over all possible slot values $v_i^s \in V_s$. The model input at turn t, $\mathbf{x}^t$, consists of the previous belief state $\mathbf{p}_s^{t-1}$, the previous memory state $\mathbf{m}^{t-1}$. The *one-hot* feature vectors $\mathbf{f}_l$ and $\mathbf{f}_{d,v_i^s}$ represent the lexical and value-specific delexicalised features extracted from the turn input (see Figure 3.1).[1] The belief state of each slot s is updated for each of its slot values $v_i^s \in V_s$. The updates are as follows:

$$\mathbf{x}_{v_i^s}^t = \mathbf{f}_l^t \oplus \mathbf{f}_{d,v_i^s}^t \oplus \mathbf{m}^{t-1} \oplus p_{v_i^s}^{t-1} \oplus p_{\emptyset}^{t-1} \tag{3.1}$$

$$g_{v_i^s}^t = \mathbf{w}_1^s \cdot \sigma\left(\mathbf{W}_0^s \mathbf{x}_{v_i^s}^t + b_0^s\right) + b_1^s \tag{3.2}$$

$$p_{v_i^s}^t = \frac{\exp(g_{v_i^s}^t)}{\exp(g_{\emptyset}^t) + \sum_{v' \in V_s} \exp(g_{v'}^t)} \tag{3.3}$$

$$\mathbf{m}^t = \sigma\left(\mathbf{W}_{m_0}^s \mathbf{x}_t + \mathbf{W}_{m_1}^s \mathbf{m}^{t-1}\right) \tag{3.4}$$

where $\oplus$ denotes vector concatenation, $p_{v_i^s}^{t-1}$ the previous probability for the given value and $p_{\emptyset}^t$ the probability that the user expressed no constraint up to turn t. Matrices $\mathbf{W}_0^s$, $\mathbf{W}_{m_0}^s$, $\mathbf{W}_{m_1}^s$ and the vector $\mathbf{w}_1^s$ are the (slot-specific) RNN weights, and b_0^s and b_1^s are the hidden and output layer RNN bias terms. For training, the model is unrolled across turns and trained using backpropagation through time and stochastic gradient descent (Graves, 2012).

[1] Henderson's work distinguished between three types of features: delexicalised features $\mathbf{f_s}$ and $\mathbf{f_{v_i^s}}$ are here subsumed by the delexicalised feature vector $\mathbf{f}_{d,v_i^s}$. The turn input $\mathbf{f}$ corresponds to the lexical feature vector $\mathbf{f}_l$.

		Score / Weight
Transcription	Best Serbian restaurant downtown?	-
ASR Hypothesis 1	Best Serbian restaurant downtown?	0.84
ASR Hypothesis 2	Best Siberian restaurant downtown?	0.16
$\mathbf{f}_l$: Lexical n-gram features		
Lexical Unigrams	Best	1.0
	restaurant	1.0
	downtown	1.0
	Serbian	0.84
	Siberian	0.16
Lexical Bigrams	restaurant downtown	1.0
	Best Serbian	0.84
	Serbian restaurant	0.84
	Best Siberian	0.16
	Siberian restaurant	0.16
Lexical Trigrams	Best Serbian restaurant	0.84
	Serbian restaurant downtown	0.84
	Best Siberian restaurant	0.16
	Siberian restaurant downtown	0.16
$\mathbf{f}_{d,v}$: Delexicalised n-gram features		
for $v =$ Serbian	VALUE	0.84
	Best VALUE	0.84
	VALUE restaurant	0.84
	Best VALUE restaurant	0.84
	VALUE restaurant downtown	0.84
for $v =$ Siberian	VALUE	0.16
	Best VALUE	0.16
	VALUE restaurant	0.16
	Best VALUE restaurant	0.16
	VALUE restaurant downtown	0.16
for v $\neq$ Serbian, Siberian	<EMPTY LIST>	-

Figure 3.1 An illustration of feature extraction performed by the word-based belief tracking model. In this example, an ASR N-best list of length 2 is used to extract lexical and delexicalised n-gram features for $n = 1,2,3$. Delexicalised feature vectors are value-specific and different for each slot value that appears in the ASR hypotheses. In this example, two food values are featured: Serbian and Siberian (often confused both by ASR systems and those lacking geographic knowledge). In this case, the same delexicalised features are active for both values. However, they are assigned different weights by the ASR system, which the word-based belief tracking model can use to choose the more likely value.

Learning Value-Specific Parameters The original RNN architecture proposed by Henderson et al. (2014d) used a second neural network component to learn mappings from lexical n-grams to specific slot values. This means that a separate set of weights $\mathbf{w_1}^{s,v_i^s}$ and b_1^{s,v_i^s} is learned for each slot value $v_i^s \in V_s$, allowing the model to capture specific rephrasings which occur in the training dataset (e.g., that the bigram *moderately priced* is a strong indicator for PRICE=MODERATE). Including these parameters transforms equation 3.2:

$$g^t_{v_i^s} = \mathbf{w}_1^s \cdot \sigma\left(\mathbf{W}_0^s \mathbf{x}^t_{v_i^s} + b_0^s\right) + b_1^s + \mathbf{w}_1^{s,v_i^s} \cdot \sigma\left(\mathbf{W}_0^s \mathbf{x}^t_{v_i^s} + b_0^s\right) + b_1^{s,v_i^s} \tag{3.5}$$

This modelling step turns the core delexicalisation-based model into a hybrid model which can learn value-specific rephrasings. However, it also leads to a very substantial increase in the number of model parameters (proportional to the number of slot values).[2] In their further work on applying pre-trained models to an extended version of the original dialogue domain, Henderson et al. (2014c) do not use this part of the network.

3.2 Beyond Domain-Specific Dialogue State Tracking

Modern dialogue systems are typically designed with a well-defined domain in mind, e.g., restaurant search, travel reservations or shopping for a new laptop. The goal of building open-domain dialogue systems capable of conversing about any topic remains far off.

Domain Adaptation in Dialogue It is well-known in machine learning that a system trained on data from one domain may not perform as well when deployed in a different domain. Researchers have investigated methods for mitigating this problem, with NLP applications in parsing (McClosky et al., 2006, 2010), sentiment analysis (Blitzer et al., 2007; Glorot et al., 2011) and many other tasks. There has been a small amount of previous work on domain adaptation for dialogue systems. Tur et al. (2007) and Margolis et al. (2010) investigated domain adaptation for dialogue act tagging. Walker et al. (2007) trained a sentence planner/generator that adapts to different individuals and domains. Dialogue State Tracking Challenge 3 (Henderson et al., 2014b) explored the ability of models to adapt to an extended version of the dialogue domain they were trained on. Models trained on data from DSTC 2 (Cambridge Restaurants) were evaluated on the tourist information domain, which contains information about restaurants, pubs, hotels and coffee shops.

[2] Interestingly, not including this component leads to a relatively small decrease in DSTC2 performance: the joint goal accuracy falls from 76.8 $\rightarrow$ 74.4. This shows that the core delexicalisation-based network accounts for most of the word-based belief tracking model's performance.

Towards Open-Domain Dialogue As the next step towards the goal of open-domain dialogue, this chapter shows how to build *dialogue state tracking models* which can operate across entirely different domains. As explained in previous sections, word-based belief tracking models powered by recurrent neural networks are well suited to dialogue state tracking, since their ability to capture contextual information allows them to model and label complex dynamic sequences. The word-based RNN models have shown competitive performance in both DSTC 2 and DSTC 3 (Henderson et al., 2014c,d). This approach is particularly well suited to the goal of building open-domain dialogue systems, as it does not require hand-crafted domain-specific resources for semantic interpretation.[3]

3.3 Multi-Domain Dialogue State Tracking

A method for training multi-domain RNN dialogue state tracking models is proposed next. A hierarchical training procedure first uses all the data available to train a *general* belief tracking model. This model learns the most frequent and general dialogue features present across the various domains. The general model is then specialised for each domain, learning domain-specific behaviour while retaining the cross-domain dialogue patterns learned during the initial training stages. These models show robust performance across all the domains investigated, typically outperforming trackers trained on target-domain data alone. The procedure can also be used to initialise dialogue systems for entirely new domains. The subsequent evaluation shows that multi-domain initialisation always improves performance, regardless of the amount of the in-domain training data available. To the best of my knowledge, this work was the first to address the question of multi-domain belief tracking.

3.3.1 Hierarchical Model Training

Delexicalised features allow transfer learning between different slot values and between entire slots. This approach can be extended to achieve transfer learning between entire domains: a model trained to talk about hotels should have some success talking about restaurants, or even laptops. If features learned across disparate domains can be incorporated into a single model, this model should be able to track belief state across all of these domains.[4]

Shared Initialisation The training procedure starts by performing *shared initialisation*: the training datasets for all dialogue domains considered are combined into a single multi-

[3]Note that this does not mean that these models can fully emulate the capabilities of models which do use domain-specific semantic decoders. These limitations will be the focus of study of subsequent chapters.

[4]To move towards domain-independence, the *value-specific* part of the word-based RNN is not used.

domain training dataset. Next, all slot value occurrences for all slots across the different domains are replaced with a generic VALUE tag.[5] The word-based RNN parameters for all slots (across all domains) are tied, and the slot-agnostic delexicalised dialogues are used to learn the parameters of a single *shared RNN model*.

Slot Specialisation The shared RNN model is trained with the purpose of extracting a very rich set of lexical and delexicalised features which capture general dialogue dynamics and keyword contexts potentially applicable across multiple domains. While these features are general, the RNN parameters are not, since not all of the features are equally relevant for different slots and/or domains. For example, [eat VALUE food] and [near VALUE] are clearly features related to FOOD and AREA slots respectively. To ensure that the model learns the relative importance of different features for each of the slots, slot specific models are trained for each slot across all the available domains. To train *slot-specialised* models, the shared RNN's parameters are replicated for each slot and specialised further by performing additional runs of stochastic gradient descent using only slot-specific training dialogues (i.e., those dialogues where at least one user utterance contains a mention of the given slot).

3.3.2 Experimental Setup

The experimental setup follows that of the Dialogue State Tracking Challenges 2 and 3. The key metric used to measure the success of belief tracking models is *goal accuracy*, which represents the ability of the system to correctly infer user goals expressed via slot-value constraints. The evaluation reports the *joint goal accuracy*, which is the marginal test accuracy across all slots. This measure represents the proportion of dialogue turns in the test set where all user constraints expressed up to that dialogue turn were decoded correctly.

Datasets The evaluation considers six dialogue domains, varying across topic and geographical location (Table 3.1). The Cambridge Restaurants data is the data from DSTC 2. The San Francisco Restaurants and San Francisco Hotels datasets were collected during the Parlance project (Gašić et al., 2014). The Tourist Information domain is the DSTC3 dataset: it contains dialogues about hotels, restaurants, pubs and coffee shops. The Michigan Restaurants and Laptops datasets are collections of dialogues sourced using Amazon Mechanical Turk. The Laptops domain contains conversations with users instructed to find laptops with certain characteristics. This domain is substantially different from the other ones, making it particularly useful for assessing the quality of the multi-domain models trained.

[5] Similarly, all mentions of slot names (across all domains) are replaced with a single SLOT name tag.

Dataset / Model	Domain	Train	Test	Slots	Size of Slots (Values per Slot)
Cambridge Restaurants	Restaurants	2118	1117	4	[112, 92, 6, 4]
SF Restaurants	Restaurants	1608	176	7	[240, 155, 60, 40, 5, 4, 3]
Michigan Restaurants	Restaurants	845	146	12	[727, 124, 6, 5, 4, 3, 3, 3, 3, 3, 3, 2]
All Restaurants	Restaurants	4398	-	23	Union of Datasets Above
Tourist Information	Tourist Info	2039	225	9	[164, 53, 29, 15, 5, 4, 3, 3, 3]
SF Hotels	Hotels Info	1086	120	7	[183, 156, 28, 5, 3, 3 , 3]
R+T+H Model	Mixed	7523	-	39	Union of Datasets Above
Laptops	Laptops	900	100	6	[5, 4, 4, 4, 4, 3]
R+T+H+L Model	Mixed	8423	-	45	Union of Datasets Above

Table 3.1 Datasets used in the experiments presented in this chapter. Where applicable, slot value counts include the special *dontcare* value.

Three combined datasets are introduced (see Table 3.1) and used to train increasingly general belief tracking models:

1. **All Restaurants**: trained using combined dialogues from three restaurant domains;
2. **R+T+H**: trained on all dialogues related to restaurants, hotels, pubs and coffee shops (five datasets; Cambridge, Michigan and SF restaurants, SF Hotels and DSTC 3);
3. **R+T+H+L** model: the most general model, trained using all the available dialogue data (six combined datasets; **R+T+H** plus the laptops domain).

Model Training and Hyperparameters Following Henderson et al. (2014c,d), stochastic gradient descent (SGD) is used to train the Recurrent Neural Networks, with the (constant) learning rate set to 0.1. The number of delexicalised features (i.e., the size of the $\mathbf{f}_{d,v_i^s}$ vector in equations 3.1-3.4) is set to 50, and the number of lexical features (i.e., the size of $\mathbf{f}_l$ in equations 3.1-3.4) is set to 4000. Surprisingly, increasing the number of delexicalised features above 50 did not result in improved performance, despite the fact that these features are the main driver of delexicalisation-based model's performance. This shows that the presence of the slot value in an utterance is the main signal that the delexicalisation-based model uses to update the belief state (rather than the context surrounding the slot value occurrence).

The model was trained for 70 epochs, with the mini-batch size set to 10 examples. L2 regularisation is used to penalise the norms of all RNN weights, with the L2 regularisation constant set to 1.0. Gradient clipping is used to deal with exploding gradients, bounding the gradients to the $[-1.0, 1.0]$ interval. The model was implemented and trained using the Theano deep learning framework (Al-Rfou et al., 2016).

Model / Domain	Cam Rest	SF Rest	Mich Rest	Tourist	SF Hotels	Laptops	Geo. Mean
Cambridge Rest.	75.0	26.2	33.1	48.7	5.5	54.1	31.3
SF Restaurants	66.8	**51.6**	31.5	38.2	17.5	47.4	38.8
Michigan Rest.	57.9	22.3	64.2	32.6	10.2	45.4	32.8
All Restaurants	75.5	49.6	67.4	48.2	19.8	53.7	48.5
Tourist Info.	71.7	27.1	31.5	62.9	10.1	55.7	36.0
SF Hotels	26.2	28.7	27.1	27.9	57.1	25.3	30.6
R+T+H	**76.8**	51.2	**68.7**	**65.0**	**58.8**	48.1	60.7
Laptops	66.9	26.1	32.0	46.2	4.6	74.7	31.0
R+T+H+L	**76.8**	50.8	64.4	63.6	57.8	**76.7**	**64.3**

Table 3.2 Goal accuracy of the six domain-specific and the three general shared models trained using different subsets of dialogue domains. All figures show the performance of ensembles of 12 models which differ only in the random initialisations of model parameters. As the geometric mean represents an agglomerative measure of model performance across six large test sets, additional statistical significance tests are not performed.

3.3.3 Evaluation

To evaluate the shared multi-domain models trained, three combinations of the six dialogue domains are used to train increasingly general belief tracking models. The domain-specific models trained using only data from each of the six dialogue domains provide the baseline performance for the three general models. Subsequently, the general models are *slot-specialised* to maximise their performance for each of the individual domains. In the last part of the evaluation, the general shared models are used to initialise high-performance belief tracking models for entirely new domains given a limited number of in-domain training dialogues.

Training General Models Training the shared RNN models is the first step of the training procedure. Table 3.2 shows the performance of shared models trained using dialogues from the six individual and the three combined domains. The joint accuracies are not comparable between the domains as each of them contains a different number of slots. The *geometric mean* (sixth root of the product of the six joint goal accuracies) is calculated to determine how well these models operate across the different dialogue domains.

The parameters of the three multi-domain models are not slot- or even domain-specific. Nonetheless, all of them improve over the domain-specific model for all but one of their constituent domains. The R+T+H model outperforms the R+T+H+L model across four domains, showing that the use of laptops-related dialogues decreases performance slightly across other more closely related domains. However, the latter model is much better at balancing its performance across all six domains, achieving the highest geometric mean and still improving over all but one of the domain-specific models.

Model	Cambridge Restaurants		SF Restaurants		Michigan Restaurants	
	Shared Model	Slot-specialised	Shared Model	Slot-specialised	Shared Model	Slot-specialised
Domain Specific	75.0	75.4	51.6	**56.5**	64.2	65.6
All Restaurants	75.5	77.3	49.6	53.6	67.4	65.9
R+T+H	76.8	**77.4**	51.2	54.6	**68.7**	65.8
R+T+H+L	76.8	77.0	50.8	54.1	64.4	66.9
	Tourist Information		**SF Hotels**		**Laptops**	
	Shared Model	Slot-specialised	Shared Model	Slot-specialised	Shared Model	Slot-specialised
Domain Specific	62.9	65.1	57.1	57.4	74.7	78.4
R+T+H	65.0	**67.1**	58.8	60.7	-	-
R+T+H+L	63.6	65.5	57.8	**61.6**	76.7	**78.9**

Table 3.3 Impact of slot specialisation on performance across the six dialogue domains.

Slot Specialising the General Model

Specialising the shared model for individual slots allows the training procedure to learn the relative importance of different delexicalised features for each slot in a given domain. Table 3.3 shows the effect of slot-specialising the domain-specific and the three shared models across the six dialogue domains. Moving down in these tables corresponds to adding more out-of-domain training data and moving right corresponds to slot-specialising the shared model for each slot in the current domain.

Slot specialisation improved performance in the vast majority of the experiments. All three slot-specialised general models outperformed the word-based RNN model's performance reported in DSTC 2.[6]

Out of Domain Initialisation

The hierarchical training procedure can exploit the available out-of-domain dialogues to initialise improved shared models for new dialogue domains. In these experiments, one of the domains is chosen to act as the *new* domain, while dialogues from the remaining domains serve as *out-of-domain* data. The number of in-domain dialogues available for training is increased at each stage of the experiment and used to train and compare the performance of two slot-specialised models. The two models are slot-specialised versions of two different general models. The first one is trained using only the available in-domain data, while the second one uses all in-domain and out-of-domain training data.

[6]Note that this refers to the 74.6 score achieved by the word-based model without the value-specific part of the network. Since the considered dialogue domains have different slots and/or slot values, this part of the network cannot be generalised to learn value-specific rephrasings across different domains without introducing additional slot-dependent parameters (which runs against the generality of shared models introduced here).

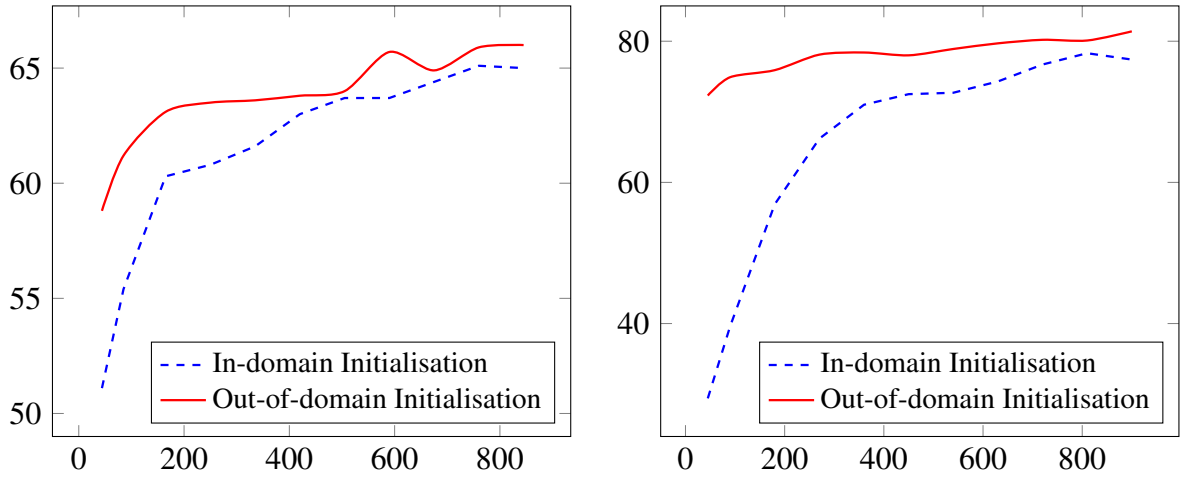


Figure 3.2 Joint goal accuracy on Michigan Restaurants (left) and the Laptops domain (right) as a function of the number of in-domain training dialogues available to the training procedure (ensembles of four models).

Two Unseen Domains The two experiments vary in the degree of similarity between the in-domain and out-of-domain dialogues. In the first experiment, Michigan Restaurants act as the new domain and the remaining R+T+H dialogues are used as out-of-domain data. In the second experiment, Laptop dialogues are the in-domain data and the remaining dialogue domains are used to initialise the more general shared model.

The two plots in Figure 3.2 show how the performance of the two differently initialised models improves as additional in-domain dialogues are introduced. In both experiments, the use of out-of-domain data helps to initialise the model to a much better starting point when the in-domain training dataset is small. The out-of-domain initialisation consistently improves performance: the joint goal accuracy is improved even when the entire in-domain dataset becomes available to the training procedure.

Discussion These results are not surprising in the case of the system trained to talk about Michigan Restaurants. Dialogue systems trained to help users find restaurants or hotels should have no trouble finding restaurants in alternative geographies. In line with these expectations, the use of a shared model initialised using R+T+H dialogues results in a model with strong starting performance. As additional in-domain dialogues are introduced, the model shows relatively minor performance gains over the domain-specific one.

The results of the Laptops experiment are more compelling, as the difference in performance between the differently initialised models becomes larger and more consistent. There are two factors at play here: exposing the training procedure to substantially different

out-of-domain dialogues allows it to learn delexicalised features not present in the in-domain training data. These features are applicable to the Laptops domain, as evidenced by the very strong starting performance. As additional in-domain dialogues are introduced, the delexicalised features not present in the out-of-domain data are learned as well, leading to consistent improvements in belief tracking performance.

Leveraging Out-of-Domain Dialogue Data In the context of these results, it is clear that the out-of-domain training data can be even more beneficial to tracking performance than data from relatively similar domains, especially if the model is to be applied to previously unseen domains. This is especially the case when the available in-domain training datasets are too small to allow the procedure to extract the appropriate delexicalised features.

3.4 Limitations of Delexicalisation-Based Models

This chapter has shown that delexicalisation-based models offer an effective and inexpensive way to bootstrap DST models for new dialogue domains with very limited training data. However, these models rely on exact string matching to detect slot-value mentions in user utterances. In practice, this means that human effort is required to handle the different ways in which a user may articulate her intent at test time. Consequently, the previous role of SLU modules is relegated to system designers, who must now account for linguistic variation by manually specifying potential rephrasings in the form of *semantic dictionaries* (the term *semantic lexicon* is used interchangeably in the literature, and in this thesis).

Domain-Specific Semantic Dictionaries Most domains considered in spoken dialogue system research are relatively small, which means system designers can typically hand-craft semantic dictionaries without much effort. Figure 3.3 shows a subset of the dictionary constructed for the Laptops domain (Vandyke et al., 2015). These dictionaries are not only domain-specific; they often cater to the specific dataset/environment that the researcher focuses on. For example, the lexicon shown in Figure 3.3 defines 21 potential rephrasings for PRICE RANGE=*moderate*, but none for DRIVERANGE=*medium* and only two for WEIGHTRANGE=*medium*. This provides insight into the system development process: the users and the system (i.e. the *policy* learned by the dialogue manager) spent far more time talking about price than about other laptop characteristics. Subsequently, the system designers focused on handling all possible ways in which the user can ask for a reasonably priced laptop.

```
INFORMABLE SLOTS: {
    PRICE RANGE: [
        BUDGET:     inexpensive, cheap, budget
        MODERATE:   average priced, mid priced, reasonable range, moderate price,
                    mid range, mid price range, average range, reasonable priced,
                    medium price range, reasonable price range, medium range,
                    moderately priced, medium price, moderate price range,
                    medium priced, moderately, moderate range, mid price,
                    average price, reasonable price, reasonably priced
        EXPENSIVE:  costly, expensive, dear
    ],
    WEIGHTRANGE: [
        HEAVY: heavy
        MIDWEIGHT: mid, mid weight
        LIGHTWEIGHT: light, light weight
    ],
    DRIVERANGE: [
        LARGE: large
        MEDIUM: medium
        SMALL: small
    ],
    ISFORBUSINESSCOMPUTING: [
        1: for business computing, for business
        0: for play, for gaming, for fun
    ],
    …
}
REQUESTABLE SLOTS: [
    BATTERYRATING: battery range, battery rating, battery capacity
    DRIVE: hard drive, drive
    DIMENSION: how big, dimension, size
    WEIGHTRANGE: weight range, weight
    PRICE: prices, price
    ISFORBUSINESSCOMPUTING: for play, for gaming, for business computing,
                            for fun, for business
    …
]
```

Figure 3.3 A subset of the semantic dictionary for the Laptops domains hand-crafted by Vandyke et al. (2015).

Handling Non-Categorical Slots Figure 3.3 illustrates another important feature of semantic dictionaries: they are often used to encode domain-specific logic for interacting with the application ontology. For instance, the IsForBusinessComputing slot has two values, 1 and 0, which indicate whether laptops are suitable for commercial use. Such *boolean slots* are particularly difficult to handle, since their expression in user utterances often depends on context and does not usually involve explicit verbalisations of the implicit slot value. The user might say: *I'm looking for a hotel with wifi* or *I'd prefer a non-smoking restaurant*.

High-quality semantic dictionaries are essential for robust language understanding in non-trivial dialogue domains. The process of constructing such dictionaries is currently handcrafted and relies on skilled system designers familiar with all components of the dialogue system pipeline. This manual design process becomes untenable as dialogue systems scale to larger and more sophisticated application domains. The next chapter focuses on removing system designers from the loop by constructing semantic dictionaries automatically.

Chapter 4

Inducing Semantic Lexicons for Belief Tracking

This chapter shows how pre-trained word vector space representations, also known as *word embeddings*, can be used to induce semantic lexicons for belief tracking models. The first part of the chapter gives an overview of unsupervised methods for inducing word vector spaces. The second part of the chapter shows that such general word vector collections can be calibrated to capture semantic similarity. Finally, the semantically specialised word vectors are used to show that semantic dictionaries for arbitrary dialogue domains can be constructed automatically. The chapter concludes with a discussion of the limitations of delexicalisation-based models powered by domain-specific semantic lexicons.

4.1 Inducing Word Vectors from Textual Corpora

Many popular methods that induce vectorial representations for words rely on the *distributional hypothesis*, which assumes that semantically similar or related words appear in similar contexts (Firth, 1957; Harris, 1954). This hypothesis supports unsupervised learning of meaningful word representations from large corpora (Curran, 2003; Mikolov et al., 2013b; Ó Séaghdha and Korhonen, 2014; Pennington et al., 2014). Word vectors trained using these methods have proven useful for many downstream tasks such as Part-of-Speech (POS) tagging (Collobert et al., 2011), machine translation (Devlin et al., 2014; Zou et al., 2013), dependency and semantic parsing (Ammar et al., 2016; Bansal et al., 2014; Chen and Manning, 2014; Johannsen et al., 2015; Socher et al., 2013a), sentiment analysis (Socher et al., 2013b), named entity recognition (Guo et al., 2014; Turian et al., 2010), and many others. An overview of existing approaches for inducing word embeddings is presented next.

4.1.1 Historical Approaches for Inducing Word Representations

Prior methods for learning word vector space representations can be split into three categories. These correspond to the different research communities they originated from: **1)** information retrieval; **2)** computational linguistics; and **3)** artificial neural networks (deep learning).

1. Approaches Developed for Information Retrieval Vector space models for words (and documents) have been widely used in NLP for the last thirty years. One of the first approaches, proposed in the context of information retrieval, was that based on Latent Semantic Analysis / Indexing (LSA/LSI) (Deerwester et al., 1990; Landauer et al., 1998). These methods start from the distributional hypothesis, assuming that similar words will appear in similar documents. They use a simple mathematical representation of documents: a term-document matrix, where rows represent different words and columns contain the term (word) counts per each document. Subsequently, Singular Value Decomposition (SVD) is used to construct a *low-rank* approximation of this matrix (the number of columns stays the same and the 'dimension' of document representations is reduced). There are several reasons for computing the low-rank approximation: the original term-document matrix may be intractably large, overly noisy (some words may rarely feature in certain kinds of documents), or overly sparse (words related to particular documents may not have occurred in any of the documents provided). Using the low-rank approximations of the bag-of-words document representation helps alleviate these issues, since the compression step retains mostly the meaningful information. Subsequently, the cosine products between the words' and/or documents' lower-dimensional representations can be used to discern how similar these are to one another. These models were the precursor to more complicated *topic models*, including the widely used Latent Dirichlet Allocation (LDA) model (Blei et al., 2003).

2. Count-based Distributional Semantic Models In computational linguistics, researchers focused on models based on word co-occurrences, encoding the core assumption of the distributional hypothesis (*'You shall know a word by the company it keeps'*, (Firth, 1957)). Unlike models used in information retrieval, *distributional semantic models* (DSMs) do not model the relation between individual words and the documents these words feature in. Instead, they derive a high-dimensional distributed representation of a word by looking at other words that the given word co-occurs with. Given a textual corpora, DSMs operate over co-occurrence matrices, which count how many times any pair of words appears close together. The 'neighbourhood' of a word is usually defined as the set of all words that appear

at least k words away from the given word.[1] The (square) co-occurrence matrices are usually very large, as their size grows with the size of the word vocabulary. DSM models produce lower-dimensional transformations of these matrices, where each row becomes a (more) compact representation of a word's distribution across different contexts.

Early examples of distributional semantic models include the Word Space (Schütze, 1993) and the Hyperspace Analogue to Language (HAL) models (Lund and Burgess, 1996). In the Word Space model, large-scale linear regression is used to induce semantic representations of words that are subsequently used to perform word sense disambiguation. The HAL model (known as the *semantic memory* model in cognitive science), uses a structured context representation, distinguishing between words that appear before or after the target word. The model takes the distance between word pairs into account: the further away two words are, the weaker their association in the co-occurrence matrix is. The HAL model subsequently inspired more sophisticated DSM models, including random indexing (Kanerva et al., 2000), the BEAGLE model (Jones and Mewhort, 2007), and many others.

3. Neural Word Embeddings At roughly the same time, neural network research started using contextual information to learn representations of words as well. Similar to DSM models, neural methods use surrounding words, rather than entire documents, to infer the words' distributed representations. Early examples of these methods include Self-Organising Maps (Kohonen et al., 2001) and Recurrent Neural Networks, which later evolved into *Neural Language Models* (Bengio et al., 2003). This line of work coined the term *word embedding*, for distributed representations of words which were learned jointly with the parameters of the language model itself. Collobert and Weston (2008) demonstrate the usefulness of pre-trained word vectors in a multi-task setting, using a single Convolutional Neural Network to perform predictions across several NLP tasks (POS tagging, chunking, named entity recognition, semantic role labelling, etc.). Five years later, Mikolov et al. (2013b) proposed WORD2VEC, a neural net toolkit which facilitated rapid training and deployment of word embeddings, cementing their role as one of the main pillars of modern NLP research.

Methods such as neural language models (Bengio et al., 2003) compute word embeddings in the first layer of the neural network. The word embeddings of the previous n words are concatenated, and the subsequent network layers (learn to) model the word sequence, with the final softmax layer predicting the next word in the sequence. The softmax layer (which normalises the probability distribution) is the main obstacle to scaling this model, as its complexity grows with the size of the vocabulary (which can include millions of

[1] The best / right size for the context window has been the subject of much research. Standard values in the literature range from 2 to 20 words left and right of the target word.

words). Moreover, the intermediate fully-connected hidden layers are also computationally expensive: the number of their parameters grows quadratically with the size of the word context considered, hampering the ability of these models to handle long-term dependencies.[2]

4.1.2 The Word2Vec Model

WORD2VEC is a comparatively simple neural network architecture for inducing word embeddings. It foregoes the use of large, computationally expensive hidden layers, allowing the model to take wider word context into account. In the seminal paper introducing this model, Mikolov et al. (2013b) propose two different learning strategies:

1. Continuous Bag-of-Words (CBOW) Language models use the preceding n words $w_{k-n},\ldots,w_{k-1}$ to estimate the probability distribution over the next word, w_k. Namely, the language model estimates $P(w_k = v_i \mid w_{k-n},\ldots,w_{k-1})$ for each word v_i in the vocabulary. To learn non-task-specific embeddings useful across a plethora of tasks, the CBOW method instead learns to predict the word w_k given the n words preceding and following the target word. Instead of concatenating the word embeddings of these $2n$ words, the CBOW model sums them together (ignoring word order) and uses the combined embedding to estimate which word is most likely to appear in that (unordered) context. For a sentence of length T, the CBOW cost function can be expressed as:

$$E_\theta = \frac{1}{T}\sum_{t=1}^{T} \log P(w_t \mid w_{t-n},\ldots,w_{t-1},w_{t+1},\ldots,w_{t+n}) \tag{4.1}$$

where θ are the parameters of the neural net predicting the central word, 'out-of-sentence' words are replaced with zero vectors, and, unlike in language modelling, word order does not matter (as the context is represented using the sum of the $2n$ surrounding words' vectors).

2. Skip-Gram with Negative Sampling (SGNS) The Skip-Gram WORD2VEC model moves further away from the language modelling objective. Instead of predicting the central word given its context, the model tries to predict all the context words given the central word. For a sentence of length T, its objective can be expressed as:

$$E_\theta = \frac{1}{T}\sum_{t=1}^{T} \sum_{i\in[-n,\ldots,-1,1,\ldots,n]} \log P(w_{t+i} \mid w_t) \tag{4.2}$$

[2]The multi-task neural network used by Collobert and Weston (2008) for multi-task learning does not involve the softmax computation, but is in a similar way limited to using narrow word context windows.

where θ stands for the parameters of the neural network's projection layer. This model conditions only on the (input) word vector w_t. The final layer of the model is a softmax layer which predicts the probability that each word in the vocabulary is in the input word's context window. This model directly implements the distributional hypothesis: words with similar contexts should produce similar predictions, which means that the learned inputs (the word embeddings themselves) will become very similar.

Negative Sampling This model is trained by feeding pairs of words which may or may not frequently feature in each others' context windows. The model itself has many parameters: $V \times D$, where V is the vocabulary size and D the size of word embeddings. Each training example (which consists of one word pair) updates each of these parameters, leading to very slow training times. For the example pair (*Los*, *Angeles*), the output softmax should predict a 1 for *Angeles* (given *Los* as input), and 0 for most other words in the vocabulary. To avoid very expensive updates, the model chooses a small number of *negative examples* (typically 2-20), chosen at random according to their frequency in the text. The embeddings (i.e., network parameters) for these words are the only ones updated, leading to very fast training times even for very large textual corpora. The Skip-Gram model making use of negative sampling[3] is often referred to as SGNS, and is the most popular method for learning word embeddings from textual corpora in use today.

Global Vectors (GloVe) Another popular method for inducing word embeddings is GloVe (Pennington et al., 2014), who go one step further in trying to decipher word meaning from the co-occurrence statistics in large textual corpora. The authors posit that the *relative ratios* of word co-occurrence probabilities, and not the word co-occurrence probabilities themselves, contain the signal which encodes the true meaning of words. In their evaluation, they show that GloVe outperforms WORD2VEC over a wide array of representative tasks: word analogy, word similarity, and named entity recognition. Moreover, GloVe operates directly over the co-occurrence matrix and not over individual context windows, allowing the model to train substantially faster than WORD2VEC.

Theoretical and Empirical Differences between different Methods There has been much work on discovering which word embedding model yields the best pre-trained word vector collection for use across a wide array of NLP tasks. For instance, Baroni et al. (2014)

[3]Other tricks are very important for achieving robust performance with WORD2VEC. These include subsampling frequent words to avoid overfitting, treating common phrases as single words, smoothing the sampling distribution for negative samples, and many others. For a full overview of these and an analysis of their effect on the embeddings, see (Levy and Goldberg, 2014; Mikolov et al., 2013b).

claim that models which do prediction (i.e., neural network models such as WORD2VEC) outperform traditional count-based models used in distributional semantics. Levy et al. (2015) show there is little theoretical grounding in such claims, and that performance differences between neural and count-based models are mostly due to improved model hyperparameters. For instance, they show that pointwise mutual information (PMI) factorisations of co-occurrence matrices can produce word embeddings of similar quality as WORD2VEC once a similar kind of context distribution smoothing is applied to the PMI matrices.

4.2 The Drawbacks of the Distributional Hypothesis

A major drawback of learning word embeddings from co-occurrence information in textual corpora is that this approach tends to coalesce the notions of *semantic similarity* and *conceptual association* (Hill et al., 2015). Furthermore, even methods that can distinguish similarity from association (e.g., based on syntactic co-occurrences) will generally fail to tell synonyms from antonyms (Mohammad et al., 2008). For example, words such as *east* and *west* or *expensive* and *inexpensive* appear in near-identical contexts, which means that distributional models produce very similar word vectors for such words. Examples of such anomalies in GloVe vectors can be seen in Table 4.1, where words such as *cheaper* and *inexpensive* are deemed similar to (their antonym) *expensive*.

The second obstacle to using distributional embeddings for building language understanding models for task-oriented dialogue systems is that similarity and antonymy can be application- or domain-specific. For example, a DST module for the restaurant search domain needs to detect whether the user wants a *cheap* or an *expensive* restaurant. Being able to distinguish between semantically different, yet conceptually related words (e.g., *cheaper* and *pricey*) is critical for the performance of the dialogue system. In particular, a statistical dialogue system can be led seriously astray by false synonyms.

Past the Distributional Hypothesis The remaining part of this chapter presents a method that addresses these two drawbacks by fine-tuning distributional word vectors using synonymy and antonymy relations drawn from either (or both): **1)** general lexical resources; or **2)** domain-specific dialogue ontologies. The method, termed *counter-fitting*, is a lightweight post-processing procedure in the spirit of *retrofitting* (Faruqui et al., 2015). The second row of Table 4.1 illustrates the results of counter-fitting: the nearest neighbours capture true similarity much more intuitively than the original GloVe vectors.

	east	expensive	British
Before	west	pricey	American
	north	cheaper	Australian
	south	costly	Britain
	southeast	overpriced	European
	northeast	inexpensive	England
After	eastward	costly	Brits
	eastern	pricy	London
	easterly	overpriced	BBC
	-	pricey	UK
	-	afford	Britain

Table 4.1 The effects of counter-fitting: nearest neighbours for three target words using GloVe word vectors before and after counter-fitting.

Counter-fitting improves word vector quality regardless of the initial word vectors provided as input.[4] In fact, applying counter-fitting to Paragram word vectors (Wieting et al., 2015) achieves the new state-of-the-art performance on SimLex-999, a dataset designed to measure how well different models judge semantic similarity between words (Hill et al., 2015). As will be shown in the latter part of this chapter, counter-fitting can also inject knowledge of dialogue domain ontologies into word vector space representations. The modified vector spaces can then be used to automatically construct semantic dictionaries, improving DST performance across two different dialogue domains.[5]

4.2.1 Related Work

Most work on improving word vector representations using lexical resources has focused on bringing words which are known to be semantically related closer together in the vector space. Some methods modify the prior or the regularization of the original training procedure (Bian et al., 2014; Kiela et al., 2015; Yu and Dredze, 2014). Wieting et al. (2015) use the Paraphrase Database (Ganitkevitch et al., 2013) to train word vectors which emphasise word similarity over word relatedness. Recently, there has been interest in lightweight post-processing procedures that use lexical knowledge to refine off-the-shelf word vectors without requiring large corpora for (re-)training as the aforementioned "heavyweight" procedures do. Faruqui et al.'s (2015) *retrofitting* approach uses similarity constraints from WordNet and other resources to pull similar words closer together.

[4] In this context, "improving" refers to improving the vector space for a specific purpose. There is no reason to expect that a vector space fine-tuned for semantic similarity will give better results on semantic relatedness. As Mohammad et al. (2008) observe, antonymous concepts are related but not similar.

[5] The tool and the produced word vectors are available at: *github.com/nmrksic/counter-fitting*.

The complications caused by antonymy for distributional methods are well-known in the semantics community. Most prior work focuses on extracting antonym pairs from text rather than exploiting them (Hashimoto et al., 2012; Lin et al., 2003; Mohammad et al., 2008, 2013; Turney, 2008). The most common use of antonymy information is to provide features for systems that detect contradictions or logical entailment (de Marneffe et al., 2008; Marcu and Echihabi, 2002; Zanzotto et al., 2009). To the best of my knowledge, there is no previous work on exploiting antonymy in dialogue systems.

Past approaches closest in spirit to counter-fitting are: **1)** Liu et al. (2015), who use antonymy and WordNet hierarchy information to modify the WORD2VEC training objective; **2)** Yih et al. (2012), who use a Siamese neural network to improve the quality of vectors produced using Latent Semantic Analysis; **3)** Schwartz et al. (2015), who build a standard distributional model from co-occurrences based on *symmetric patterns*, with specified antonymy patterns counted as negative co-occurrences; and **4)** Ono et al. (2015), who use thesauri and distributional data to train word embeddings specialised for capturing antonymy.

4.3 Counter-fitting Word Vectors to Linguistic Constraints

The starting point of the procedure is an indexed set of word vectors $V = \{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N\}$, with one vector for each word in the vocabulary. Counter-fitting injects semantic relations into this vector space to produce new word vectors $V' = \{\mathbf{v}'_1, \mathbf{v}'_2, \ldots, \mathbf{v}'_N\}$.

As input, the counter-fitting procedure takes two sets of *linguistic constraints*, S and A. These sets consist of example word pairs (i.e., constraints) that stand in synonymy and antonymy relations, respectively. The elements of each set are pairs of word indices; for example, each pair (i, j) in S is such that the i-th and j-th words in the vocabulary are synonyms. The objective function used to counter-fit the pre-trained word vectors V to the sets of linguistic constraints A and S contains three different terms:

1. Antonym Repel (AR): This term serves to *push* antonymous words' vectors away from each other in the transformed vector space V':

$$\text{AR}(V') = \sum_{(u,w)\in A} \tau\left(\delta - d(\mathbf{v}'_u, \mathbf{v}'_w)\right)$$

where $d(\mathbf{v}_i, \mathbf{v}_j) = 1 - \cos(\mathbf{v}_i, \mathbf{v}_j)$ is a distance derived from cosine similarity and $\tau(x) \triangleq max(0, x)$ imposes a margin on the cost. Intuitively, δ is the "ideal" minimum distance between antonymous words. In the experiments presented in this chapter, δ is set to 1.0, which corresponds to imposing vector orthogonality.

2. Synonym Attract (SA): The counter-fitting procedure seeks to bring word vectors of known synonymous word pairs closer together:

$$\mathrm{SA}(V') = \sum_{(u,w)\in S} \tau\left(d(\mathbf{v}'_u, \mathbf{v}'_w) - \gamma\right)$$

where γ is the "ideal" maximum distance between synonymous words ($\gamma = 0$ is used in this chapter; the procedure tries to make the representations of synonyms as similar as possible).

3. Vector Space Preservation (VSP): The topology of the original vector space describes relationships between words in the vocabulary captured using distributional information from very large textual corpora. The VSP term bends the transformed vector space towards the original one as much as possible in order to preserve the useful semantic information contained in the original vectors:

$$\mathrm{VSP}(V, V') = \sum_{i=1}^{N} \sum_{j\in N(i)} \tau\left(d(\mathbf{v}'_i, \mathbf{v}'_j) - d(\mathbf{v}_i, \mathbf{v}_j)\right)$$

For computational efficiency, the distances between every pair of words in the vocabulary are not calculated repeatedly as the procedure transforms the vector space. Instead, for each word v_i, the VSP term is applied to its neighbourhood $N(i)$, which denotes the set of words within a certain radius ρ around the i-th word's vector in the original vector space V. Counter-fitting is relatively insensitive to the choice of ρ, with values between 0.2 and 0.4 showing little difference in performance. The value of $\rho = 0.2$ is used throughout this chapter.[6] The full counter-fitting cost function is given by the weighted sum of the three terms:

$$C(V, V') = k_1\mathrm{AR}(V') + k_2\mathrm{SA}(V') + k_3\mathrm{VSP}(V, V')$$

where $k_1, k_2, k_3 \geq 0$ are hyperparameters that control the relative importance of each term. In all experiments presented in this chapter, these are set to be equal: $k_1 = k_2 = k_3 = 0.1$ (the value of 0.1 results in fastest convergence). To minimise the cost function for a set of starting vectors V and produce counter-fitted vectors V', stochastic gradient descent (SGD) is run for twenty epochs, which suffices for the new word vector estimates to converge.[7]

[6] Since the SimLex-999 dataset only has an evaluation set, the WordSim-353 gold-standard association dataset was used to tune the neighbourhood radius ρ (Finkelstein et al., 2002). The values of synonymy and antonymy margins correspond to enforcing vector similarity or orthogonality and as such were not tuned.

[7] For each synonymy and antonymy constraint, the counter-fitting procedure translates both words' vectors along the gradient of the defined cost function. The norms of word vectors are renormalised after every epoch of SGD. The full implementation of the procedure is available at: *github.com/nmrksic/counterfitting*.

4.3.1 Injecting Dialogue Domain Ontologies into Vector Spaces

Dialogue state tracking (DST) models capture users' goals given their utterances. Goals are represented as sets of constraints expressed by *slot-value* pairs such as FOOD=*Indian* or PARKING=*allowed*. The set of slots S and the set of values V_s for each slot make up the *ontology* of the dialogue domain. As explained in the previous chapter, the recurrent neural network (RNN) framework for dialogue state tracking proposed by Henderson et al. (2014c,d) does not use spoken language understanding (SLU) decoders to convert user utterances into meaning representations. Instead, this model operates directly over the n-gram features extracted from the automated speech recognition (ASR) hypotheses.

A drawback of this approach is that the RNN model can only perform exact string matching to detect the slot names and values mentioned by the user. It cannot recognise synonymous words such as *pricey* and *expensive*, or even subtle morphological variations such as *moderate* and *moderately*. To mitigate this problem, one can use *semantic dictionaries*: lists of rephrasings for the values in the ontology. Manual construction of dictionaries is highly labour-intensive; however, if one could automatically detect high-quality rephrasings, this capability would come at no extra cost to the system designer.

To obtain a set of word vectors which can be used for creating a semantic dictionary, the domain ontology must be *injected* into the vector space. This can be achieved by introducing antonymy constraints between all the possible values of each slot (i.e., *Chinese* and *Indian*, *Chinese* and *Thai*, *expensive* and *cheap*, etc.). The remaining linguistic constraints can come from semantic lexicons: the richer the sets of injected synonyms and antonyms are, the better the resulting word representations should become.

4.4 Experiments

4.4.1 Word Vectors and Semantic Lexicons

Initial Word Vectors Two different collections of pre-trained word vectors are used as input to the counter-fitting procedure:

1. **GloVe Common Crawl** 300-dimensional vectors made available by Pennington et al. (2014), and trained on a very large corpus consisting of 840 billion tokens. The word vectors were induced using the GloVe procedure, presented earlier in this chapter.

2. **Paragram-SL999** 300-dimensional vectors made available by Wieting et al. (2015). These vectors are specialised for semantic similarity using synonymous (but not antonymous) word pairs to fine-tune the initial (distributional) GloVe word vectors.

Source of Linguistic Constraints The sets of synonymy and antonymy constraints S and A were obtained from two well-known semantic lexicons:

1. **PPDB 2.0 (Pavlick et al., 2015):** the second release of the Paraphrase Database (PPDB). A new feature of this version is that it assigns relation types to its word pairs. The *Equivalence* relation is used extract synonyms, while the *Exclusion* constraint provides antonyms. The largest available (XXXL) version of the PPDB database is used in all experiments, and only single-token terms are considered.

2. **WordNet (Miller, 1995)**: WordNet (WN) is a well known semantic lexicon which contains vast amounts of high quality human-annotated synonym and antonym pairs. Any two words in the vocabulary which had antonymous word senses were considered antonyms; WordNet synonyms were not used.

In total, the two lexicons yielded 12,802 antonymy and 31,828 synonymy pairs for the vocabulary, which consisted of 76,427 most frequent words in English OpenSubtitles.[8]

4.4.2 Improving Lexical Similarity Predictions

The experiments in this section investigate whether counter-fitting pre-trained word vectors with linguistic constraints improves their usefulness for judging semantic similarity. The principal metric reported is Spearman's rank correlation with the SimLex-999 dataset (Hill et al., 2015). SimLex contains 999 word pairs ranked by a large number of annotators instructed to consider only semantic similarity (but not relatedness) between words. The annotators (sourced through Amazon Mechanical Turk) were instructed to give each word pair a rating between 0 and 10, corresponding to the similarity of the two words' meaning. Before starting the annotation process, they were shown examples of *synonymous* words (e.g., *cup* and *mug*, *glasses* and *spectacles*), examples of *similar*, but not synonymous words (e.g., *love* and *affection*, *frog* and *toad*), and examples of *related* but non-similar words (e.g., *car* and *crash*, *tyre* and *car*). Each word pair was scored by 50 different annotators. Subsequently, the 999 word pairs were sorted by their average similarity score.

Table 4.2 contains a summary of recently reported competitive scores for SimLex-999, as well as the performance of the unaltered, retrofitted and counter-fitted GloVe and Paragram-SL999 word vectors. To the best of my knowledge, the 0.685 figure reported for the latter was the previous SimLex-999 high score. This figure is above the average inter-annotator agreement of 0.67, which has been referred to as the ceiling performance in most prior work.

[8]The vocabulary was downloaded from: *invokeit.wordpress.com/frequency-word-lists/*

Model / Word Vectors	ρ
Neural MT Model (Hill et al., 2014)	0.52
Symmetric Patterns (Schwartz et al., 2015)	0.56
Non-distributional Vectors (Faruqui and Dyer, 2015)	0.58
GloVe vectors (Pennington et al., 2014)	0.41
GloVe vectors + Retrofitting	0.53
GloVe + Counter-fitting	0.58
Paragram-SL999 (Wieting et al., 2015)	0.69
Paragram-SL999 + Retrofitting	0.68
Paragram-SL999 + Counter-fitting	**0.74**
Inter-annotator agreement	0.67
Annotator/gold standard agreement	0.78

Table 4.2 SimLex-999 performance. Retrofitting uses the code provided by the authors.

However, the average inter-annotator agreement is not the only meaningful measure of ceiling performance. An alternative way to estimate the highest attainable SimLex-999 performance is to compare: **a)** the average rank correlation of the ranking produced by the model and the gold standard ranking to: **b)** the correlation that individual human annotators' rankings achieved with the gold standard ranking. The SimLex-999 authors informed me that the average annotator agreement with the gold standard is 0.78. This figure is now reported as a potentially fairer ceiling performance for SimLex-999 on the dataset's official webpage (*www.cl.cam.ac.uk/ fh295/simlex.html*). As shown in Table 4.2, the reported performance of all the models and word vectors falls well below this figure.

Retrofitting pre-trained word vectors improves GloVe vectors, but not the already semantically specialised Paragram-SL999 vectors. Counter-fitting substantially improves both sets

Semantic Resource	Glove	Paragram
Baseline (no linguistic constraints)	0.41	0.69
PPDB− (PPDB antonyms)	0.43	0.69
PPDB+ (PPDB synonyms)	0.46	0.68
WordNet− (WordNet antonyms)	0.52	0.74
PPDB− and PPDB+	0.50	0.69
WordNet− and PPDB−	0.53	0.74
WordNet− and PPDB+	0.58	0.74
WordNet− and PPDB− and PPDB+	0.58	0.74

Table 4.3 SimLex-999 performance when different sets of linguistic constraints are used for counter-fitting (for two different collections of word vectors).

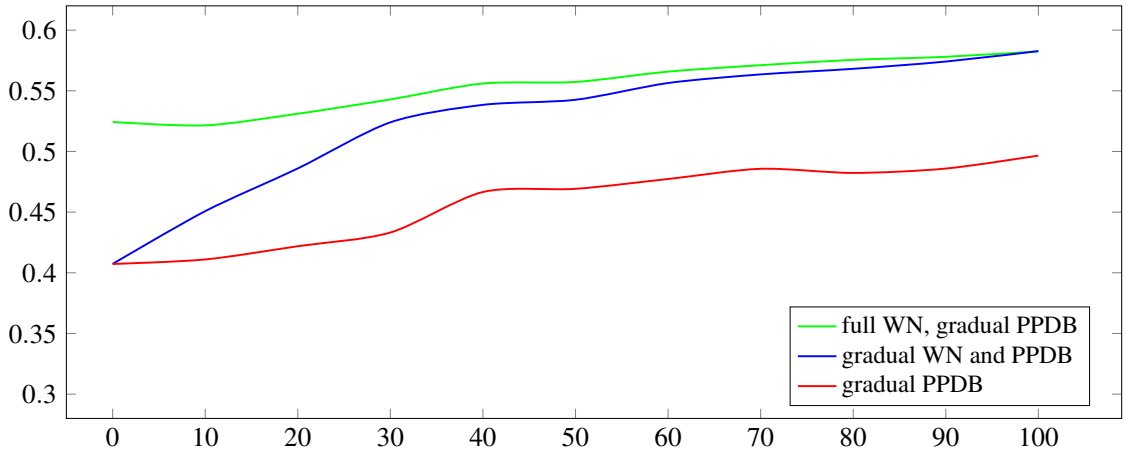


Figure 4.1 SimLex-999 score as a function of the proportion of the linguistic constraints injected into GloVe word vectors.

of vectors, showing that injecting antonymy relations goes a long way towards improving word vectors for the purpose of making semantic similarity judgements.

Table 4.3 shows the effect of injecting different subsets of linguistic constraints. GloVe vectors benefit from all three sets of constraints, whereas the quality of Paragram vectors, already exposed to PPDB, only improves with the injection of WordNet antonyms. Figure 4.1 shows how the SimLex score of GloVe vectors improves with the number of introduced constraints. The effect of adding additional constraints becomes weaker as more constraints are added. However, the improvements do not saturate, with performance steadily improving as additional constraints are injected into the vector space.

Table 4.4 illustrates how incorrect similarity predictions based on the original (Paragram) vectors can be fixed through counter-fitting. The table presents eight false synonyms and

False Synonyms	**Fixed**	**False Antonyms**	**Fixed**
sunset, sunrise	✓	dumb, dense	
forget, ignore		adult, guardian	
girl, maid		polite, proper	✓✓
happiness, luck	✓✓	strength, might	
south, north	✓	water, ice	
go, come	✓	violent, angry	✓✓
groom, bride		cat, lion	✓✓
dinner, breakfast		laden, heavy	✓✓
-	-	engage, marry	

Table 4.4 Top word pair outliers in counter-fitted vectors (compared to the SimLex gold standard ranking). A word pair is included in this list if its assigned rank is in the top/bottom 200 pairs and its position in the gold standard ranking differs by more than 500 positions.

Dataset	Train	Dev	Test	#Slots
Restaurants	1612	506	1117	4
Tourist Information	1600	439	225	9

Table 4.5 Number of dialogues in the dataset splits used for the DST experiments.

nine false antonyms: word pairs with predicted rank in the top (bottom) 200 word pairs and gold standard rank 500 or more positions lower (higher). Eight of these errors are fixed by counter-fitting: the difference between predicted and gold-standard ranks is now 100 or less. Interestingly, five of the eight corrected word pairs do not appear in the sets of linguistic constraints; these are indicated by double ticks in the table. This shows that secondary (i.e., indirect) interactions through the three terms of the cost function do contribute to the semantic content of the transformed vector space.

4.4.3 Improving Dialogue State Tracking

Table 4.5 shows the dialogue state tracking datasets used for evaluation. These datasets come from the Dialogue State Tracking Challenges 2 and 3 (Henderson et al., 2014a,b).

Four different sets of word vectors were used to construct semantic dictionaries: the original GloVe and Paragram-SL999 vectors, as well as versions counter-fitted to each domain ontology. The constraints used for counter-fitting were all those from the previous section as well as antonymy constraints between the slot values for each slot. All vocabulary words within some radius t of a slot value were treated as its rephrasings. The optimal value of t was determined using a grid search: a dictionary was generated and a DST model was trained (using that dictionary) for each potential t. These models were evaluated on the development set, and the highest-performing dictionary was used in the subsequent evaluation.[9]

Table 4.6 shows the performance of RNN models which used the constructed dictionaries. The dictionaries induced from the pre-trained vectors substantially improved tracking performance over the baselines (which used no semantic dictionaries). The dictionaries created using the counter-fitted vectors improved performance even further. Contrary to the SimLex-999 experiments, starting from the Paragram vectors did not lead to superior performance (compared to GloVe + counter-fitting), which shows that injecting the application-specific ontology is at least as important as the quality of the initial word vectors.

[9]The best performing thresholds t across all eight experiments were in the interval between 0.35 and 0.55, with development set performance showing relatively little variance between values in that interval.

Word Vector Space	Restaurants	Tourist Info
Baseline (no dictionary)	68.6	60.5
GloVe	72.5	60.9
GloVe + Counter-fitting	73.4	**62.8**
Paragram-SL999	73.2	61.5
Paragram-SL999 + Counter-fitting	**73.5**	61.9

Table 4.6 Performance of RNN belief trackers (ensembles of 4 models) which use: a) no semantic dictionary; b) dictionaries generated using the four sets of word vectors.

4.5 Conclusion

This chapter presented the counter-fitting method for injecting linguistic constraints into word vector space representations. The method efficiently post-processes word vectors to improve their usefulness for tasks which involve making semantic similarity judgements. Its focus on separating vector representations of antonymous word pairs led to substantial improvements on genuine similarity estimation tasks. Moreover, counter-fitting can tailor word vectors for the downstream task of dialogue state tracking by injecting antonymy constraints derived from dialogue ontologies into the word vector space. As shown in the evaluation, such specialised word vectors can be used to construct semantic dictionaries which improve the language understanding capabilities of spoken dialogue systems.

4.5.1 Drawbacks of Semantic Lexicons

There are several reasons why the approach of defining explicit string mappings for particular slot values does not promise to scale as statistical dialogue systems are deployed in increasingly complex dialogue domains.

1. **Morphology-Rich Languages**: word morphology induced by word gender, verb cases, declension and other linguistic phenomena can lead to a proliferation of word forms through their different inflections. For instance, the German word for Scottish (*Schottisch*) can take alternative inflectional word forms *Schottische*, *Schottisches*, *Schottischem*, *Schottischer*, *Schottischen*, while the Serbian adjective *jeftin* (cheap) can take word forms *jeftin*, *jeftina*, *jeftine*, *jeftino*, *jeftini*, *jeftinom*, *jeftinog*, *jeftinim*, and *jeftinih* depending on word gender and its use/position in the sentence. Any word-based model making use of semantic lexicons would require the system designer to specify these inflections to achieve robust performance in the target language.

2. **Scaling to Large Dialogue Domains**: large, more complex dialogue domains could potentially contain conflicting, context-dependent slot values. An example of these would be mid-priced laptops versus mid-range hard drive sizes, both of which may be referred to with a plethora of similar expressions (*medium*, *moderate*, *average*, etc.).

Despite these drawbacks, this chapter has shown that semantically specialised word vectors can improve DST performance when used to induce domain-specific semantic lexicons. In order to move towards a more data-driven paradigm, the next chapter presents a novel language understanding model which reasons entirely over word vectors, moving past the dependency on delexicalised *n*-grams, exact matching and semantic lexicons.

Chapter 5

The Neural Belief Tracker

This chapter presents the Neural Belief Tracker (NBT), presented in Mrkšić et al. (2017a); Mrkšić and Vulić (2018). The NBT is a language understanding model designed to move past the word-based delexicalisation paradigm and allow language understanding models to scale to linguistically rich user input across disparate dialogue domains and different languages.

5.1 Motivation

The dialogue state tracking (DST) component of a spoken dialogue system serves to interpret user input and update the belief state, which is used by the downstream *dialogue manager* to decide which action the system should perform next. The Dialogue State Tracking Challenge (DSTC) series of shared tasks provided a common evaluation framework accompanied by labelled DST datasets (Williams et al., 2016). In this framework, the dialogue system is supported by a *domain ontology* which describes the range of user intents the system can process. The ontology defines a collection of *slots* and the *values* that each slot can take. The system must track the search constraints expressed by users (*goals* or *informable* slots) and questions the users ask about search results (*requests*), taking into account each user utterance (input via a speech recogniser) and the dialogue context (e.g., what the system just said). The example in Figure 5.1 shows the true state after each user utterance in a three-turn conversation. As can be seen in this example, DST models depend on identifying mentions of ontology items in user utterances. This becomes a non-trivial task when confronted with lexical variation, the dynamics of context and noisy speech recognition output.

Historically, statistical approaches to building dialogue systems relied on separate Spoken Language Understanding (SLU) modules to address lexical variability within a single dialogue turn. However, training such models requires substantial amounts of domain-specific annotation. As seen in the previous two chapters, turn-level SLU and cross-turn DST can

User: I'm looking for a cheaper restaurant
`inform(price=cheap)`
System: Sure. What kind - and where?
User: Thai food, somewhere downtown
`inform(price=cheap, food=Thai, area=centre)`
System: The House serves cheap Thai food
User: Where is it?
`inform(price=cheap, food=Thai, area=centre);`
`request(address)`
System: The House is at 106 Regent Street

Figure 5.1 Annotated dialogue states in a sample dialogue. Underlined words show rephrasings which are typically handled using semantic dictionaries.

be coalesced into a single word-based DST model to achieve superior belief tracking performance. As explained in Chapter 3, joint models typically rely on a strategy known as *delexicalisation*, whereby slots and values mentioned in the text are replaced with generic labels. Once the dataset is transformed in this manner, one can extract a collection of template-like *n*-gram features such as [want *tagged-value* food]. To perform belief tracking, the shared model iterates over all slot-value pairs, extracting delexicalised feature vectors and making a separate binary decision regarding each slot value pair.

The Limits of Delexicalisation Delexicalisation introduces a hidden dependency that is rarely discussed: how can the model identify slot/value mentions that are not exact string matches with the values specified in the domain ontology? For toy domains, one can manually construct *semantic dictionaries* which list potential rephrasings for all slot values. Figure 5.2 gives an example of such a dictionary for three slot-value pairs in the DSTC2 dialogue domain. The use of such dictionaries is essential for the performance of existing delexicalisation-based models. In a way, these dictionaries reintroduce a more opaque version of domain-specific SLU modules. Unlike SLU modules, semantic dictionaries are not even learned from data, instead requiring (expensive) manual effort by the system designer.

FOOD=CHEAP: [affordable, budget, low-cost, low-priced, inexpensive, cheaper, ...]
RATING=HIGH: [best, high-rated, highly rated, top-rated, cool, chic, popular, trendy, ...]
AREA=CENTRE: [center, downtown, central, city centre, midtown, town centre, ...]

Figure 5.2 An example semantic dictionary with rephrasings for three ontology values.

The counter-fitting approach presented in Chapter 4 showed that domain-specific semantic lexicons can be constructed automatically, saving substantial human effort. However, the proposed approach is still fundamentally limited by the delexicalisation paradigm, which relies on exact matching of predefined rephrasings with ontology-defined slot values. It ignores linguistic phenomena such as morphology, multi-sense words, or contextual sentence meaning which can only be disambiguated by considering the entire dialogue history. For this reason, word-based models powered by semantic dictionaries do not promise to scale to more complex dialogue domains with large domain ontologies.

The primary motivation of the work presented in this chapter is to overcome the limitations that delexicalisation imposes on belief tracking models. Two new models are introduced, collectively called the Neural Belief Tracker (NBT) family. The proposed models couple SLU and DST, efficiently learning to handle variation without requiring any hand-crafted resources. To do that, NBT models move away from exact string matching and instead reason entirely over pre-trained word vectors. The vectors making up the user utterance and preceding system output are first composed into intermediate distributed representations. These representations are then used to decide which of the ontology-defined intents have been expressed by the user up to that point in the conversation. The NBT model efficiently learns from the available data by: **1)** leveraging semantic information from pre-trained word vectors to resolve lexical/morphological ambiguity; **2)** maximising the number of parameters shared across ontology values; and **3)** having the flexibility to learn domain-specific paraphrasings and other kinds of variation that make it infeasible to rely on exact matching and delexicalisation as a robust strategy.

To the best of my knowledge, NBT models are the first to successfully use pre-trained word vector spaces to improve the language understanding capability of belief tracking models. Evaluation on two datasets shows that: **a)** NBT models match the performance of delexicalisation-based models which make use of hand-crafted semantic lexicons; and **b)** the NBT models significantly outperform those models when such resources are not available. Consequently, this framework is better-suited to scaling belief tracking models for deployment in real-world dialogue systems operating over sophisticated application domains where the creation of such domain-specific lexicons would be infeasible.

5.2 The Neural Belief Tracker

The Neural Belief Tracker (NBT) is a model designed to detect the slot-value pairs that make up the user's goal at a given turn during the flow of dialogue. Its input consists of the system dialogue acts preceding the user input, the user utterance itself, and a single candidate

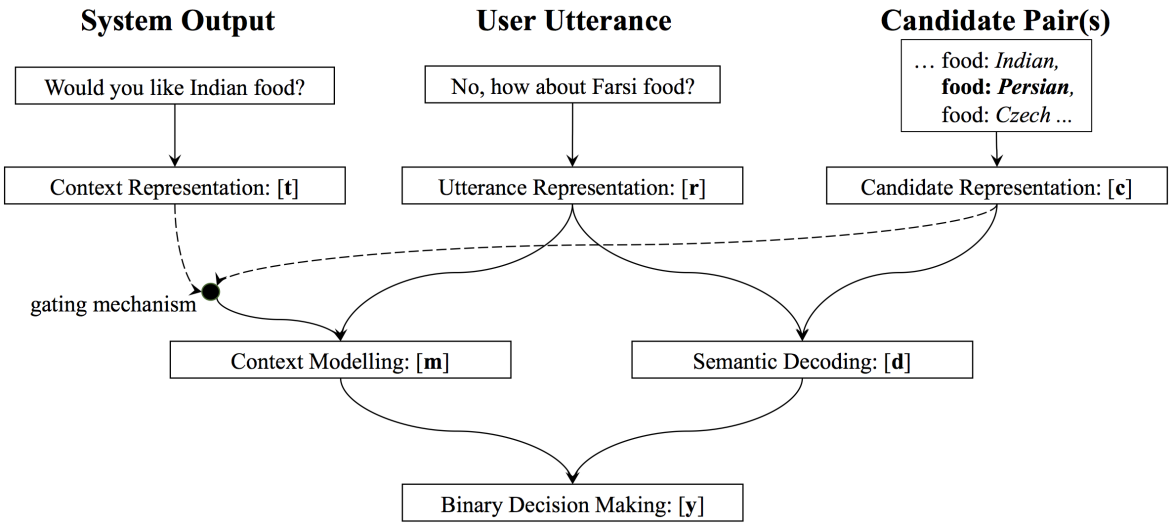


Figure 5.3 Architecture of the NBT Model. The implementation of the three representation learning subcomponents can be modified, as long as these produce adequate vector representations which the downstream model components can use to decide whether the current candidate slot-value pair was expressed in the user utterance (taking into account the preceding system act).

slot-value pair that it needs to make a decision about. For instance, the model might have to decide whether the goal FOOD=*Italian* has been expressed in *'I'm looking for good pizza'*. To perform belief tracking, the NBT model *iterates* over all candidate slot-value pairs (defined by the ontology), and decides which ones have just been expressed by the user.

Figure 5.3 presents the flow of information in the NBT models. The first layer in the NBT hierarchy performs representation learning given the three model inputs, producing vector representations for the user utterance $\mathbf{r}$, the current candidate slot-value pair $\mathbf{c}$ and the system dialogue acts $\mathbf{t_q}, \mathbf{t_s}, \mathbf{t_v}$. Subsequently, the learned vector representations interact through the *context modelling* and *semantic decoding* submodules to obtain the intermediate *interaction summary* vectors $\mathbf{d_r}, \mathbf{d_c}$ and $\mathbf{d}$ (the full definitions of these will be provided in the following sections). These are used as input to the final *decision-making* module which decides whether the user expressed the intent represented by the candidate slot-value pair.

5.2.1 Representation Learning

For any given user utterance, system act(s) and candidate slot-value pair, the representation learning submodules produce vector representations which act as input for the downstream components of the model. All representation learning subcomponents make use of pre-trained collections of word vectors. As shown in the previous chapter, specialising word

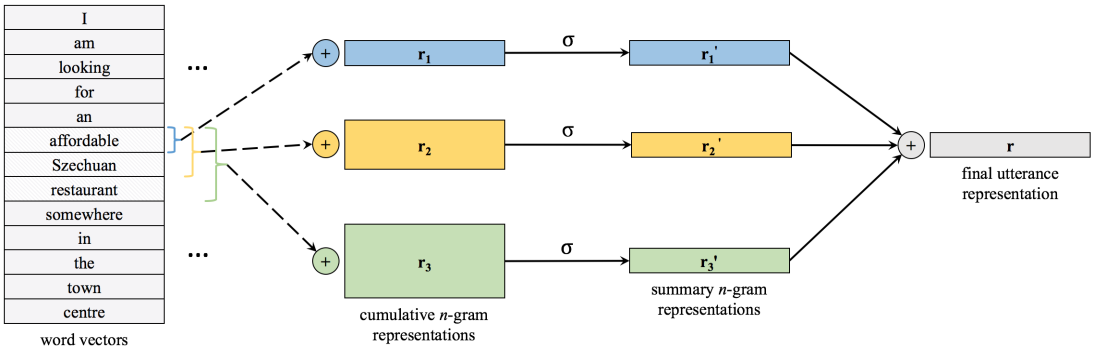


Figure 5.4 NBT-DNN MODEL. Word vectors of n-grams ($n = 1, 2, 3$) are summed to obtain *cumulative* n-grams, then passed through another hidden layer and summed to obtain the utterance representation $\mathbf{r}$.

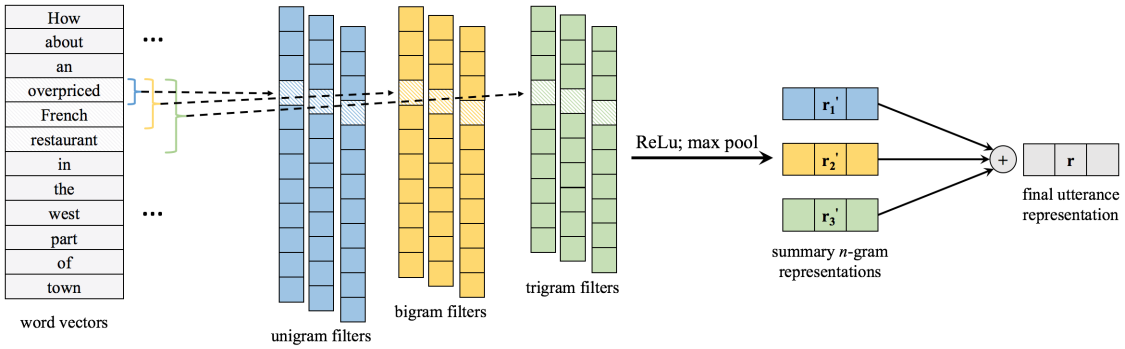


Figure 5.5 NBT-CNN Model. L convolutional filters of window sizes $1, 2, 3$ are applied to word vectors of the given utterance ($L = 3$ in the diagram, but $L = 300$ in the system). The convolutions are followed by the ReLU activation function and max-pooling to produce summary n-gram representations. These are summed to obtain the utterance representation $\mathbf{r}$.

vectors to express *semantic similarity* rather than *relatedness* is essential for improving belief tracking performance. For this reason, semantically-specialised Paragram-SL999 word vectors (Wieting et al., 2015) are used throughout this chapter. The NBT training procedure keeps these vectors fixed: that way, at test time, unseen words semantically related to familiar slot values (i.e. *inexpensive* to *cheap*) will be recognised purely by their position in the original vector space (see Rocktäschel et al. (2016)). This means that the NBT model parameters can be shared across all values of the given slot, or even across all slots.

Let u represent a user utterance consisting of k_u words $u_1, u_2, \ldots, u_{k_u}$. Each word has an associated word vector $\mathbf{u}_1, \ldots, \mathbf{u}_{k_u}$. Two model variants which differ in the method used to produce vectorial representations of u are investigated: NBT-DNN and NBT-CNN. Both act over the constituent n-grams of the utterance. Let $\mathbf{v}_i^n$ be the concatenation of the n word vectors starting at index i, so that:

$$\mathbf{v}_i^n = \mathbf{u}_i \oplus \ldots \oplus \mathbf{u}_{i+n-1} \tag{5.1}$$

where $\oplus$ denotes vector concatenation. The simpler of the two models, termed NBT-DNN, is shown in Figure 5.4. This model computes cumulative n-gram representation vectors $\mathbf{r}_1$, $\mathbf{r}_2$ and $\mathbf{r}_3$, which are the n-gram 'summaries' of the unigrams, bigrams and trigrams in the user utterance:

$$\mathbf{r}_n = \sum_{i=1}^{k_u - n + 1} \mathbf{v}_i^n \tag{5.2}$$

Each of these three vectors is then non-linearly mapped to intermediate representations of the same dimension:

$$\mathbf{r}'_n = \sigma(W_n^s \mathbf{r}_n + b_n^s) \tag{5.3}$$

where the weight matrices and bias terms map the cumulative n-grams to vectors of the same dimensionality and σ denotes the sigmoid activation function. A separate set of parameters is maintained for each slot (indicated by superscript s). The three vectors are then summed to obtain a single representation for the user utterance:

$$\mathbf{r} = \mathbf{r}'_1 + \mathbf{r}'_2 + \mathbf{r}'_3 \tag{5.4}$$

The cumulative n-gram representations used by this model are just unweighted sums of all word vectors in the utterance. Ideally, the model should learn to recognise which parts of the utterance are more relevant for the subsequent classification task. For instance, it could

learn to ignore verbs or stop words and pay more attention to adjectives and nouns which are more likely to express slot values.

NBT-CNN The second model draws inspiration from successful applications of Convolutional Neural Networks (CNNs) for language understanding (Collobert et al., 2011; Kalchbrenner et al., 2014; Kim, 2014). These models typically apply a number of convolutional filters to n-grams in the input sentence, followed by non-linear activation functions and max-pooling. Following this approach, the NBT-CNN model applies $L = 300$ different filters for n-gram lengths of $1, 2$ and 3 (Figure 5.5). Let $F_n^s \in R^{L \times nD}$ denote the collection of filters for each value of n, where $D = 300$ is the word vector dimensionality. If $\mathbf{v}_i^n$ denotes the concatenation of n word vectors starting at index i, let $\mathbf{m}_n = [\mathbf{v}_1^n; \mathbf{v}_2^n; \ldots; \mathbf{v}_{k_u-n+1}^n]$ be the list of n-grams that convolutional filters of length n run over. The three intermediate representations are then given by:

$$R_n = F_n^s \, \mathbf{m}_n \tag{5.5}$$

Each column of the intermediate matrices R_n is produced by a single convolutional filter of length n. The summary n-gram representations are then obtained by pushing the intermediate representations through a rectified linear unit (ReLU) activation function (Nair and Hinton, 2010) and then max-pooling over time (i.e., columns of the matrix) to get a single feature for each of the L filters applied to the utterance:

$$\mathbf{r}'_n = \texttt{maxpool}\left(\texttt{ReLU}\left(R_n + b_n^s\right)\right) \tag{5.6}$$

where b_n^s is a bias term broadcast across all filters. Finally, the three summary n-gram representations are summed to obtain the final utterance representation vector $\mathbf{r}$ (as in Equation 5.4). The NBT-CNN model is (by design) better suited to longer utterances, as its convolutional filters interact directly with subsequences of the utterance, and not just their noisy summaries given by the NBT-DNN's cumulative n-grams.

5.2.2 Semantic Decoding

The NBT diagram in Figure 5.3 shows that the utterance representation $\mathbf{r}$ and the candidate slot-value pair representation $\mathbf{c}$ directly interact through the *semantic decoding* module. This component decides whether the user explicitly expressed an intent matching the current candidate pair (i.e., without taking the dialogue context into account). Examples of such matches would be *'I want Thai food'* with FOOD=*Thai* or more demanding ones such as *'a pricey restaurant'* with PRICE=*expensive*. This is where the use of high-quality pre-trained word vectors comes into play: a delexicalisation-based model could deal with the former

example but would be helpless in the latter case, unless a human expert had provided a semantic dictionary listing all potential rephrasings for each value in the domain ontology.

Let the vector space representations of a candidate pair's slot name and value be given by $\mathbf{c_s}$ and $\mathbf{c_v}$ (with vectors of multi-word slot names/values summed together). The NBT model learns to map this tuple into a single vector $\mathbf{c}$ of the same dimensionality as the utterance representation $\mathbf{r}$. These two representations are then forced to interact in order to learn a similarity metric which discriminates between interactions of utterances with slot-value pairs that they either do or do not express:

$$\mathbf{c} = \sigma\left(W_c^s(\mathbf{c_s} + \mathbf{c_v}) + b_c^s\right) \tag{5.7}$$

$$\mathbf{d} = \mathbf{r} \otimes \mathbf{c} \tag{5.8}$$

where $\otimes$ denotes *element-wise* vector multiplication. The dot product, which may seem like the more intuitive similarity metric, would reduce the rich set of features in $\mathbf{d}$ to a single scalar. The element-wise multiplication allows the downstream network to make better use of its parameters by learning non-linear interactions between sets of features in $\mathbf{r}$ and $\mathbf{c}$.[1]

5.2.3 Context Modelling

This 'decoder' does not yet suffice to extract intents from utterances in human-machine dialogue. To understand some queries, the belief tracker must be aware of *context*, that is the flow of dialogue leading up to the latest user utterance. While all previous system and user utterances are important, the most relevant one is the last system utterance, in which the dialogue system could have performed (among others) one of the following *system acts*:

1. **System Request**: The system asks the user about the value of a specific slot T_q. If the system utterance is: *'what price range would you like?'* and the user answers with *'any'*, the model should know that the user is talking about PRICE RANGE, and not about other slots such as AREA or FOOD.

2. **System Confirm:** The system asks the user to confirm whether a specific slot-value pair (T_s, T_v) is part of their desired constraints. For example, if the user responds to *'how about Turkish food?'* with *'yes'*, the model must be aware of the system act in order to update the belief state with the correct slot value.

[1] An alternative approach would be to concatenate $\mathbf{r}$ and $\mathbf{c}$ and pass that vector to the downstream decision-making network. However, this set-up led to very weak performance since the relatively small datasets used in evaluation did not suffice for the network to learn to model the interaction between the two feature vectors.

If one makes the Markovian decision to only consider the last set of system acts, context modelling can be incorporated into the NBT by adding another hierarchical component. Let $\mathbf{t_q}$ and $(\mathbf{t_s}, \mathbf{t_v})$ be the word vectors of the arguments for the system request and confirm acts (zero vectors if none). The model computes the following measures of similarity between the system acts, candidate pair $(\mathbf{c_s}, \mathbf{c_v})$ and utterance representation $\mathbf{r}$:

$$\mathbf{m_r} = (\mathbf{c_s} \cdot \mathbf{t_q})\mathbf{r} \tag{5.9}$$

$$\mathbf{m_c} = (\mathbf{c_s} \cdot \mathbf{t_s})(\mathbf{c_v} \cdot \mathbf{t_v})\mathbf{r} \tag{5.10}$$

where $\cdot$ denotes dot product. The computed similarity terms act as gating mechanisms which only pass the utterance representation through if the system asked about the current candidate slot or slot-value pair. This type of interaction is particularly useful for the confirm system act: if the system asks the user to confirm, the user is likely not to mention any slot values, but to just respond affirmatively or negatively. This means that the model must consider the *three-way interaction* between the utterance, candidate slot-value pair and the slot value pair offered by the system. If (and only if) the latter two are the same should the model consider the affirmative or negative polarity of the user utterance when making its binary decision.

Binary Decision Maker The intermediate representations for semantic decoding and context modelling are passed through another hidden layer and combined to make the final decision. If $\phi_{dim}(\mathbf{x}) = \sigma(W\mathbf{x} + b)$ is a layer which maps input vector $\mathbf{x}$ to a vector of size dim, the input to the final binary softmax (which represents the decision) is given by:

$$\mathbf{y} = \phi_2\big(\phi_{100}(\mathbf{d}) + \phi_{100}(\mathbf{m_r}) + \phi_{100}(\mathbf{m_c})\big) \tag{5.11}$$

5.2.4 Rule-Based Belief State Updates

In spoken dialogue systems, belief tracking models operate over the output of automatic speech recognition (ASR). Despite improvements to speech recognition, the need to make the most out of imperfect speech recognition will persist as dialogue systems are deployed to increasingly noisy environments.

In this work, a simple rule-based belief state update mechanism is defined and applied to ASR N-best lists. For dialogue turn t, let sys^{t-1} denote the preceding system output, and let h^t denote the list of N ASR hypotheses h_i^t with posterior probabilities p_i^t. For any hypothesis h_i^t, slot s and slot value $v \in V_s$, NBT models estimate $\mathbb{P}(s, v \mid h_i^t, sys^{t-1})$, which is the (turn-level) probability that (s, v) was expressed in the given hypothesis. The predictions for N such hypotheses are then combined using the following expression:

$$\mathbb{P}(s,v \mid h^t, sys^{t-1}) = \sum_{i=1}^{N} p_i^t \, \mathbb{P}\left(s,v \mid h_i^t, sys^t\right) \tag{5.12}$$

This turn-level belief state estimate is then combined with the (cumulative) belief state up to time $(t-1)$ to get the updated belief state estimate:

$$\mathbb{P}(s,v \mid h^{1:t}, sys^{1:t-1}) \;=\; \lambda \, \mathbb{P}\left(s,v \mid h^t, sys^{t-1}\right) + \; (1-\lambda) \, \mathbb{P}\left(s,v \mid h^{1:t-1}, sys^{1:t-2}\right) \tag{5.13}$$

where λ is the coefficient which determines the relative weight of the turn-level and previous turns' belief state estimates.[2] For slot s, the set of its *detected values* at turn t is given by:

$$V_s^t = \{v \in V_s \;\mid\; \mathbb{P}\left(s,v \mid h^{1:t}, sys^{1:t-1}\right) \geq 0.5\} \tag{5.14}$$

For informable (i.e. goal-tracking) slots, the value in V_s^t with the highest probability is chosen as the current goal (if $V_s^t \neq \{\emptyset\}$). For requests, all slots in V_{req}^t are deemed to have been requested. As requestable slots serve to model single-turn user queries, they require no belief tracking across turns.

5.2.5 Statistical Belief State Updates

The NBT model eschews the use of semantic lexicons by relying on semantically-specialised vector space representations to deal with linguistic variation. To emulate the principal advantage of delexicalisation-based approaches, which can deal with slot values not seen in the training data, the NBT models decompose the (per-slot) multi-class value prediction problem into many binary ones. The model iterates through all slot values defined by the ontology and decides which ones have just been expressed by the user. These can then be combined with the previous belief state using a rule-based update, as shown in the previous section. This approach comes at a cost: the NBT is little more than an SLU decoder capable of modelling the preceding system acts. Its parameters do not learn to handle the previous belief state, which is essential for probabilistic modelling in POMDP-based dialogue systems (Thomson and Young, 2010; Young et al., 2010). This section shows how the NBT framework can be extended to (learn to) perform statistical belief state updates.

Problem Definition For any given slot s, let $\mathbf{b}_s^{t-1}$ be the true belief state at time $t-1$ (this is a vector of length $|V_s|+2$, accounting for all slot values and two special values, *dontcare* and *NONE*). At time t, let the intermediate representations representing the preceding system

[2] The coefficient was tuned on the DSTC 2 development set; the best performance was achieved at $\lambda = 0.55$.

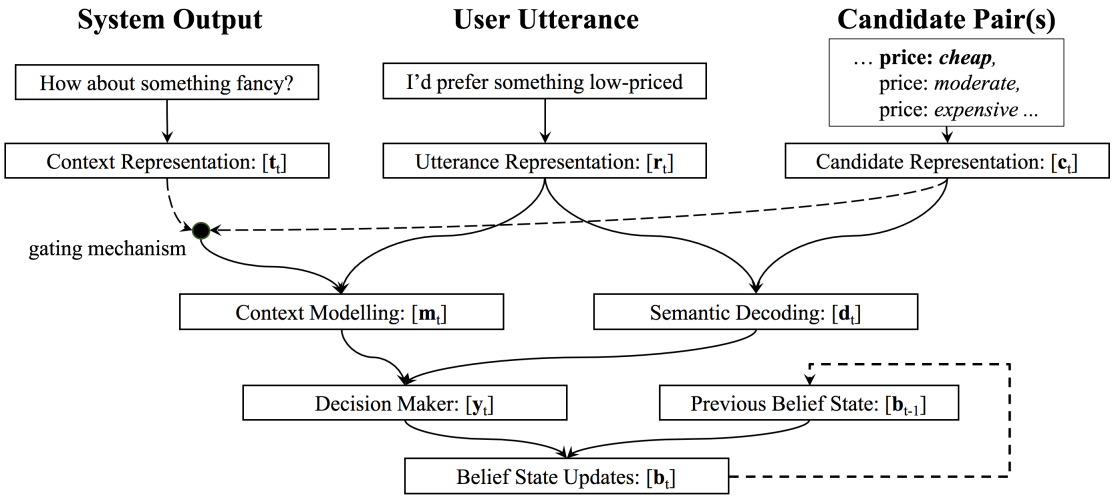


Figure 5.6 The architecture of the NBT Model with the learned statistical belief state update. Rather than just producing separate turn-level estimates for all values ($\mathbf{y}_t$), this model learns to combine these predictions with the previous belief state $\mathbf{b}_{t-1}$.

acts and the current user utterance be $\mathbf{m}^t$ and $\mathbf{r}^t$. When the NBT model decides about slot value $v \in V_s$, it uses the intermediate candidate slot-value representation for the value, $\mathbf{c}_v^t$, to implicitly parametrise the model for making a decision about that value.

In the previous section, the binary decision making module of the rule-based NBT models produced independent estimates for each slot value. In the case of *statistical* belief state updates, the turn-level estimates for all values are still produced separately, yielding a vector of turn-level predictions $\mathbf{y}_s^t$. However, this vector is then combined with the previous belief state estimate for slot s, $\mathbf{b}_s^{t-1}$, to estimate the new belief state $\mathbf{b}_s^t$:

$$\mathbf{b}_s^t = \phi(\mathbf{y}_s^t, \mathbf{b}_s^{t-1}) \tag{5.15}$$

In line with the NBT framework, the only criteria for the belief state update mechanism ϕ are: **a)** that it is a differentiable function that can backpropagated through during NBT training; and **b)** that it produces a valid probability distribution $\mathbf{b}_s^t$ as output. Figure 5.6 shows the NBT architecture expanded with the belief state update component.

From Binary Decisions to Probability Distributions The rule-based update used *separate* estimates for each slot value, choosing the most likely value as the current prediction. An advantage of that approach is that it implicitly models the probability that no value had been expressed. If $P(s, v|\mathbf{r^t}, \mathbf{m^t}) < 0.5)$ for all slot value pairs, the belief state remains unchanged.

Modelling the belief state jointly means that a separate NONE value must be added to the set of candidate slots to capture the probability that no value had been expressed up to that dialogue turn. As all slot values are still predicted using the same set of parameters (producing a pre-softmax feature vector of size equal to number of slot values), an additional constant scalar feature is appended to the vector to represent the NONE value.[3] The remaining $|V|$ features, produced by NBT decoders with tied parameters, are learned jointly and implicitly parametrised by the values' word vector representations.

Variants of the Statistical Belief State Update

Two different approaches for combining the previous belief state $\mathbf{b}_s^{t-1}$ with the new turn-level prediction $\mathbf{y}_s^t$ are presented next.

1. Learned Markovian Update: the previous belief state $\mathbf{b}_s^{t-1}$ and the current turn-level estimate $\mathbf{y}_s^t$ are combined using a one-step belief state update:

$$\mathbf{b}_s^t = softmax\left(W_{curr}\mathbf{y}_s^t + W_{past}\mathbf{b}_s^{t-1}\right) \tag{5.16}$$

where the W_{curr} and W_{past} matrices learn to combine the two signals into a new belief state. This model violates the NBT design paradigm: each row of the two matrices learns to operate over *specific* slot values. This means the model will not learn to predict or maintain slot values as part of the belief state if it has not encountered these values during training. Even though turn-level NBT output $\mathbf{y}_s^t$ may contain the right prediction, the parameters of the corresponding row in W_{curr} will not be trained to update the belief state, since its parameters (for the given value) will not have been updated during training. Similarly, the same row in W_{past} will not learn to maintain the given slot value as part of the belief state.

To overcome the data sparsity and preserve the NBT model's ability to deal with unseen values, one can use the fact that there are fundamentally only two different actions that a belief tracker needs to perform: **1)** maintain the same prediction as in the previous turn; or **2)** update the prediction given strong indication that a new slot value has been expressed. To facilitate transfer learning between different ontology values, the second belief state update model introduces additional constraints for the one-step belief state update.

[3] Appending values of 0 or 1 produced the same results, as the remaining feature estimates scaled to larger values to indicate whether other slot values had been expressed.

2. Constrained Markovian Update: this model constrains the two matrices so that each of them contains only two different scalar values. One scalar value populates the diagonal elements, and the other one is replicated for all off-diagonal elements:

$$W_{curr,i,j} = \begin{cases} a_{curr}, & \text{if } i = j \\ b_{curr}, & \text{otherwise} \end{cases} \qquad W_{past,i,j} = \begin{cases} a_{past}, & \text{if } i = j \\ b_{past}, & \text{otherwise} \end{cases} \tag{5.17}$$

where the four scalar values are learned jointly with the NBT parameters. The diagonal values learn the relative importance of propagating the previous value (a_{past}) or of accepting a newly detected value (a_{curr}). The off-diagonal elements learn how turn-level signals (b_{curr}) or past probabilities for other values (b_{past}) impact the predictions for the current belief state (i.e., how hard it is to overwrite any value from the belief state with a different value). The parameters acting over all slot values are in this way tied, ensuring that the model can deal with slot values it did not encounter during training.

The two versions of the learned belief state update are compared to the rule-based update presented in the previous section. Unlike that update, both variants of the statistical update require no tuning, and as such deliver fully data-driven belief tracking.

5.3 Experiments

5.3.1 Datasets

Two datasets were used for training and evaluation. Both consist of user conversations with task-oriented dialogue systems designed to help users find suitable restaurants around Cambridge, UK. The two corpora share the same domain ontology, which contains three *informable* (i.e., goal-tracking) slots: FOOD, AREA and PRICE. The users can specify values for these slots in order to find restaurants which best meet their criteria. Once the system suggests a restaurant, the users can ask about the values of up to eight *requestable* slots (PHONE NUMBER, ADDRESS, etc.). The two datasets are:

1. **DSTC 2**: This experiment uses the transcriptions, ASR hypotheses and turn-level semantic labels provided for the Dialogue State Tracking Challenge 2 (Henderson et al., 2014a). The official transcriptions contain various spelling errors which were corrected manually.[4] The training data contains 2207 dialogues and the test set consists of 1117 dialogues. The NBT models are trained on transcriptions but tested on the ASR hypotheses provided in the original challenge.

[4]The cleaned version of the dataset is available at *mi.eng.cam.ac.uk/ nm480/dstc2-clean.zip*

2. **WOZ 2.0**: Wen et al. (2017) performed a Wizard of Oz style experiment in which Amazon Mechanical Turk users assumed the role of the system or the user of a task-oriented dialogue system based on the DSTC 2 ontology. Users typed instead of using speech, which means performance in the WOZ experiments is more indicative of the model's capacity for semantic understanding than its robustness to ASR errors. Whereas in the DSTC 2 dialogues users would quickly adapt to the system's (lack of) language understanding capability, the WOZ experimental design gave them freedom to use more sophisticated language. For these experiments, the original WOZ dataset from Wen et al. (2017) was expanded, using the same data collection procedure, yielding a total of 1200 dialogues. These dialogues were divided into 600 training, 200 validation and 400 test set dialogues.[5]

Training Examples The two corpora are used to create training data for two separate experiments. For each dataset, each training set utterance is used to generate one example for *each* of the slot-value pairs in the ontology. An example consists of a transcription, its context (i.e., a list of preceding system acts) and a candidate slot-value pair. The binary label for each example indicates whether or not its utterance and context express the example's candidate pair. For instance, '*I would like Irish food*' would generate a positive example for candidate FOOD=*Irish*, and a negative example for every other slot value in the ontology.

Evaluation Two key evaluation metrics introduced by Henderson et al. (2014a) are used to assess the quality of the trained models:

1. **Goals** ('joint goal accuracy'): the proportion of dialogue turns where all the user's search goal constraints were correctly identified;

2. **Requests**: similarly, the proportion of dialogue turns where user's requests for information were identified correctly.

5.3.2 Models

Two NBT model variants are evaluated: NBT-DNN and NBT-CNN. The Adam optimizer (Kingma and Ba, 2015) with cross-entropy loss is used to train the models, backpropagating through all the NBT subcomponents while keeping the pre-trained word vectors fixed (in order to allow the model to deal with unseen words at test time). The model is trained separately for each slot. Due to the high class bias (most of the constructed examples are negative), a fixed number of positive examples is included in each mini-batch.

[5]The WOZ 2.0 dataset is available at *mi.eng.cam.ac.uk/ñm480/woz_2.0.zip*.

Baseline Models For each of the two datasets, the NBT models are compared to:

1. **Delexicalisation-Based Models** A baseline system that implements a well-known competitive delexicalisation-based model for that dataset. For DSTC 2, the model is that of Henderson et al. (2014c; 2014d). This model is an *n*-gram based neural network model with recurrent connections between turns (but not inside utterances) which replaces occurrences of slot names and values with generic delexicalised features. For WOZ 2.0, the NBT models are compared to a more sophisticated belief tracking model presented in (Wen et al., 2017). This model uses an RNN for belief state updates and a CNN for turn-level feature extraction. Unlike NBT-CNN, their CNN operates not over vectors, but over delexicalised features akin to those used by Henderson et al. (2014c).

2. **Delexicalisation + Dictionary** The same baseline model supplemented with a task-specific semantic dictionary (produced by the baseline system creators).[6] The DSTC 2 dictionary contains only three rephrasings. Nonetheless, the use of these rephrasings translates to substantial performance gains (see Table 5.1). This result supports the claim that the vocabulary used by Mechanical Turkers in DSTC 2 was constrained by the system's inability to cope with lexical variation and ASR noise. The WOZ dictionary includes 38 rephrasings, showing that the unconstrained language used by Mechanical Turkers in the Wizard-of-Oz setup requires more elaborate lexicons.

Both baseline models map exact matches of ontology-defined intents (and their lexicon-specified rephrasings) to one-hot delexicalised *n*-gram features. This means that pre-trained vectors cannot be incorporated directly into these models.

NBT Hyperparameters Model hyperparameters were tuned on the respective validation sets. The initial Adam learning rate was set to 0.001, and $\frac{1}{8}$th of positive examples were included in each mini-batch. The batch size did not affect performance: it was set to 256 in all experiments. Gradient clipping (to $[-2.0, 2.0]$) was used to handle exploding gradients. The dropout technique (Srivastava et al., 2014) was used for regularisation (with 50% dropout rate on all intermediate distributed representations). Both NBT models were implemented in the TensorFlow framework (Abadi et al., 2015).

[6] The two dictionaries are available at *mi.eng.cam.ac.uk/ñm480/sem-dict.zip*

DST Model	DSTC2		WOZ 2.0	
	Goals	Requests	Goals	Requests
Delexicalisation-Based Model	69.1	95.7	70.8	87.1
Delexicalisation-Based Model + Semantic Dictionary	72.9*	95.7	83.7*	87.6
NEURAL BELIEF TRACKER: NBT-DNN	72.6*	96.4	**84.4***	91.2*
NEURAL BELIEF TRACKER: NBT-CNN	**73.4***	**96.5**	84.2*	**91.6***

Table 5.1 DSTC2 and WOZ 2.0 test set accuracies for: **a)** joint goals; and **b)** turn-level requests. The asterisk indicates statistically significant improvement over the baseline trackers (paired t-test; $p < 0.05$).

5.4 Results

Table 5.1 shows the performance of NBT models trained and evaluated on DSTC 2 and WOZ 2.0 datasets. The NBT models outperformed the baseline models in terms of both joint goal and request accuracies. For goals, the gains are *always* statistically significant (paired t-test, $p < 0.05$). Moreover, there was no statistically significant variation between the NBT and the lexicon-supplemented models, showing that the NBT can handle semantic relations which otherwise had to be explicitly encoded in semantic dictionaries.

While the NBT models perform well across the board, one can compare their performance on the two datasets to understand the strengths of this framework. The improvement over the baseline is greater on WOZ 2.0, which corroborates the intuition that the NBT's ability to learn linguistic variation is vital for this dataset containing longer sentences, richer vocabulary and no ASR errors. By comparison, the language of the subjects in the DSTC2 dataset is less rich, and compensating for ASR errors is the main hurdle: given access to the DSTC2 test set transcriptions, the NBT models' goal accuracy rises to 0.96. This indicates that future work should focus on better ASR compensation if the model is to be deployed in environments with challenging acoustics.

5.4.1 The Importance of Word Vector Spaces

The NBT models use the semantic relations embedded in the pre-trained word vectors to handle semantic variation and produce high-quality intermediate representations. Table 5.2 shows the performance of NBT-CNN[7] models making use of three different word vector collections: **1)** 'random' word vectors initialised using the XAVIER initialisation (Glorot and Bengio, 2010); **2)** distributional GloVe vectors (Pennington et al., 2014), trained using co-occurrence information in large textual corpora; and **3)** *semantically specialised* Paragram-

[7]The NBT-DNN model showed the same performance trends. For brevity, only NBT-CNN is shown here.

Word Vectors	DSTC2		WOZ 2.0	
	Goals	Requests	Goals	Requests
XAVIER (No Info.)	64.2	81.2	81.2	90.7
GloVe	69.0*	96.4*	80.1	91.4
Paragram-SL999	**73.4***	**96.5***	**84.2***	**91.6**

Table 5.2 DSTC2 and WOZ 2.0 test set performance (*joint goals* and *requests*) of the NBT-CNN model making use of three different word vector collections. The asterisk indicates statistically significant improvement over the baseline XAVIER (randomly initialised) word vectors (paired t-test; $p < 0.05$).

SL999 vectors (Wieting et al., 2015), which are obtained by injecting *semantic similarity constraints* from the Paraphrase Database (Ganitkevitch et al., 2013) into the distributional GloVe vectors in order to improve their semantic content.

The results in Table 5.2 show that the use of semantically specialised word vectors leads to considerable performance gains: Paragram-SL999 vectors (significantly) outperformed GloVe and XAVIER vectors for goal tracking on both datasets. The gains are particularly robust for noisy DSTC 2 data, where both collections of pre-trained vectors consistently outperformed random initialisation. The gains are weaker for the noise-free WOZ 2.0 dataset, which seems to be large (and clean) enough for the NBT model to learn task-specific rephrasings and compensate for the lack of semantic content in the word vectors. For this dataset, GloVe vectors do not improve over the randomly initialised ones. A potential explanation for the drop in performance could lie in the fact that distributional models keep related, yet antonymous words close together (e.g., *north* and *south*, *expensive* and *inexpensive*), offsetting the useful semantic content embedded in this vector spaces.

5.4.2 Learning the Belief State Update

Table 5.3 shows the performance of the two variants of the statistical belief state update, comparing them to the rule-based update used in previous experiments.[8] The performance of statistical updates varies across the two datasets. For DSTC 2, the rule-based update tuned on the DSTC 2 development set consistently outperforms the statistical updates, showing that the end-to-end learning mechanism implemented by statistical update mechanisms can not cope with the speech recognition errors.[9] The discrepancy in performance is likely due

[8]The Paragram-SL999 vectors were used for these experiments. Similar drops in performance as those in Table 5.2 are observed when the other two word vector collections are deployed.

[9]The DSTC 2 model is again trained using the training set transcriptions provided in the original challenge. At test time, the joint prediction for the 10 ASR hypotheses is given by the weighted sum of predictions for each of the individual ASR hypotheses.

Model Variant	DSTC 2	WOZ 2.0
Markovian Belief State Update	67.8	82.1
Constrained Markovian Belief State Update	68.4	**84.8**
Rule-Based Belief State Update	**73.4**	84.2

Table 5.3 The joint goal accuracy of the two variants of the statistical belief state update, compared to the hand-tuned, rule-based belief state update (average performance of four models). This table omits the performance on requests, reported in all previous experiments in this chapter. That metric showed the performance for single-turn questions, and these are not affected by changes to the belief state update mechanism employed by the model.

to the fact that the rule-based update is tuned using the performance over speech recognition hypotheses, whereas the statistical update has no access to the speech recognition output during training (or during hyperparameter optimisation).

For the WOZ 2.0 dataset, the statistical updates come closer to the performance of the rule-based update. Interestingly, the constrained Markovian update outperforms both the value-specific Markovian update and the rule-based update. This shows that the transfer learning between values facilitated by the constrained model (which uses only four scalar values as parameters) is more important for belief tracking performance than the substantially larger set of parameters used by the value-specific update mechanism.

The results in Table 5.3 make a wider point about the relative importance of intra-turn Dialogue State Tracking and turn-level Spoken Language Understanding. In environments with high ASR noise, the quality of the learned dialogue state tracking rules is essential for overall language understanding performance. Conversely, in environments with no speech recognition errors (such as text-based chat interfaces), the quality of the SLU module determines how well the system can cope with linguistic variation, while more elaborate DST mechanisms have limited impact on the overall language understanding performance.

5.5 Conclusion

This chapter presented a novel neural belief tracking (NBT) framework designed to overcome current obstacles to deploying dialogue systems to real-world dialogue domains. The NBT models offer the known advantages of coupling Spoken Language Understanding and Dialogue State Tracking, without relying on hand-crafted semantic lexicons to achieve state-of-the-art performance. The presented evaluation demonstrated these benefits: the NBT models match the performance of models which make use of such lexicons and vastly outperform them when these are not available. Finally, the performance of NBT models improves

with the semantic quality of the underlying word vectors. To the best of my knowledge, this work was the first to move past intrinsic evaluation and show that *semantic specialisation* boosts performance in downstream language understanding tasks.

The work presented in subsequent chapters will show how the coupling between semantic specialisation and data-driven language understanding performed by NBT models can be used to train belief tracking models which work across different languages, offsetting the problems caused by linguistic phenomena such as word gender, compounding or declension. To do that, the next chapter revisits the notion of semantic specialisation proposed in Chapter 4, showing how existing semantic lexicons can be used to craft high-quality semantic word vector spaces across many languages. The subsequent chapter will then show how such vector spaces can be used to bootstrap high quality language understanding models for lower-resource languages.

Chapter 6

Exploiting Lexical Resources

This chapter presents my work on *semantic specialisation*, first explored in Chapter 4. The work on counter-fitting showed that semantic lexicons can be produced by injecting linguistic constraints into word vector spaces and then using words' neighbours as their rephrasings. The work on the Neural Belief Tracker showed that use of such vectors substantially boosts the performance of data-driven language understanding models. This chapter focuses on the semantic specialisation paradigm in more detail, showing that external linguistic constraints can be used to induce semantically specialised vectors across a plethora of different languages.

6.1 Introduction

Word representation learning has become a research area of central importance in modern natural language processing. The common techniques for inducing distributed word representations are grounded in the distributional hypothesis, relying on co-occurrence information in large textual corpora to learn meaningful word representations (Levy and Goldberg, 2014; Mikolov et al., 2013b; Ó Séaghdha and Korhonen, 2014; Pennington et al., 2014). Recently, methods which go beyond stand-alone unsupervised learning have gained increased popularity. These models typically build on distributional ones by using human- or automatically-constructed knowledge bases to enrich the semantic content of existing word vector collections. Often this is done as a post-processing step, where the distributional word vectors are refined to satisfy constraints extracted from a lexical resource such as WordNet or the Paraphrase Database (Faruqui et al., 2015; Mrkšić et al., 2016; Wieting et al., 2015). Throughout this thesis, this approach is termed *semantic specialisation*.

The work presented in this chapter advances the semantic specialisation paradigm in a number of ways. A novel algorithm for semantic specialisation, ATTRACT-REPEL, is presented first. Similar to counter-fitting, ATTRACT-REPEL uses synonymy and antonymy

en_morning			en_carpet			en_woman		
Slavic+EN	**Germanic**	**Romance+EN**	**Slavic+EN**	**Germanic**	**Romance+EN**	**Slavic + EN**	**Germanic**	**Romance+EN**
en_daybreak	de_vormittag	pt_madrugada	en_rug	de_teppichboden	en_rug	ru_женщина	de_frauen	fr_femme
en_morn	nl_krieken	it_mattina	bg_килим	nl_tapijten	it_moquette	bg_жените	sv_kvinnliga	en_womanish
bg_разсъмване	en_dawn	en_dawn	ru_ковролин	en_rug	it_tappeti	hr_žena	sv_kvinna	es_mujer
hr_svitanje	nl_zonsopkomst	pt_madrugadas	bg_килими	de_teppich	pt_tapete	en_womanish	sv_kvinnor	pt_mulher
hr_zore	sv_morgonen	es_madrugada	pl_dywany	en_carpeting	es_moqueta	bg_жена	de_weib	es_fémina
bg_изгрев	de_tagesanbruch	it_nascente	bg_мокет	de_teppiche	it_tappetino	pl_kobieta	en_womanish	en_womens
en_dawn	en_sunrise	en_morn	pl_dywanów	sv_mattor	en_carpeting	hr_treba	sv_kvinno	pt_feminina
ru_утро	nl_opgang	es_aurora	hr_tepih	sv_matta	pt_carpete	bg_жени	de_frauenzimmer	pt_femininas
bg_аврора	de_sonnenaufgang	fr_matin	pl_wykładziny	en_carpets	pt_tapetes	en_womens	sv_honkön	es_femina
hr_jutro	nl_dageraad	fr_aurora	ru_ковер	nl_tapijt	fr_moquette	pl_kobiet	sv_kvinnan	fr_femelle
ru_рассвет	de_anbruch	es_amaneceres	ru_коврик	nl_kleedje	en_carpets	hr_žene	nl_vrouw	pt_fêmea
hr_zora	sv_morgon	en_sunrises	hr_ćilim	nl_vloerbedekking	es_alfombra	pl_niewiasta	de_madam	fr_femmes
hr_zoru	en_daybreak	es_mañanero	en_carpeting	de_brücke	es_alfombras	hr_žensko	sv_kvinnligt	it_donne
pl_poranek	de_morgengrauen	fr_matinée	pl_dywan	de_matta	fr_tapis	hr_ženke	sv_gumman	es_mujeres
en_sunrise	nl_zonsopgang	it_mattinata	ru_ковров	nl_matta	pt_tapeçaria	pl_samica	sv_female	pt_fêmeas
bg_зазоряване	nl_goedemorgen	pt_amanhecer	en_carpets	en_mat	it_zerbino	ru_самка	sv_gumma	es_hembras
bg_сутрин	sv_gryningen	en_cockcrow	ru_килим	de_matte	it_tappeto	bg_женска	sv_kvinnlig	en_wife
en_sunrises	en_mornin	pt_aurora	en_mat	en_doilies	es_tapete	hr_ženka	sv_feminin	fr_nana
bg_зора	sv_gryning	pt_alvorecer	hr_sag	nl_mat	es_manta	ru_дама	en_wife	es_hembra

Figure 6.1 Nearest neighbours for three example words across Slavic, Germanic and Romance language groups (with English included as part of each word vector collection). Semantically dissimilar words have been underlined.

constraints drawn from lexical resources to tune word vector spaces using linguistic information that is difficult to capture with conventional distributional training. The presented evaluation shows that ATTRACT-REPEL outperforms previous methods which make use of similar lexical resources, achieving state-of-the-art results on two word similarity datasets: SimLex-999 (Hill et al., 2015) and SimVerb-3500 (Gerz et al., 2016).

Subsequently, ATTRACT-REPEL is deployed in a multilingual setting, using semantic relations extracted from BabelNet (Ehrmann et al., 2014; Navigli and Ponzetto, 2012), a cross-lingual lexical resource, to inject constraints between words of different languages into the word representations. The use of these constraints allows the procedure to embed vector spaces of multiple languages into a single vector space, exploiting information from high-resource languages to improve the word representations of lower-resource ones. Table 6.1 illustrates the effects of cross-lingual ATTRACT-REPEL specialisation by showing the nearest neighbours for three English words across three cross-lingual spaces. In each case, the vast majority of each words' neighbours are meaningful synonyms/translations.[1]

While there is a considerable amount of prior research on joint learning of cross-lingual vector spaces (see Section 6.2.2), to the best of my knowledge this work is the first to apply semantic specialisation to this problem.[2] The efficacy of this approach is demonstrated with state-of-the-art results on the four languages in the Multilingual SimLex-999 dataset (Leviant

[1]Some (negative) effects of the distributional hypothesis do persist. For example, *nl_krieken* (Dutch for *cherries*), is identified as a synonym for *en_morning*, presumably because the frequently used idiom *'het krieken van de dag'* translates to *'the crack of dawn'*.

[2]This approach is not suited for languages for which no lexical resources exist. However, many languages have some coverage in cross-lingual lexicons. For instance, BabelNet 3.7 automatically aligns WordNet to Wikipedia, providing accurate cross-lingual mappings between 271 languages.

and Reichart, 2015). In order to show that this approach yields semantically informative vectors for lower-resource languages, intrinsic evaluation datasets for Hebrew and Croatian are collected. Subsequent evaluation shows that cross-lingual specialisation significantly improves word vector quality even for these two (comparatively) low-resource languages.[3]

6.2 Related Work

6.2.1 Semantic Specialisation

The usefulness of distributional word representations has been demonstrated across many application areas: Part-of-Speech (POS) tagging (Collobert et al., 2011), machine translation (Devlin et al., 2014; Zou et al., 2013), dependency and semantic parsing (Ammar et al., 2016; Bansal et al., 2014; Chen and Manning, 2014; Johannsen et al., 2015; Socher et al., 2013a), sentiment analysis (Socher et al., 2013b), named entity recognition (Guo et al., 2014; Turian et al., 2010), and many others. The importance of semantic specialisation for downstream tasks is relatively unexplored, with improvements in performance so far observed for dialogue state tracking (Mrkšić et al., 2016, 2017a), spoken language understanding (Kim et al., 2016a,b) and judging lexical entailment (Vulić et al., 2016).

Semantic specialisation methods (broadly) fall into two categories: **a)** those which train distributed representations 'from scratch' by combining distributional knowledge and lexical information during training; and **b)** those which *inject* lexical information into pre-trained collections of word vectors. Methods from both categories make use of similar lexical resources. Common examples of these include WordNet (Miller, 1995), FrameNet (Baker et al., 1998) or the Paraphrase Databases (PPDB) (Ganitkevitch and Callison-Burch, 2014; Ganitkevitch et al., 2013; Pavlick et al., 2015).

Learning from Scratch Some methods modify the prior or the regularisation of the original training procedure using the set of linguistic constraints (Aletras and Stevenson, 2015; Bian et al., 2014; Kiela et al., 2015; Xu et al., 2014; Yu and Dredze, 2014). Other ones modify the skip-gram (Mikolov et al., 2013b) objective function by introducing semantic constraints (Liu et al., 2015; Yih et al., 2012) to train word vectors which emphasise word similarity over relatedness. Osborne et al. (2016) propose a method for incorporating prior knowledge into the Canonical Correlation Analysis (CCA) method used by Dhillon et al. (2015) to learn spectral word embeddings. While such methods introduce semantic similarity constraints

[3] All resources relating to this chapter are available at: *www.github.com/nmrksic/attract-repel*. These include: **1)** the ATTRACT-REPEL source code; **2)** bilingual word vector collections combining English with 51 other languages; **3)** Hebrew and Croatian intrinsic evaluation datasets collected for this work.

extracted from lexicons, approaches such as the one proposed by Schwartz et al. (2015) use *symmetric patterns* (Davidov and Rappoport, 2006) to push away antonymous words in their pattern-based vector space. Ono et al. (2015) combine both approaches, using thesauri and distributional data to train embeddings specialised for capturing antonymy. Faruqui and Dyer (2015) use many different lexicons to create interpretable sparse binary vectors which achieve competitive performance across a range of intrinsic evaluation tasks.

In theory, word representations produced by models which consider distributional and lexical information jointly could be as good (or even better) than representations produced by post-hoc fine-tuning of distributional word vectors. However, their performance has not surpassed that of fine-tuning methods. The SimLex-999 dataset web page lists models with state-of-the-art performance, none of which learn representations jointly.[4]

Fine-Tuning Pre-trained Vectors Rothe and Schütze (2015) fine-tune word vector spaces to improve the representations of synsets/lexemes found in WordNet. Faruqui et al. (2015) and Jauhar et al. (2015) use synonymy constraints in a procedure termed *retrofitting* to bring the vectors of semantically similar words close together, while Wieting et al. (2015) modify the skip-gram objective function to fine-tune word vectors by injecting paraphrasing constraints from PPDB. The counter-fitting procedure, presented in Chapter 4, builds on the retrofitting approach by jointly injecting synonymy and antonymy constraints (Mrkšić et al., 2016); the same idea is reassessed by Nguyen et al. (2016). Kim et al. (2016a) further expand this line of work by incorporating *semantic intensity* information for the constraints, while Recski et al. (2016) use ensembles of rich *concept dictionaries* to further improve a combined collection of semantically specialised word vectors.

ATTRACT-REPEL belongs to the second family of models, providing a portable, light-weight approach for injecting external knowledge into arbitrary vector spaces. ATTRACT-REPEL outperforms previously proposed post-processors, setting the new state-of-art performance on the SimLex-999 word similarity dataset. Moreover, starting from distributional vectors allows ATTRACT-REPEL to use existing cross-lingual resources to tie distributional vector spaces of different languages into a unified vector space which benefits from positive semantic transfer between its constituent languages.

6.2.2 Cross-Lingual Word Representations

Most existing models which induce cross-lingual word representations rely on cross-lingual distributional information (Huang et al., 2015; Klementiev et al., 2012; Soyer et al., 2015; Zou et al., 2013, inter alia). These models differ in the cross-lingual signal/supervision they

[4] *www.cl.cam.ac.uk/ fh295/simlex.html*

use to tie languages into unified bilingual vector spaces. Some models learn on the basis of parallel word-aligned data (Coulmance et al., 2015; Luong et al., 2015a) or sentence-aligned data (Chandar et al., 2014; Gouws et al., 2015; Hermann and Blunsom, 2014a,b). Other ones require document-aligned data (Søgaard et al., 2015; Vulić and Moens, 2016), while some learn on the basis of available bilingual dictionaries (Duong et al., 2016; Faruqui and Dyer, 2014; Lazaridou et al., 2015; Mikolov et al., 2013a; Vulić and Korhonen, 2016b).

The inclusion of cross-lingual information results in shared cross-lingual vector spaces which can: **a)** boost performance on monolingual tasks such as word similarity (Faruqui and Dyer, 2014; Rastogi et al., 2015; Upadhyay et al., 2016); and **b)** support cross-lingual tasks such as bilingual lexicon induction (Duong et al., 2016; Gouws et al., 2015; Mikolov et al., 2013a), cross-lingual information retrieval (Mitra et al., 2016; Vulić and Moens, 2015), and transfer learning for resource-lean languages (Guo et al., 2015; Søgaard et al., 2015).

However, prior work on cross-lingual word embedding has tended not to exploit pre-existing linguistic resources such as BabelNet. In this chapter, cross-lingual constraints derived from such repositories are used to induce high-quality cross-lingual vector spaces by facilitating semantic transfer from high- to lower-resource languages. The presented experiments show that cross-lingual vector spaces produced by ATTRACT-REPEL consistently outperform a representative selection of five strong cross-lingual word embedding models for both intrinsic and (in the subsequent chapter) extrinsic evaluation across several languages.

6.3 The ATTRACT-REPEL Model

The ATTRACT-REPEL procedure builds on the Paragram (Wieting et al., 2015) and counter-fitting procedures (Mrkšić et al., 2016), both of which inject linguistic constraints into existing vector spaces to improve their ability to capture semantic similarity.

Let V be the vocabulary, S the set of synonymous word pairs (e.g., *intelligent* and *brilliant*), and A the set of antonymous word pairs (e.g., *vacant* and *occupied*). The optimisation procedure operates over mini-batches of synonym and antonym pairs B_S and B_A (which list k_1 synonym and k_2 antonym pairs). For ease of notation, let each word pair (x_l, x_r) in these two sets correspond to a vector pair $(\mathbf{x}_l, \mathbf{x}_r)$, so that a mini-batch is given by $B_S = [(\mathbf{x}_l^1, \mathbf{x}_r^1), \ldots, (\mathbf{x}_l^{k_1}, \mathbf{x}_r^{k_1})]$ (similarly for B_A).

Next, let $T_S = [(\mathbf{t}_l^1, \mathbf{t}_r^1), \ldots, (\mathbf{t}_l^{k_1}, \mathbf{t}_r^{k_1})]$ and $T_A = [(\mathbf{t}_l^1, \mathbf{t}_r^1), \ldots, (\mathbf{t}_l^{k_2}, \mathbf{t}_r^{k_2})]$ be the pairs of *negative examples* for each synonymy and antonymy example pair in mini-batches B_S and B_A. These negative examples are chosen from the word vectors present in B_S or B_A so that:

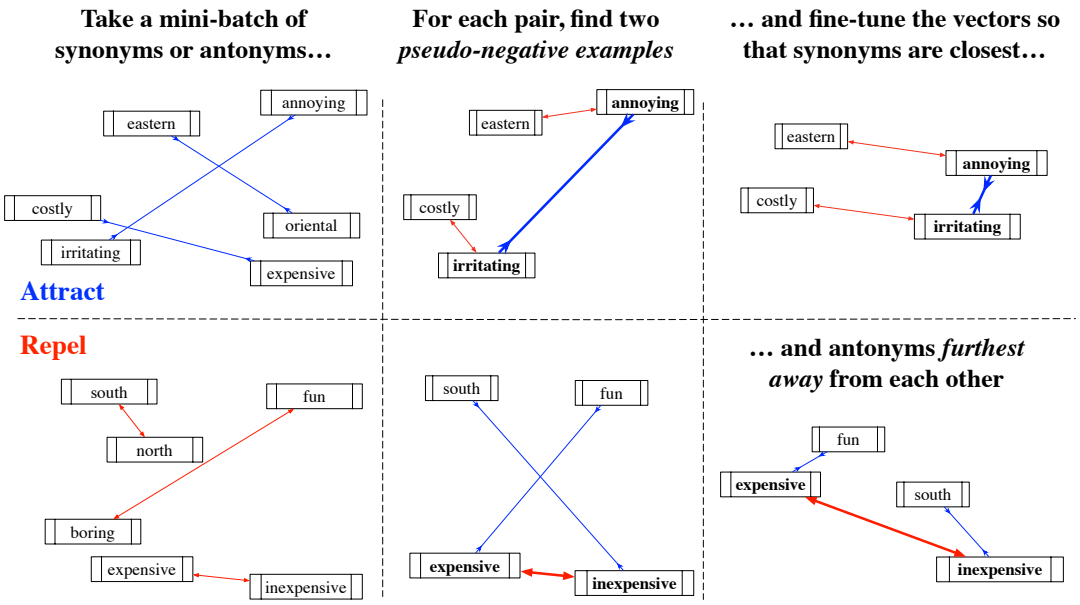


Figure 6.2 ATTRACT-REPEL procedure acting on two mini-batches of synonymous and antonymous word pairs (mini-batch size is 3 in this animation, but 50 in the experiments).

- For each synonymy pair $(\mathbf{x}_l, \mathbf{x}_r)$, the negative example pair $(\mathbf{t}_l, \mathbf{t}_r)$ is chosen from the remaining in-batch vectors so that $\mathbf{t}_l$ is the one closest (cosine similarity) to $\mathbf{x}_l$ and $\mathbf{t}_r$ is the in-batch word vector closest to $\mathbf{x}_r$.
- For each antonymy pair $(\mathbf{x}_l, \mathbf{x}_r)$, the negative example pair $(\mathbf{t}_l, \mathbf{t}_r)$ is chosen from the remaining in-batch vectors so that $\mathbf{t}_l$ is the one furthest away from $\mathbf{x}_l$ and $\mathbf{t}_r$ is the in-batch word vector furthest away from $\mathbf{x}_r$.

These negative examples are used to: **a)** force synonymous pairs to be closer to each other than to their respective negative examples; and **b)** to force antonymous pairs to be further away from each other than from their negative examples. Figure 6.2 provides a geometric animation of the ATTRACT-REPEL procedure. The first term of the cost function pulls synonymous words together:

$$S(B_S, T_S) = \sum_{i=1}^{k_1} \left[\tau\left(\delta_{syn} + \mathbf{x}_l^i \mathbf{t}_l^i - \mathbf{x}_l^i \mathbf{x}_r^i\right) + \tau\left(\delta_{syn} + \mathbf{x}_r^i \mathbf{t}_r^i - \mathbf{x}_l^i \mathbf{x}_r^i\right) \right] \tag{6.1}$$

where $\tau(x) = \max(0, x)$ is the hinge loss function and δ_{syn} is the similarity margin which determines how much closer synonymous vectors should be to each other than to their respective negative examples. The second part of the cost function pushes antonymous word

Specialisation Method	Attract Term	Repel Term	Regularisation
RETROFITTING	Geometric	None	L2
COUNTER-FITTING	Geometric	Geometric	Pairwise
PARAGRAM	Context Sensitive	None	L2
ATTRACT-REPEL	Context Sensitive	Context Sensitive	L2

Table 6.1 Theoretical comparison of the four state-of-the-art vector specialisation methods discussed in this chapter. Retrofitting and counter-fitting use geometric Attract and Repel terms which update word vector pairs without considering their relation to other words. Conversely, PARAGRAM and ATTRACT-REPEL implement more sophisticated fine-tuning which considers the relation of each word to other vectors in its neighbourhood. For regularisation, counter-fitting uses computationally expensive pairwise terms to preserve each word's neighbourhood. The other three methods use (computationally efficient) L2 regularisation to pull each word towards its initial distributional word vector.

pairs away from each other:

$$A(B_A, T_A) = \sum_{i=1}^{k_2} \left[\tau\left(\delta_{ant} + \mathbf{x}_l^i \mathbf{x}_r^i - \mathbf{x}_l^i \mathbf{t}_l^i\right) + \tau\left(\delta_{ant} + \mathbf{x}_l^i \mathbf{x}_r^i - \mathbf{x}_r^i \mathbf{t}_r^i\right) \right] \tag{6.2}$$

In addition to these two terms, an additional regularisation term is used to *preserve* the abundance of high-quality semantic content present in the initial (distributional) vector space, as long as this information does not contradict the injected linguistic constraints. If $V(B)$ is the set of all word vectors present in the given mini-batch, then:

$$R(B_S, B_A) = \sum_{\mathbf{x}_i \in V(B_S \cup B_A)} \lambda_{reg} \left\| \widehat{\mathbf{x}}_i - \mathbf{x}_i \right\|_2 \tag{6.3}$$

where λ_{reg} is the L2 regularisation constant and $\widehat{\mathbf{x}}_i$ denotes the original (distributional) word vector for word x_i. The full cost function is given by the sum of all three terms:

$$C(B_S, T_S, B_A, T_A) = S(B_S, T_S) + A(B_A, T_A) + R(B_S, B_A) \tag{6.4}$$

Comparison to Prior Work ATTRACT-REPEL draws inspiration from three methods: **1)** retrofitting (Faruqui et al., 2015); **2)** PARAGRAM (Wieting et al., 2015); and **3)** counter-fitting (Mrkšić et al., 2016). Table 6.1 summarises the main differences between these specialisation methods and ATTRACT-REPEL. Whereas retrofitting and PARAGRAM do not consider antonymy, counter-fitting models both synonymy and antonymy. ATTRACT-REPEL differs from this method in two important ways:

1. **Context-Sensitive Updates:** Counter-fitting uses attract and repel terms which pull synonyms together and push antonyms apart without considering their relation to other word vectors. For example, the method's attract term is given by:

$$Attract(S) = \sum_{(\mathbf{x}_l,\mathbf{x}_r)\in S} \tau(\delta_{syn} - \mathbf{x}_l\mathbf{x}_r) \tag{6.5}$$

   where S is the set of synonyms and δ_{syn} is the synonymy margin. Conversely, ATTRACT-REPEL fine-tunes vector spaces by operating over mini-batches of example pairs, updating word vectors only if the position of their negative example implies a stronger semantic relation than that expressed by the position of its target example. Importantly, ATTRACT-REPEL makes fine-grained updates to both the example pair *and* the negative examples, rather than updating the example word pair but ignoring how this affects its relation to all other word vectors.

2. **Regularisation:** Counter-fitting preserves distances between pairs of word vectors in the initial vector space, trying to *pull* the words' neighbourhoods with them as they move to incorporate external knowledge. The radius of this initial neighbourhood introduces an opaque hyperparameter to the procedure. Conversely, ATTRACT-REPEL implements standard L2 regularisation, which pulls each word's vector towards its initial distributional word vector.

The subsequent evaluation provides a thorough comparison of these methods, showing that ATTRACT-REPEL outperforms counter-fitting in both mono- and cross-lingual setups.

Optimisation ATTRACT-REPEL treats the word embeddings for all words in the vocabulary like the first layer of a neural network, backpropagating into them to optimise the cost function defined in Equations 6.1 - 6.4. Following Wieting et al. (2015), the AdaGrad algorithm (Duchi et al., 2011) is used to train word vectors for five epochs, which suffices for the magnitude of the parameter updates to converge.[5]

Early Stopping Similar to Faruqui et al. (2015), Wieting et al. (2015) and Mrkšić et al. (2016), the training procedure does not rely on early stopping. By not making use of language-

[5]The use of alternative optimisation algorithms (i.e., Adam (Kingma and Ba, 2015) or Adadelta (Zeiler, 2012)) resulted in substantially weaker performance across all experiments.

specific validation sets, the ATTRACT-REPEL procedure can induce semantically specialised word vectors for languages with no intrinsic evaluation datasets.[6]

Hyperparameter Tuning To optimise model hyperparameters, ATTRACT-REPEL uses Spearman's correlation of the final word vectors with the Multilingual WordSim-353 gold-standard association dataset (Leviant and Reichart, 2015). The ATTRACT-REPEL procedure has six hyperparameters: the regularisation constant λ_{reg}, the similarity and antonymy margins δ_{sim} and δ_{ant}, mini-batch sizes k_1 and k_2, and the size of the PPDB constraint set used for each language (larger sizes include more constraints, but also a larger proportion of false synonyms). These parameters were tuned separately for each of the four SimLex languages, choosing the hyperparameters which achieved the best final WordSim-353 score.[7]

6.4 Experimental Setup

6.4.1 Distributional Vectors

Sixteen experimental languages are used for evaluation: English (EN), German (DE), Italian (IT), Russian (RU), Dutch (NL), Swedish (SV), French (FR), Spanish (ES), Portuguese (PT), Polish (PL), Bulgarian (BG), Croatian (HR), Irish (GA), Persian (FA) and Vietnamese (VI). The first four languages are those of the Multilingual SimLex-999 dataset.

For the SimLex languages, four well-known, high-quality word vector collections are used in the experiments: **a)** The Common Crawl GloVe English vectors from Pennington et al. (2014); **b)** German vectors from Vulić and Korhonen (2016a); **c)** Italian vectors from Dinu et al. (2015); and **d)** Russian vectors from Kutuzov and Andreev (2015). Additionally, for each of the 16 languages, the skip-gram with negative sampling variant of the `word2vec` model (Mikolov et al., 2013b) was trained on the latest Wikipedia dump of each language, inducing 300-dimensional word vectors.[8]

[6] Many languages are present in semi-automatically constructed lexicons such as BabelNet or PPDB (see the discussion in Section 6.4). However, intrinsic evaluation datasets such as SimLex-999 exist for very few languages, as they require expert translators and skilled annotators.

[7] The grid search was run over $\lambda_{reg} \in [10^{-3}, \ldots, 10^{-10}]$, $\delta_{sim}, \delta_{ant} \in [0, 0.1, \ldots, 1.0]$, $k_1, k_2 \in [10, 25, 50, 100, 200]$ and over the six PPDB sizes for the four SimLex languages. $\lambda_{reg} = 10^{-9}$, $\delta_{sim} = 0.6$, $\delta_{ant} = 0.0$ and $k_1 = k_2 \in [10, 25, 50]$ consistently achieved the best performance ($k_1 = k_2 = 50$ is used across all experiments for consistency). The PPDB constraint set size *XL* was best for English, German and Italian, and *M* achieved the best performance for Russian.

[8] The frequency cut-off was set to 50: words that occurred less frequently were removed from the vocabularies. Other `word2vec` parameters were set to the standard values (Vulić and Korhonen, 2016a): 15 epochs, 15 negative samples, global (decreasing) learning rate: 0.025, subsampling rate: $1e-4$.

	English		German		Italian		Russian	
	syn	ant	syn	ant	syn	ant	syn	ant
English	640	5	246	11	356	24	196	9
German	-	-	135	2	277	13	175	6
Italian	-	-	-	-	159	7	220	11
Russian	-	-	-	-	-	-	48	1

Table 6.2 Linguistic constraint counts (in thousands). For each language pair, the two figures show the number of injected synonymy and antonymy constraints. Monolingual constraints (the diagonal elements) are underlined.

Multi-Sense Embeddings Compounding different word senses into a single word vector can adversely impact the models making use of these word vector collections for various downstream tasks (Li and Jurafsky, 2015). However, adding an additional word sense disambiguation step would introduce more uncertainty into the spoken dialogue system pipeline. Consequently, all word vector collections used in this thesis assign each word in the vocabulary a single vectorial representation.

6.4.2 Linguistic Constraints

Table 6.2 shows the number of monolingual and cross-lingual constraints for the four SimLex languages extracted for the vocabularies of the four well-known word vector collections. The sources of these constraints are discussed next.

Monolingual Similarity The Multilingual Paraphrase Database (Ganitkevitch and Callison-Burch, 2014) is used as the source of these constraints. This resource contains paraphrases automatically extracted from parallel-aligned corpora for ten of the sixteen experimental languages. The remaining six languages (HE, HR, SV, GA, VI, FA) serve as examples of *lower-resource* languages, as they have no monolingual synonymy constraints.

Cross-Lingual Similarity BabelNet is used as the source of all cross-lingual constraints in the experiments. BabelNet is a multilingual semantic network automatically constructed by linking Wikipedia to WordNet (Ehrmann et al., 2014; Navigli and Ponzetto, 2012). It groups words from different languages into *Babel synsets*. Two words from any (distinct) language pair are considered synonymous if they belong to (at least) one set of synonymous Babel synsets. BabelNet word senses tagged as *conceptual* were used, but those tagged as *Named Entities* were ignored across the presented experiments.

Given a large collection of *cross-lingual* semantic constraints (e.g., the translation pair *en_sweet* and *it_dolce*), ATTRACT-REPEL can use them to bring the vector spaces of different

languages together into a shared cross-lingual space. Ideally, sharing information across languages should lead to improved semantic content for each language, especially for those with limited monolingual resources.

Antonymy BabelNet is used to extract both mono- and cross-lingual antonymy constraints. Following Faruqui et al. (2015), who found PPDB constraints more beneficial than those from WordNet, BabelNet is not used as a source of monolingual synonymy constraints.

Availability of Resources Both PPDB and BabelNet are created automatically. However, PPDB relies on large, high-quality parallel corpora such as Europarl (Koehn, 2005). In total, Multilingual PPDB provides collections of paraphrases for 22 languages. On the other hand, BabelNet uses Wikipedia's *inter-language links* and statistical machine translation (Google Translate) to provide cross-lingual mappings for 271 languages. The presented experiments investigate whether PPDB and BabelNet can be used jointly to improve word representations for lower-resource languages by tying them into bilingual spaces with high-resource ones. This claim is validated on Hebrew and Croatian, which act as 'lower-resource' languages because of their lack of any PPDB resource and their relatively small Wikipedia sizes.[9]

6.5 Evaluation

6.5.1 Datasets

Spearman's rank correlation with SimLex-999 (Hill et al., 2015) acts as the intrinsic evaluation metric throughout the experiments. Unlike other gold standard resources such as WordSim-353 (Finkelstein et al., 2002) or MEN (Bruni et al., 2014), SimLex contains word pairs scored by annotators instructed to discern between semantic similarity and conceptual association, so that related but non-similar words (e.g., *book* and *read*) have a low rating. Spearman's correlation with SimVerb-3500 (Gerz et al., 2016) is reported as well. This is a novel semantic similarity dataset which focuses on verb pair similarity.

Multilingual SimLex-999 Leviant and Reichart (2015) translated SimLex-999 to German, Italian and Russian, crowd-sourcing the similarity scores from native speakers of these languages. This is the principle resource used for multilingual intrinsic evaluation in this

[9]Hebrew and Croatian Wikipedias (which are used to induce their BabelNet constraints) currently consist of 203,867 / 172,824 articles, ranking them 40th / 42nd by size.

chapter.[10] To further investigate the portability of this approach to lower-resource languages, the same experimental setup was used to collect SimLex-999 datasets for Hebrew and Croatian (Mrkšić et al., 2017b). The 999 word pairs and the annotator instructions were translated into Hebrew and Croatian by native speakers and then scored by 10 (native) annotators. The inter-annotator agreement scores (Spearman's rank correlation) were 0.77 (pairwise) and 0.87 (mean) for Croatian, and 0.59 (pairwise) and 0.71 (mean) for Hebrew.[11,12]

6.5.2 Experiments

Monolingual and Cross-Lingual Specialisation Starting from the four distributional vectors for the SimLex languages, semantic specialisation is performed using: **a)** monolingual synonyms; **b)** monolingual antonyms; and **c)** the combination of both. Cross-lingual synonyms and antonyms are next added to these constraints and used to induce a shared four-lingual vector space for these languages.

Comparison to Baseline Methods Both mono- and cross-lingual specialisation was performed using ATTRACT-REPEL and counter-fitting, in order to conclusively determine which of the two methods exhibits superior performance. Retrofitting and PARAGRAM methods only inject synonymy, and their cost functions can be expressed using sub-components of counter-fitting and ATTRACT-REPEL cost functions. As such, the performance of the two investigated methods when they make use of similarity (but not antonymy) constraints illustrates the performance range of the two preceding models.

Importance of Initial Vectors Three different sets of initial word vectors are used to assess the importance of the starting point for the specialisation procedure. These are: **a)** the four well-known distributional word vector collections for the original SimLex languages; **b)** distributional vectors trained on the latest Wikipedia dumps; and **c)** word vectors randomly initialised using the XAVIER initialisation (Glorot and Bengio, 2010).[13]

[10] Leviant and Reichart (2015) also re-scored the original English SimLex. Results on their version are reported in this chapter, but numbers for the original dataset are also reported for comparability.

[11] The Hebrew and Croatian SimLex-999 datasets are available at: *www.github.com/nmrksic/attract-repel*

[12] Homonyms (different words with the same spelling) occur frequently in modern Hebrew because vowels are omitted in writing. This is the likely reason for the lower inter-annotator scores for this language.

[13] The XAVIER initialisation populates the values for each word vector by uniformly sampling from the interval $[-\frac{\sqrt{6}}{\sqrt{d}}, +\frac{\sqrt{6}}{\sqrt{d}}]$, where d is the vector dimensionality. This is a typical initialisation method in neural network research (Bengio et al., 2013; Goldberg, 2015).

Specialisation for Lower-Resource Languages In this experiment, bilingual vector spaces are constructed by starting from distributional vector spaces for two languages and using linguistic constraints to tie them into a shared cross-lingual vector space. The languages combined are: **a)** one of the four SimLex languages; with **b)** each of the other twelve languages.[14] Since each pair contains at least one SimLex language, one can analyse the improvement over monolingual specialisation to understand how robust the performance gains are across different language pairs. Finally, the newly collected SimLex datasets for Hebrew and Croatian are used to evaluate the extent to which bilingual semantic specialisation using ATTRACT-REPEL and BabelNet constraints can improve word representations for lower-resource languages.

Comparison to State-of-the-Art Bilingual Spaces The English-Italian and English-German bilingual spaces induced by ATTRACT-REPEL are compared to five state-of-the-art methods for constructing bilingual vector spaces: **1.** (Mikolov et al., 2013a), re-trained using the constraints used by ATTRACT-REPEL; and **2.-5.** (Gouws et al., 2015; Hermann and Blunsom, 2014a; Vulić and Korhonen, 2016a; Vulić and Moens, 2016). The latter models use various sources of supervision (word-, sentence- and document-aligned corpora), which means they cannot be trained using the same set of constraints. For these models, competitive setups proposed in Vulić and Korhonen (2016a) are replicated. The goal of this experiment is to show that vector spaces induced by ATTRACT-REPEL exhibit better intrinsic and extrinsic performance when deployed in language understanding tasks.

6.5.3 Results and Discussion

Table 6.3 shows the effects of monolingual and cross-lingual semantic specialisation of four well-known distributional vector spaces for the SimLex languages. Monolingual specialisation leads to very strong improvements in the SimLex performance across all languages. Cross-lingual specialisation brings further improvements, with all languages benefiting from sharing the cross-lingual vector space. German and Italian in particular show strong evidence of effective transfer (+0.19 / +0.11 over monolingual specialisation), with Italian vectors' performance coming close to the top-performing English ones.

Comparison to Baselines Table 6.3 gives an exhaustive comparison of ATTRACT-REPEL to counter-fitting: ATTRACT-REPEL achieved substantially stronger performance in all experiments. I believe these results conclusively show that the fine-grained updates and

[14] ATTRACT-REPEL hyperparameters used are: $\delta_{sim} = 0.6$, $\delta_{ant} = 0.0$ and $\lambda_{reg} = 10^{-9}$, which achieved the best performance when tuned for the original SimLex languages. The largest available PPDB size was used for the six languages with available PPDB (French, Spanish, Portuguese, Polish, Bulgarian and Dutch).

Word Vectors	English	German	Italian	Russian
Monolingual Distributional Vectors	0.32	0.28	0.36	0.38
COUNTER-FITTING: Mono-Syn	0.45	0.24	0.29	0.46
COUNTER-FITTING: Mono-Ant	0.33	0.28	0.47	0.42
COUNTER-FITTING: Mono-Syn + Mono-Ant	0.50	0.26	0.35	0.49
COUNTER-FITTING: Cross-Syn	0.46	0.43	0.45	0.37
COUNTER-FITTING: Mono-Syn + Cross-Syn	0.47	0.40	0.43	0.45
COUNTER-FITTING: Mono-Syn + Mono-Ant + Cross-Syn + Cross-Ant	0.53	0.41	0.49	0.48
ATTRACT-REPEL: Mono-Syn	0.56	0.40	0.46	0.53
ATTRACT-REPEL: Mono-Ant	0.42	0.30	0.45	0.41
ATTRACT-REPEL: Mono-Syn + Mono-Ant	0.65	0.43	0.56	0.56
ATTRACT-REPEL: Cross-Syn	0.57	0.53	0.58	0.46
ATTRACT-REPEL: Mono-Syn + Cross-Syn	0.61	0.58	0.59	0.54
ATTRACT-REPEL: Mono-Syn + Mono-Ant + Cross-Syn + Cross-Ant	**0.71**	**0.62**	**0.67**	**0.61**

Table 6.3 Multilingual SimLex-999. The effect of using the COUNTER-FITTING and ATTRACT-REPEL procedures to inject mono- and cross-lingual synonymy and antonymy constraints into the four collections of distributional word vectors. The best results set the new state-of-the-art performance for all four languages.

L2 regularisation employed by ATTRACT-REPEL present a better alternative to the context-insensitive attract/repel terms and pair-wise regularisation employed by counter-fitting.

State-of-the-Art Wieting et al. (2016) note that the hyperparameters of the widely used Paragram-SL999 vectors (Wieting et al., 2015) are tuned on SimLex-999, and as such are not comparable to methods which hold out the dataset. This implies that further work which uses these vectors (e.g., (Mrkšić et al., 2016; Recski et al., 2016)) as the starting point does not yield meaningful high scores either. The English score of 0.71 on the Multilingual SimLex-999 reported here corresponds to **0.751** on the original SimLex-999. As such, it outperforms the 0.706 score reported by Wieting et al. (2016) and sets a new high score for this dataset. Similarly, the SimVerb-3500 score of these vectors is **0.674**, outperforming the current state-of-the-art score of 0.628 reported by Gerz et al. (2016).

Word Vectors	EN	DE	IT	RU
Random Initialisation (No Information)	0.01	-0.03	0.02	-0.03
ATTRACT-REPEL: Monolingual Constraints	0.54	0.33	0.29	0.35
ATTRACT-REPEL: Mono + Cross-Lingual Constraints	0.66	0.49	0.59	0.51
Distributional (Wikipedia) Vectors	0.32	0.31	0.28	0.19
ATTRACT-REPEL: Monolingual Constraints	0.61	0.48	0.53	0.52
ATTRACT-REPEL: Mono + Cross-Lingual Constraints	0.66	0.60	0.65	0.54

Table 6.4 Multilingual SimLex-999. The effect of injecting linguistic constraints into: **1)** random vectors initialised using the XAVIER initialisation (Glorot and Bengio, 2010); or **2)** distributional WORD2VEC vectors trained on the latest Wikipedia dumps.

	Mono.	SimLex Languages				PPDB available						No PPDB available					
	Spec.	**EN**	**DE**	**IT**	**RU**	**NL**	**FR**	**ES**	**PT**	**PL**	**BG**	**HR**	**HE**	**GA**	**VI**	**FA**	**SV**
English	0.65	-	0.69	0.70	0.70	0.70	0.72	0.72	0.70	0.70	0.68	0.70	0.66	0.65	0.67	0.68	0.70
German	0.43	0.61	-	0.58	0.56	0.55	0.60	0.59	0.56	0.54	0.52	0.53	0.50	0.49	0.48	0.51	0.55
Italian	0.56	0.69	0.65	-	0.64	0.67	0.68	0.68	0.66	0.66	0.62	0.63	0.59	0.60	0.58	0.61	0.63
Russian	0.56	0.63	0.59	0.62	-	0.61	0.61	0.62	0.58	0.60	0.61	0.59	0.56	0.57	0.58	0.58	0.60

Table 6.5 SimLex-999 performance. Tying the SimLex languages into bilingual vector spaces with 16 different languages. The first number in each row represents monolingual specialisation. All but two of the bilingual spaces improved over these baselines. The EN-FR vectors set a new high score of **0.754** on the original (English) SimLex-999.

Starting Distributional Spaces Table 6.4 repeats the previous experiment with two different sets of initial vector spaces: **a)** randomly initialised word vectors; and **b)** skip-gram with negative sampling vectors trained on the latest Wikipedia dumps. The randomly initialised vectors serve to decouple the impact of injecting external knowledge from the information embedded in the distributional vectors. The random vectors benefit from both mono- and cross-lingual specialisation: the English performance is surprisingly strong, with other languages suffering more from the lack of initialisation.

When comparing distributional vectors trained on Wikipedia to the high-quality word vector collections used in Table 6.3, the Italian and Russian vectors in particular start from substantially weaker SimLex scores. The difference in performance is largely mitigated through semantic specialisation. However, all vector spaces still exhibit weaker performance compared to those in Table 6.3. This shows that the quality of initial distributional vectors is important, but can in large part be compensated for through semantic specialisation.

Bilingual Specialisation Table 6.5 shows the effect of combining the four original SimLex languages with each other and with twelve other languages. Bilingual specialisation substantially improves over monolingual specialisation for *all language pairs*. This indicates that the effect of positive semantic transfer is language independent to a large extent.

Interestingly, even though no monolingual synonymy constraints are used for the six right-most languages, combining them with the SimLex languages still improved word vector quality for these four high-resource languages. The reason why even resource-deprived languages such as Irish help improve vector space quality of high-resource ones such as English or Italian is that they provide implicit indicators of semantic similarity. English words which map to the same Irish word are likely to be synonyms, even if those English pairs are not present in the PPDB datasets (Faruqui and Dyer, 2014).

	Distributional Vectors	+ English	+ German	+ Italian	+ Russian
Hebrew	0.28	0.51	0.46	**0.52**	0.45
Croatian	0.21	**0.62**	0.49	0.58	0.54
English	0.32	-	0.61	**0.66**	0.63
German	0.28	**0.58**	-	0.55	0.49
Italian	0.36	**0.69**	0.66	-	0.63
Russian	0.38	**0.56**	0.52	0.55	-

Table 6.6 Bilingual semantic specialisation for: **a)** Hebrew and Croatian; and **b)** the original SimLex languages. Each row shows how SimLex scores for that language improve when its distributional vectors are tied into a bilingual space with the four high-resource languages.

Lower-Resource Languages The previous experiment showed that bilingual specialisation further improves the (already) high-quality estimates for high-resource languages. However, it does little to show how much (or if) the word vectors of lower-resource languages improve during bilingual specialisation. Table 6.6 investigates this proposition using the newly collected SimLex datasets for Hebrew and Croatian.

Tying the distributional vectors of these two languages (which have no monolingual synonymy constraints) into cross-lingual spaces with high-resource ones (which do, in this case from PPDB) leads to substantial improvements in SimLex performance. Table 6.6 also shows how the distributional vectors of the four SimLex languages improve when tied to other languages (in each row, monolingual constraints are used only for the 'added' high-resource language). Hebrew and Croatian exhibit similar trends to the original SimLex languages: tying to English and Italian leads to stronger gains than tying to the morphologically more complex languages like German and Russian.

Public Repository of Specialised Word Vectors Across all experiments in this chapter, tying other languages' word vector spaces to semantically specialised English word vectors resulted in strong SimLex-999 performance. This shows that bilingual ATTRACT-REPEL specialisation with English promises to produce high-quality vector spaces for many lower-resource languages which have coverage among the 271 BabelNet languages (but are not available in monolingual resources such as PPDB). To make such resources readily available, I produced bilingual vector space collections which combine English with 51 other languages: the 16 presented in this chapter and another 35 world languages. To produce these specialised bilingual word vector collections, cross-lingual constraints drawn from BabelNet were combined with monolingual English constraints from PPDB. These bilingual word vector spaces are publicly available at: *www.github.com/nmrksic/attract-repel*

Model	EN-IT		EN-DE	
	EN	IT	EN	DE
(Mikolov et al., 2013a)	0.32	0.28	0.32	0.28
(Hermann and Blunsom, 2014a)	0.40	0.34	0.38	0.35
(Gouws et al., 2015)	0.25	0.18	0.25	0.14
(Vulić and Korhonen, 2016a)	0.32	0.27	0.32	0.33
(Vulić and Moens, 2016)	0.23	0.25	0.20	0.25
Bilingual ATTRACT-REPEL	**0.70**	**0.69**	**0.69**	**0.61**

Table 6.7 Comparison of the intrinsic quality (SimLex-999) of bilingual spaces produced by the ATTRACT-REPEL method to those produced by five state-of-the-art methods for constructing bilingual vector spaces.

Existing Bilingual Spaces Table 6.7 compares the SimLex-999 performance of bilingual English-Italian and English-German vectors produced by ATTRACT-REPEL to five previously proposed approaches for constructing bilingual vector spaces. For both languages in both language pairs, ATTRACT-REPEL achieves substantial gains over all of these methods. The next chapter will investigate whether these differences in intrinsic performance lead to substantial gains in downstream dialogue state tracking evaluation.

6.6 Conclusion

This chapter presented a novel ATTRACT-REPEL method for injecting linguistic constraints into word vector spaces. The procedure *semantically specialises* word vectors by jointly injecting mono- *and* cross-lingual synonymy and antonymy constraints, creating unified cross-lingual vector spaces which achieve state-of-the-art performance on the well-established SimLex-999 dataset and its multilingual variants. ATTRACT-REPEL can also induce high-quality vectors for lower-resource languages by tying them into bilingual vector spaces with high-resource ones. The next chapter investigates whether the substantial gains in intrinsic performance translate to gains in the downstream task of dialogue state tracking.

Chapter 7

Belief Tracking across Languages

Language understanding methods for dialogue systems have traditionally focused on English as the *lingua franca* of the research community. Consequently, the problem of deploying existing spoken dialogue system frameworks to other languages is relatively underexplored. This chapter delves into this problem, investigating the important factors for applying research presented in previous chapters to two new languages: Italian and German. The first part of the chapter studies the interplay between semantic specialisation of word embeddings and downstream language understanding performance; this work was first presented in Mrkšić et al. (2017b). The second part of the chapter goes further, investigating the importance of modelling morphological phenomena for achieving robust performance in languages with complex morphology; this work was first presented in Vulić, Mrkšić, et al. (2017).

7.1 Motivation

An important motivation for training word vectors is to improve the lexical coverage of supervised models for language understanding tasks such as question answering (Iyyer et al., 2014) or textual entailment (Rocktäschel et al., 2016). In this chapter, *dialogue state tracking* (DST) acts as the extrinsic task for evaluating word vector quality. Dialogue State Tracking involves understanding the goals expressed by the user and updating the system's distribution over such goals as the conversation progresses and new information becomes available. As such, any model which relies on word embeddings as the building blocks of DST models depends on the quality of their embedded semantics to make correct decisions.[1]

[1] As shown in Chapter 4, representation models that do not distinguish between synonyms and antonyms can have grave implications in downstream language understanding applications such as dialogue state tracking. For instance, a user looking for *'an affordable Chinese restaurant in west Cambridge'* does not want a recommendation for *'an expensive Thai place in east Oxford'*.

English	Italian	German
PRICE RANGE	PREZZO	PREISKLASSE
cheap	economico	Billig
moderate	moderato	Mäßig
expensive	caro	Teuer
AREA	AREA	GEGEND
centre	centro	Zentrum
north	nord	Norden
west	ovest	Westen
south	sud	Süden
east	est	Osten
FOOD	CIBO	ESSEN
English	Inglese	Englisch
Italian	Italiano	Italiänisch
German	Tedesco	Deutsch
French	Francese	Französisch
...	...	...

Figure 7.1 Subsets of the Cambridge Restaurants domain ontology in three languages.

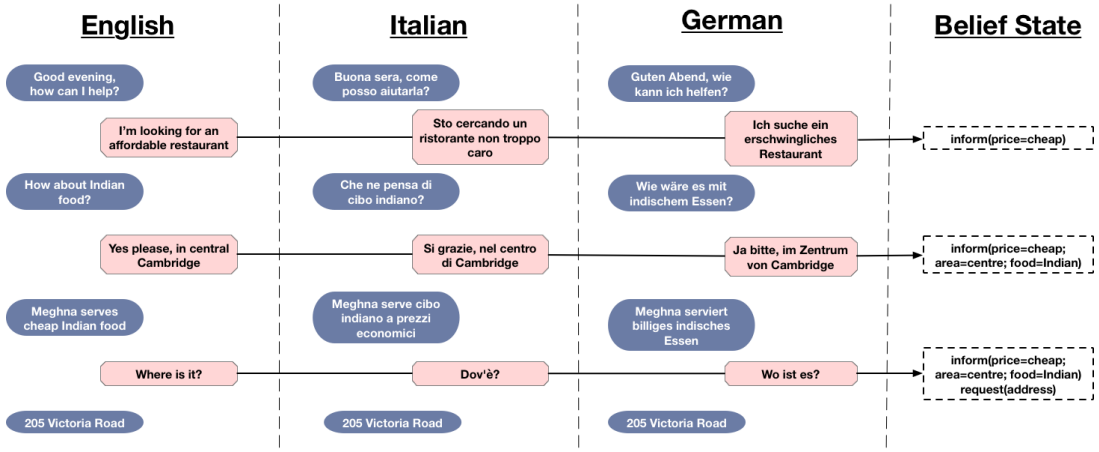


Figure 7.2 Example dialogues in three languages (EN, IT, DE), showing that belief states are independent of language, but instead grounded by the underlying domain ontology.

Language-Agnostic DST Model To detect intents in user utterances, most existing models rely on either (or both): **1)** Spoken Language Understanding models which require large amounts of annotated training data; or **2)** hand-crafted, domain-specific lexicons which try to capture lexical and morphological variation. The Neural Belief Tracker (NBT), presented in Chapter 5, overcomes both issues by reasoning purely over pre-trained word vectors (Mrkšić et al., 2017a). The NBT learns to compose these vectors into intermediate utterance and context representations. These are then used to decide which of the ontology-defined intents (goals) have been expressed by the user.

The NBT training procedure keeps the initial word vectors fixed. That way, at test time, unseen words semantically related to familiar slot values (i.e., *affordable* or *cheaper* to *cheap*) are recognized purely by their position in the original vector space. Thus, it is essential that deployed word vectors are specialized for semantic similarity, as distributional effects which keep antonymous words' vectors together can be very detrimental to DST performance (e.g., by matching *northern* to *south* or *inexpensive* to *expensive*). In this chapter, DST performance of the NBT model powered by different word vector collections is used as the principal measure of the vectors suitability for downstream language understanding tasks.

7.1.1 The Multilingual WOZ 2.0 Dataset

The downstream evaluation in this chapter is based on the WOZ 2.0 dataset collected by Wen et al. (2017) and Mrkšić et al. (2017a). The dataset is based on the same ontology used for the second Dialogue State Tracking Challenge (Henderson et al., 2014a). It consists of 1,200 Wizard-of-Oz (Fraser and Gilbert, 1991) dialogues in which Amazon Mechanical Turk users assumed the role of the dialogue system or the caller looking for restaurants in Cambridge, UK. Since users typed instead of using speech and interacted with intelligent assistants, the language they used was more sophisticated than in case of DSTC2, where users would quickly adapt to the system's inability to cope with complex queries.

For the experiments in this chapter, the ontology and all 1,200 dialogues were translated to Italian and German using *gengo.com*, a web-based human translation platform. Figures 7.1 and 7.2 show subsets of the translated ontology, as well as an example dialogue across three languages. The translation was performed by 24 professional translators, instructed to consider full dialogue context, rather than to translate specific utterances without considering the preceding dialogue turns. This means that the performance across languages is directly comparable, with the DST models' performance acting as an indicator of the models' (and the employed word vectors') suitability for each of the target languages.[2]

[2]The Italian and German DST datasets created for this work (henceforth refered to as the Multilingual WOZ 2.0 dataset) are available at *www.github.com/nmrksic/attract-repel*.

The evaluation metric used across all DST experiments is the *joint goal accuracy*, which represents the proportion of test set dialogue turns where all the search constraints expressed up to that point in the conversation were decoded correctly.

7.2 The Importance of Semantics

The work on *semantic specialisation* in Chapter 6 showed that large-scale semantic lexicons such as BabelNet can be used to craft monolingual or cross-lingual vectors of high semantic quality. As an intrinsic measure of semantic quality, previous experiments reported Spearman's average rank correlation with the SimLex-999 gold standard dataset. The experiments in this section investigate two different propositions:

1. **Intrinsic versus Downstream Performance** As mono- and cross-lingual semantic specialisation improves the semantic content of word vector collections according to SimLex-999, one could expect that the NBT model would perform higher-quality belief tracking when it makes use of the specialised word vectors. This experiment investigates the difference in DST performance for English, German and Italian when the NBT model employs the following word vector collections: **1)** distributional word vectors; **2)** monolingual semantically specialised vectors; and **3)** monolingual subspaces of the cross-lingual semantically specialised EN-DE-IT-RU vectors (presented in Chapter 6). For each language, the experiments compare the reported NBT performance to the performance the NBT model achieves using five state-of-the-art bilingual vector spaces used as baselines in Chapter 6 (Gouws et al., 2015; Hermann and Blunsom, 2014a; Mikolov et al., 2013a; Vulić and Korhonen, 2016a; Vulić and Moens, 2016).

2. **Training a Multilingual DST Model** The values expressed by the domain ontology (e.g., *cheap*, *north*, *Thai*, etc.) are language independent. If one assumes common semantic grounding across languages, the ontologies can be *decoupled* from the dialogue corpora, and a *single ontology* (i.e., its values' vector representations) can be used as the source of target labels across all languages. Since high-performing English DST is attainable, the Italian and German ontologies (i.e., all slot-value pairs in these ontologies) are replaced by the original English ontology (see Figure 7.2). The use of a single ontology coupled with cross-lingual vectors allows one to combine the training data for all languages and train a single NBT model capable of performing belief tracking across all three languages at once. Given a high-quality cross-lingual vector space, combining the languages effectively increases the training set size and should therefore lead to improved performance across all languages.

Word Vector Space	English	Italian	German
EN-IT/EN-DE (Mikolov et al., 2013a)	78.2	71.1	50.5
EN-IT/EN-DE (Hermann et al., 2014a)	71.7	69.3	44.7
EN-IT/EN-DE (Gouws et al., 2015)	75.0	68.4	45.4
EN-IT/EN-DE (Vulić and Korhonen, 2016a)	81.6	71.8	50.5
EN-IT/EN-DE (Vulić et al., 2016)	72.3	69.0	38.2
Monolingual Distributional Vectors	77.6	71.2	46.6
A-R: Monolingual Specialisation	80.9	72.7	52.4
A-R: Cross-Lingual Specialisation	80.3	75.3	55.7
Multilingual Model: English Ontology Grounding	**82.8**	**77.1**	**57.7**

Table 7.1 NBT model accuracy across three languages. Each figure shows the performance of the model trained using the subspace of the given vector space corresponding to the target language. For the English figures, the stronger of the EN-IT / EN-DE figures is shown.

7.2.1 Results and Discussion

Importance of Underlying Word Vectors The DST performance of the NBT model on English, German and Italian WOZ 2.0 datasets is shown in Table 7.1.[3] The first five rows show the performance when the model employs the five baseline word vector spaces. The subsequent three rows show the performance of: **a)** distributional word vectors; **b)** monolingual semantically specialised variants of these vectors; and **c)** their EN-DE-IT-RU cross-lingual specialisation. The last row shows the performance of the multilingual DST model. In this experiment, the training data for all three languages is combined and used to train a single model which uses cross-lingual vectors coupled with the English ontology to do belief tracking across all three languages. As can be seen in Table 7.1, the multilingual model improves over all previous monolingual models across all three languages.

The results in Table 7.1 show that both types of specialisation improve over DST performance achieved using the distributional vectors or the five baseline bilingual spaces. Interestingly, the bilingual vectors of Vulić and Korhonen (2016a) outperform the specialised vectors for English (but not Italian or German) despite their weaker SimLex performance, showing that intrinsic evaluation does not capture all relevant aspects pertaining to word vectors' usability for downstream tasks.

Bootstrapping DST Models Figure 7.3 investigates the usefulness of multilingual training for bootstrapping DST models for new languages with less data: the two figures display the

[3] All DST experiments in this chapter use an NBT-CNN model with the rule-based belief state update. As seen in Chapter 5 (Section 5.4.2), the performance of the statistical belief state update shows substantial variance across datasets. To mitigate this factor and better understand the impact of the underlying word vectors on language understanding performance, all experiments in this chapter use the rule-based belief state update.

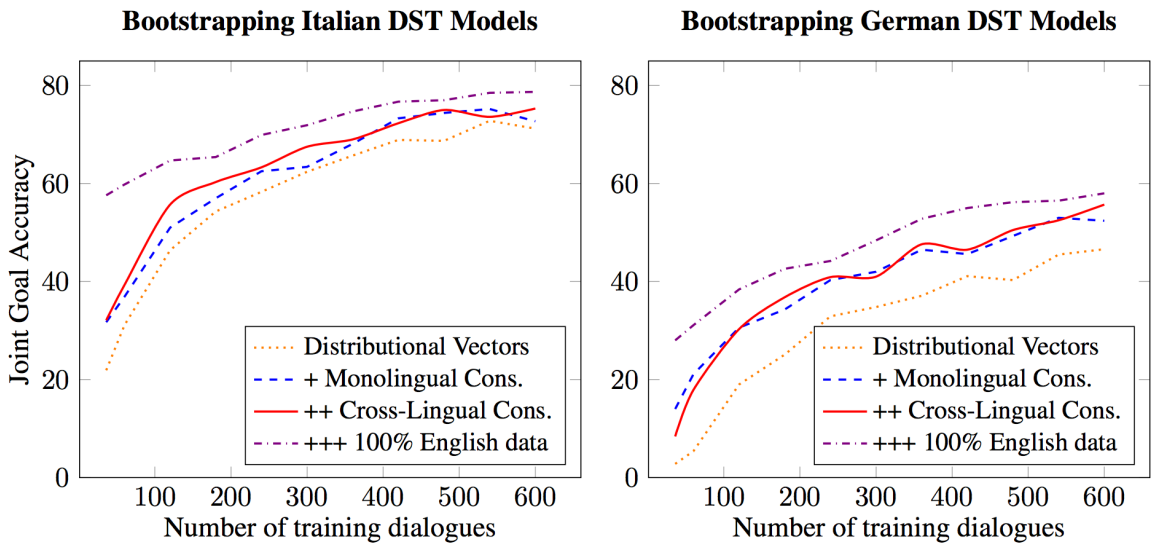


Figure 7.3 Joint goal accuracy of the NBT-CNN model for Italian (left) and German (right) WOZ 2.0 test sets as a function of the number of in-language dialogues used for training.

Italian / German performance of models trained using different proportions of the in-language training datasets. The top-performing dash-dotted curve shows the performance of the model trained using the language-specific dialogues and all of the English training data.

The multilingual DST models offer substantial performance improvements over the monolingual ones, with particularly large gains in the low-data scenarios investigated in Figure 7.3 (dash-dotted purple line). This figure also shows that the difference in performance between mono- and cross-lingually specialised vectors is not very substantial. Again, the large disparity in SimLex scores induced only minor improvements in DST performance.

Conclusions In summary, these results show that: **a)** semantically specialised vectors benefit DST performance; **b)** large gains in SimLex scores do not always induce large downstream gains; and **c)** high-quality cross-lingual spaces facilitate transfer learning between languages and offer an effective method for bootstrapping DST models for lower-resource languages. Finally, German DST performance is substantially weaker than both English and Italian, corroborating the intuition that linguistic phenomena such as cases and compounding make German DST very challenging. The second part of this chapter delves deeper into this problem, showing that language-specific morphology can be modelled through the underlying word vector space, leading to substantial gains in downstream performance for morphologically rich languages such as German.

7.3 The Importance of Morphology

Morphologically rich languages, in which *"substantial grammatical information... is expressed at word level"* (Tsarfaty et al., 2010), pose specific challenges for NLP methods. This is not always considered when techniques are evaluated on languages such as English or Chinese, which do not have rich morphology. In the case of distributional vector space models, morphological complexity brings two challenges to the fore:

1. **Estimating Rare Words:** A single lemma can have many different surface realisations. Naively treating each realisation as a separate word leads to sparsity problems and a failure to exploit their shared semantics. On the other hand, lemmatising the entire corpus can obfuscate the differences that exist between different word forms even though they share some aspects of meaning.

2. **Embedded Semantics:** Morphology can encode semantic relations such as antonymy (e.g., *literate* and *illiterate*, *expensive* and *inexpensive*) or near-synonymy (e.g., *north*, *northern*, *northerly*, *northernmost*, etc.).

7.3.1 Morph-Fitting

The two challenges can be tackled jointly by introducing a resource-light vector space fine-tuning procedure termed *morph-fitting*. Unlike the work on semantic specialisation presented in the previous chapter, the proposed method does not require curated knowledge bases or gold lexicons. Instead, it makes use of the observation that morphology implicitly encodes semantic signals pertaining to synonymy (e.g., German word inflections *katalanisch, katalanischem, katalanischer* denote the same semantic concept in different grammatical roles), and antonymy (e.g., *mature* vs. *immature*), capitalising on the proliferation of word forms in morphologically rich languages. Morph-fitting is steered by a set of linguistic constraints derived from simple language-specific rules which describe (a subset of) morphological processes in a language. The constraints emphasise similarity on one side (by extracting *inflectional* morphological synonyms), and antonymy on the other (by extracting *derivational* morphological antonyms). Morph-fitting is illustrated in Figure 7.4.

The key idea of the fine-tuning process is to pull synonymous examples described by the constraints closer together in the transformed vector space, while at the same time pushing antonymous examples away from each other. The injection of morphological constraints enables: **a)** the estimation of more accurate vectors for low-frequency words which are linked to their high-frequency forms by the constructed constraints, tackling the data sparsity

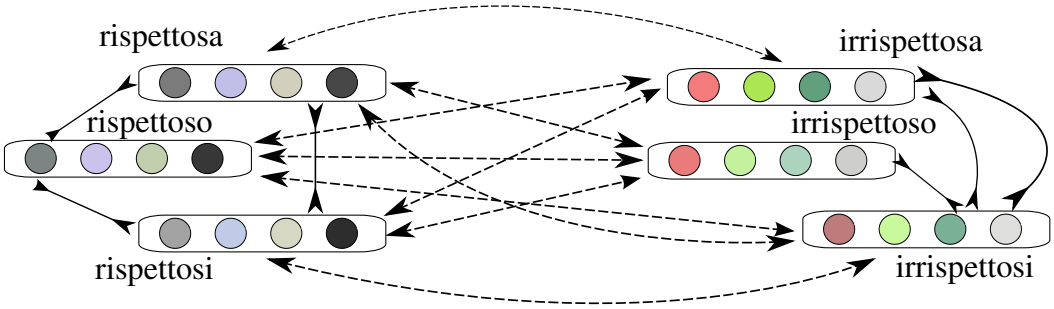


Figure 7.4 Morph-fitting in Italian. Representations for *rispettoso*, *rispettosa*, *rispettosi* (EN: *respectful*), are pulled closer together in the vector space (solid lines; ATTRACT constraints). At the same time, the model pushes them away from their antonyms (dashed lines; REPEL constraints) *irrispettoso*, *irrispettosa*, *irrispettosi* (EN: *disrespectful*), obtained through morphological affix transformation captured by language-specific rules.

problem;[4] and **b)** specialising the distributional space to distinguish between similarity and relatedness, supporting language understanding tasks such as dialogue state tracking.

As in the previous chapter, the focus is on four languages with varying levels of morphological complexity: English (EN), Italian (IT), German (DE) and Russian (RU). The starting point is a comprehensive list of word tokens for each language: vocabularies W_{en}, W_{de}, W_{it}, W_{ru} are compiled by retaining all word forms from the four Wikipedias with word frequency over 10, as shown in Table 7.2. The next step is extracting sets of linguistic constraints from these vocabularies using a set of simple language-specific *if-then-else* rules. Examples of generated constraints are shown in Table 7.3.[5] These constraints are then used as input for the ATTRACT-REPEL algorithm, presented in the previous chapter.

7.3.2 Language-Specific Rules and Constraints

The ATTRACT-REPEL procedure is entirely driven by the input ATTRACT and REPEL sets of constraints. As shown in Chapter 6, these can be extracted from a variety of semantic databases such as WordNet (Miller, 1995) or BabelNet (Ehrmann et al., 2014; Navigli and Ponzetto, 2012). Morph-fitting takes an alternative approach, extracting constraints *without* curated knowledge bases in a spectrum of languages by exploiting inherent language-

[4]For instance, the vector for the word *katalanischem*, which occurs only 9 times in the German Wikipedia will be pulled closer to the more reliable vectors for *katalanisch* and *katalanischer*, with frequencies of 2097 and 1383, respectively.

[5]A native speaker can easily come up with these sets of morphological rules (or at least with a reasonable subset of them) without any linguistic training. What is more, the rules for DE, IT, and RU were created by non-native, non-fluent speakers with a limited knowledge of the three languages, exemplifying the simplicity and portability of the approach.

Language	Vocabulary Size	Synonym Count	Antonym Count
English	1,368,891	231,448	45,964
German	1,216,161	648,344	54,644
Italian	541,779	278,974	21,400
Russian	950,783	408,400	32,174

Table 7.2 Vocabulary sizes and counts of ATTRACT (synonym) and REPEL (antonym) constraints generated by the language-specific rules for each language.

specific properties related to the morphology of each language. This relaxation ensures wider portability to languages and domains without readily available or adequate resources.

The core difference between *inflectional* and *derivational morphology* can be summarised as follows: the former refers to a set of processes through which the word form expresses meaningful syntactic information, e.g., verb tense, without any change to the semantics of the word. On the other hand, derivational morphology refers to the formation of new words with semantic shifts in meaning (Cotterell and Schütze, 2017; Haspelmath and Sims, 2013; Lazaridou et al., 2013; Schone and Jurafsky, 2001; Zeller et al., 2013).

Extracting ATTRACT Pairs To synthesise ATTRACT constraints, morph-fitting uses *inflectional* morphology rules which preserve the full meaning of a word, modifying it only to reflect grammatical roles such as verb tense or case markers (e.g., *(en_read, en_reads)* or *(de_katalanisch, de_katalanischer)*). English is widely recognised as a morphologically simple language (Avramidis and Koehn, 2008; Cotterell et al., 2016). Consequently, morph-fitting uses only two inflectional rules for this language:

1. **(R1)** *If* $w_1, w_2 \in W_{en}$, *where* $w_2 = w_1$ + *ing/ed/s*, *then* add (w_1, w_2) and (w_2, w_1) to the set of ATTRACT constraints A. This rule yields pairs such as *(look, looks), (look, looking), (look, looked)*, and many others.

2. **(R2)** Let $w[:-1]$ denote a function which strips the last character from word w. Then, *if* w_1 ends with the letter *e and* $w_1 \in W_{en}$ *and* $w_2 \in W_{en}$, *where* $w_2 = w_1[:-1]$ + *ing/ed*, *then* add (w_1, w_2) and (w_2, w_1) to A. This rule creates pairs such as *(create, creating)*, *(create, created)*, and many others.

The other three languages, with more complicated morphology, require a larger number of rules. In Italian, the set of rules spans: **1)** the regular formation of plural (*libro / libri*); **2)** regular verb conjugation (*aspettare / aspettiamo*); **3)** regular formation of past participle (*aspettare / aspettato*); and **4)** rules regarding grammatical gender (*bianco / bianca*). In addition to these, another set of rules is defined for German and Russian: **5)** regular declension: *(de_asiatisch / de_asiatischem), (ru_*трава */ ru_*траве*), (ru_*трава */ ru_*травой*)*, etc.

Language	Constraint	Relation	Rule
EN	(sunflower, sunflowers)	SYN	Singular-Plural
	(suffer, suffered)	SYN	Past Participle
	(ambiguous, unambiguous)	ANT	Derivational Antonymy
	(regular, irregular)	ANT	Derivational Antonymy
IT	(zucchero, zuccheri)	SYN	Singular-Plural
	(vincere, vincono)	SYN	Conjugation
	(rapido, rapida)	SYN	Gender
	(visibilità, invisibilità)	ANT	Derivational Antonymy
DE	(Kategorie, Kategorien)	SYN	Singular-Plural
	(kaufst, kauft)	SYN	Conjugation
	(katalanisch, katalanischem)	SYN	Declension
	(dokumentiert, undokumentiert)	ANT	Derivational Antonymy

Table 7.3 Example synonymous (inflectional) and antonymous (derivational) constraints generated by the morph-fitting approach across three languages.

Extracting REPEL Pairs As another source of implicit semantic information, morph-fitting synthesises a set of plausible *derivational antonyms*. These consist of pairs of words that denote concepts with opposite meaning, generated through a derivational process. To generate potential antonyms, a standard set of English antonymy prefixes is used: $AP_{en} =$ *{dis, il, un, in, im, ir, mis, non, anti}* (Fromkin et al., 2013). The antonym generation rule is simple: *If* $w_1, w_2 \in W_{en}$, *such that* w_2 is generated by adding a prefix from AP_{en} to w_1, *then* add (w_1, w_2) and (w_2, w_1) to the set of REPEL constraints R. This rule generates pairs such as *(advantage, disadvantage)* and *(regular, irregular)*. An additional rule replaces the suffix *-ful* with *-less*, extracting antonyms such as *(careful, careless)*.

Following the same principle for the other languages, the following sets of antonymy prefixes are used: $AP_{de} =$ *{un, nicht, anti, ir, in, miss}*, $AP_{it} =$ *{in, ir, im, anti}*, and $AP_{ru} =$ *{*не, анти*}*. For instance, the procedure generates Italian pairs *(rispettoso, irrispettoso), (rispettosa, irrispettosa)* and *(rispettosi, irrispettosi)* (see Figure 7.4). For German, another rule targeting suffix replacement is used: *-voll* is replaced by *-los* (*geschmackvoll / geschmacklos*).

The set of REPEL constraints is expanded further by transitively combining antonymy pairs from the previous step with inflectional ATTRACT pairs. This step yields additional constraints such as *(rispettosa / irrispettosi, rispettoso / irrispetosa, etc.)* (see Figure 7.4). The final Attract and Repel constraint counts are shown in Table 7.2. Table 7.3 gives an overview of the kinds of constraints generated by the morph-fitting procedure.

Use of Existing Morphological Resources The use of morphological resources to improve the representations of morphemes and words is an active area of research. The majority of proposed architectures encode morphological information, provided either as gold standard morphological resources (Sylak-Glassman et al., 2015), who use CELEX (Baayen et al., 1995), or as an external analyser such as Morfessor (Creutz and Lagus, 2007). This information is combined with distributional information at training time in the language modelling objective function (Bhatia et al., 2016; Botha and Blunsom, 2014; Cotterell and Schütze, 2015; Luong et al., 2013; Qiu et al., 2014, i.a.). The key idea is to learn a morphological composition function (Cotterell and Schütze, 2017; Lazaridou et al., 2013) which synthesises the representation of a word given the representations of its constituent morphemes. Contrary to morph-fitting, these approaches typically rely on external morphological resources.

7.3.3 Experimental Setup

Starting Word Vectors For each of the four languages, the skip-gram with negative sampling (SGNS) model (Mikolov et al., 2013b) is trained on the latest Wikipedia dump of each language. 300-dimensional word vectors are induced, with the frequency cut-off set to 10. The vocabulary sizes for each language are shown in Table 7.2. Other SGNS parameters were set to their standard values: 15 epochs, 15 negative samples, global learning rate: .025, subsampling rate: $1e-4$. These vectors are henceforth referred to as SGNS-LARGE.

To assess its dependence on the initial distributional vectors, morph-fitting is also applied to six well-known distributional spaces across three languages (EN, IT and DE), available from prior work (Dinu et al., 2015; Luong et al., 2015b; Mikolov et al., 2013a; Pennington et al., 2014; Vulić and Korhonen, 2016a).

Morph-fitting Variants Two different morph-fitting variants are investigated: **1)** MFIT-A, which uses only the ATTRACT constraints generated using rules based on inflectional morphology; and **2)** MFIT-AR, which includes the derivational REPEL constraints as well.

Morph-fixed Vectors *Morph-fixing* is a simple baseline method which uses the same knowledge as morph-fitting. This approach fixes the vector of each word to the distributional vector of its most frequent inflectional synonym, tying the vectors of low-frequency words to their more frequent inflections. For each word w_1, the set of $M+1$ words W_{w_1} is constructed, consisting of the word w_1 itself and all M words which co-occur with w_1 in the generated ATTRACT constraints. The word w'_{max} with the maximum frequency in the training data is then chosen as the *pivot*, and all other word vectors in W_{w_1} are replaced with this vector. The

Initial Vector Space	Distrib. Vectors	+ MFIT-A	+ MFIT-AR
EN: GloVe-6B (300) (Pennington et al., 2014)	0.324	0.376	**0.438**
EN: SG-BOW2-PW (300) (Mikolov et al., 2013a)	0.339	0.385	**0.439**
DE: SG-DEPS-PW (300) (Vulić and Korhonen, 2016a)	0.267	0.318	**0.325**
DE: BiSkip-DE (256) (Luong et al., 2015b)	0.354	0.414	**0.421**
IT: SG-DEPS-PW (300) (Vulić and Korhonen, 2016a)	0.237	0.351	**0.391**
IT: CBOW5-Wacky (300) (Dinu et al., 2015)	0.363	0.417	**0.446**

Table 7.4 Results on multilingual SimLex-999 (EN, DE, IT) with two morph-fitting variants.

morph-fixed vectors (MFIX) serve as the primary baseline, since they outperformed another baseline method based on *stemming* across all intrinsic and extrinsic experiments.

7.3.4 Intrinsic Evaluation

Experiments Across Languages Both morph-fitting variants are tested across all languages. The results are summarised in Table 7.4, where six well-known distributional vector spaces for EN, DE and IT are morph-fitted using morphological constraints for their vocabularies. The scores in this table confirm the effectiveness and robustness of morph-fitting across languages, suggesting that the idea of fitting to morphological constraints is indeed language-agnostic, given the set of language-specific rule-based constraints.

Figures 7.6a-7.6d show the same trends in performance for the morph-fitted SGNS-LARGE vectors. The figure also demonstrates that morph-fitted vector spaces consistently outperform the morph-fixed ones. The comparisons between MFIT-A and MFIT-AR show that both sets of constraints are important for the fine-tuning process. MFIT-A yields consistent gains over the initial spaces, and (consistent) further improvements are achieved by also incorporating the antonymous REPEL constraints.

Comparison to Other Specialisation Methods Other specialisation models can be used in place of ATTRACT-REPEL, driven by the same set of morphological constraints. Figure 7.5 compares ATTRACT-REPEL to retrofitting (Faruqui et al., 2015) and counter-fitting (presented in Chapter 4). The results show that ATTRACT-REPEL substantially outperforms these methods across all languages. In line with the results presented in Chapter 6, the context-sensitive vector space updates again make far better use of the available semantic constraints than the context-insensitive global updates performed by retro- and counter-fitting.

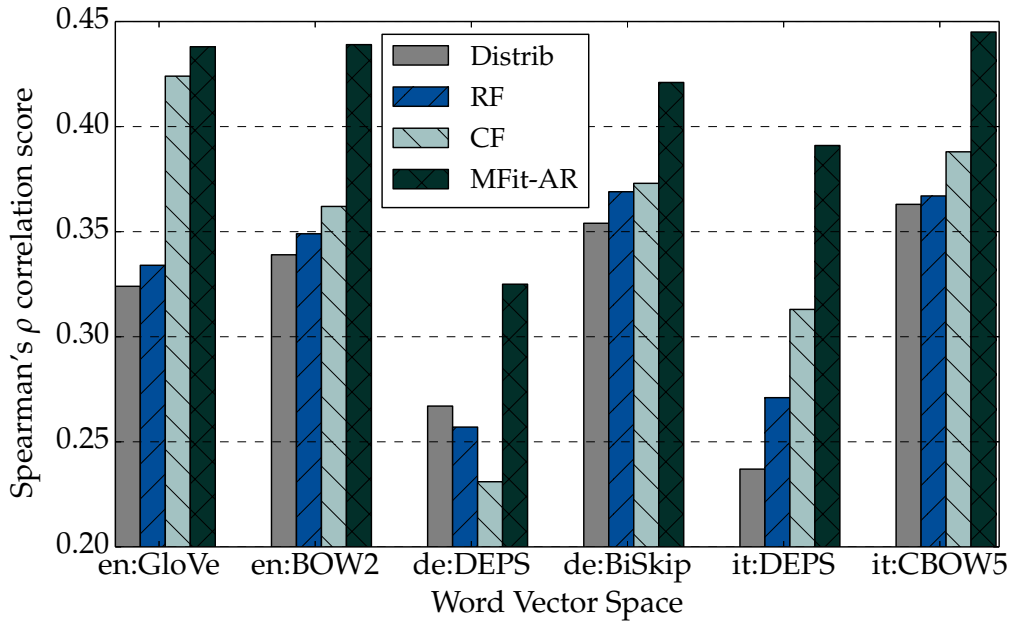


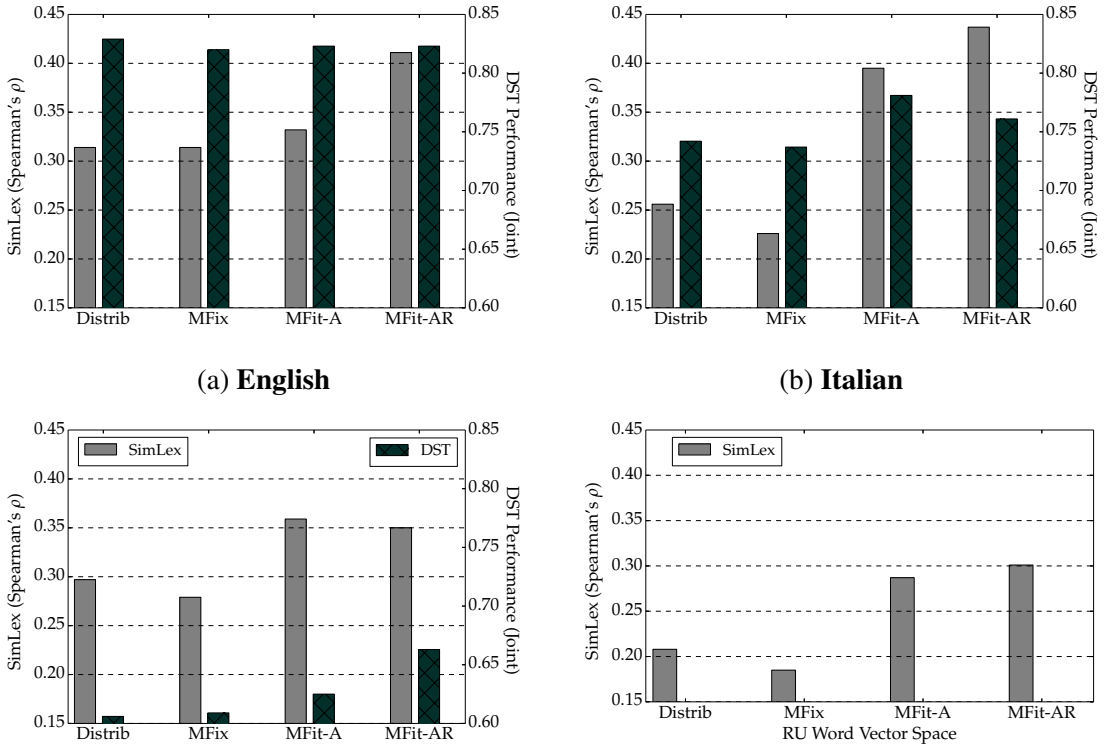
Figure 7.5 A comparison of ATTRACT-REPEL (the MFIT-AR variant) with retrofitting (RF) and counter-fitting (CF). The same six distributional word vector spaces from Table 7.4 are used as the starting points for all three procedures.

7.3.5 Downstream Evaluation

The Neural Belief Tracking (NBT) models for English, German and Italian are trained using the four variants of the SGNS-LARGE vectors: the initial distributional vectors, the *morph-fixed* vectors, and the two variants of the *morph-fitted* vectors (MFIT-A and MFIT-AR).

Previous experiments showed that semantic specialisation of the employed word vectors benefits DST performance across all three languages. However, large gains on SimLex-999 did not always induce correspondingly large gains in downstream performance. The following experiments investigate the extent to which morph-fitting improves DST performance, and whether these gains exhibit stronger correlation with intrinsic performance.

Results and Discussion The dark bars (against the right axes) in Figure 7.6 show the DST performance of NBT models making use of the four word vector collections. Italian and German benefit from both kinds of morph-fitting: Italian performance increases from 74.1 $\rightarrow$ 78.1 (MFIT-A) and German performance rises even more: 60.6 $\rightarrow$ 66.3 (MFIT-AR), setting new state-of-the-art scores for both datasets. The morph-fixed vectors do not enhance DST performance, probably because fixing word vectors to their highest-frequency inflectional form eliminates useful semantic content encoded in the original vectors. On the



(a) **English** (b) **Italian**

(c) **German** (d) **Russian**

Figure 7.6 An overview of the results (Spearman's rank correlation) for four languages on SimLex-999 (grey bars, left *y* axis) and the downstream DST performance (dark bars, right *y* axis). The four-by-two bars for each language correspond to the SGNS-LARGE vectors, their morph-fixed variant, and the two morph-fitted versions of these vector collections. Note that there are no DST datasets available for Russian.

other hand, morph-fitting makes use of this information, supplementing it with semantic relations between different morphological forms. These conclusions are in line with the SimLex gains, where morph-fitting outperforms both distributional and morph-fixed vectors.

Intrinsic versus Downstream Performance English performance shows little variation across the four word vector collections investigated in the experiments. This corroborates the intuition that, as a morphologically simpler language, English stands to gain less from fine-tuning the morphological variation for downstream applications. This result again points at the discrepancy between intrinsic and extrinsic evaluation: the considerable gains in SimLex performance do not necessarily induce similar gains in downstream performance. Additional discrepancies between SimLex and downstream DST performance are detected for German and Italian. Despite the slight drop in SimLex performance with German MFIT-AR vectors compared to the MFIT-A ones, their relative performance is reversed in the DST task. The opposite trend is observed in Italian, where the MFIT-A vectors score lower than the MFIT-AR vectors on SimLex, but higher on the DST task.

In summary, these results show that SimLex is not a perfect proxy for downstream performance in language understanding tasks. Regardless, its performance does correlate with downstream performance to a large extent, providing an efficient indicator for the usefulness of specific word vector collections for extrinsic tasks such as dialogue state tracking.

7.4 Conclusion

This chapter used two novel non-English Dialogue State Tracking (DST) datasets (in German and Italian) to show that semantically rich cross-lingual vectors facilitate language transfer in DST, providing an effective method for bootstrapping belief tracking models for new languages. The cross-lingual vectors can be used to train a single model that performs DST in all three languages, in each case outperforming the monolingual model. To the best of my knowledge, this is the first work on multilingual training of any component of a statistical dialogue system. The results indicate that multilingual training holds great promise for bootstrapping language understanding models for other languages, especially for dialogue domains where data collection is very resource-intensive.

In the second part of the chapter, simple morphological rules for the four languages were used to improve the semantics of existing word vector collections by relying on semantic content embedded in sub-word level information. As shown in the DST evaluation, using this information to further expand the lexical coverage of word vector collections has a very

strong effect on downstream performance. In fact, improving the semantic content through morphology had a stronger impact than embedding information from large-scale semantic lexicons such as BabelNet or PPDB.

Future Work The results in both parts of this chapter highlight the lack of strong correlation between the popular intrinsic evaluation datasets such as SimLex-999 and downstream language understanding performance in tasks such as dialogue state tracking. An interesting direction for future work would be to uncover which properties/features of word embeddings used as building blocks for data-driven language understanding models are the most useful indicators of strong downstream performance.

Chapter 8

Conclusion

This thesis proposed a data-driven paradigm for performing language understanding in task-oriented spoken dialogue systems. Even though the investigated dialogue domains are limited in scope, existing *word-based* models rely on exact matching supplemented with hand-crafted semantic lexicons to perform robust language understanding. The proposed Neural Belief Tracking (NBT) models forsake this *delexicalisation* paradigm in favour of reasoning over vectorial representations of words, also known as *word embeddings*. The NBT model is fully data-driven: it makes use of the semantic content embedded in word vectors without making use of any hand-crafted rules or lexicons to infer user goals.

The data-driven approach used by the Neural Belief Tracker is entirely dependent on the semantic quality of the underlying word vector space. Consequently, a major part of this thesis investigated the problem of *semantic specialisation*, where external lexical information is used to improve the semantic content of word vector collections. Semantic specialisation is first used to improve the semantic content of monolingual word vectors, leading to improved performance both when these vectors are used to induce semantic lexicons for delexicalisation-based models, and when they are used to power the data-driven Neural Belief Tracker. Subsequently, semantic specialisation is used to induce cross-lingual vector spaces, facilitating the training of multilingual data-driven language understanding models. The multilingual models improve over monolingual ones, providing an effective method for bootstrapping language understanding models for lower-resource languages.

8.1 Summary of Contributions

The presented research on data-driven language understanding models powered by semantically specialised word vectors offers several important takeaways, summarised next.

The Power of Delexicalisation The word-based delexicalisation framework proposed by Henderson et al. (2014c; 2014d) is considered to be the state-of-the-art approach for performing language understanding in task-oriented spoken dialogue systems. This framework rests on identifying the occurrences of onotology-defined *slot values* to extract useful features for predicting user intent (the *belief state*). The research presented in this thesis first demonstrated the power of delexicalisation by extending this approach to construct multi-domain belief tracking models. These models can use data from disparate dialogue domains for training, producing a single model which achieves superior performance across all constituent domains. Moreover, the proposed multi-domain training procedure yields an effective way to train models for low-resource domains by using knowledge from resource-rich domains to boost language understanding performance in low-resource ones.

The Limitations of Delexicalisation The presented research next shifts focus to the deficiencies of delexicalisation-based models, scrutinising their dependence on *semantic lexicons*, which are lists of rephrasings specified by the system designer to deal with linguistic variation. For instance, the system designer for a dialogue system built to help users purchase laptops might hand-code direct mappings from (sequences of) words in user input to specific ontology-defined intents. An example list of rephrasings for ontology value PRICE=MODERATE contains potential rephrasings such as: *average*, *reasonable*, *mid priced*, *mid range*, and 15 others. The construction of such lexicons is an arduous process requiring many iterations between users and system designers. Even this short list of rephrasings hints at the complications which permeate the construction of semantic lexicons: specifying a mapping from *mid* to *moderate* introduces a conflict with another slot value, DRIVERANGE=MIDDLE. The system designer can only decide which rephrasing to use after observing user behaviour with both and deciding which rephrasing led to higher success rates and user satisfaction.

Inducing Domain-Specific Semantic Lexicons To reduce the human cost of constructing domain-specific lexicons, the counter-fitting procedure uses pre-trained word vector spaces to induce lists of potential rephrasings for ontology values. This work highlights the ineptitude of standard distributional word vector spaces (such as WORD2VEC or GloVe) for this task. These models are based on the *distributional hypothesis*, inducing similar representations for words that occur in similar contexts. While this usually leads to words with similar semantics being clustered together, it also means that words like *expensive* and *inexpensive* become close neighbours, which is very detrimental for the downstream language understanding task. The counter-fitting procedure introduces *antonymous* relations between ontology values, supplementing the fine-tuning process with similarity and antonymy relations from large-

scale lexical resources such as the Paraphrase Database (Ganitkevitch et al., 2013). The resulting vector spaces can be used to induce semantic lexicons which boost the performance of delexicalisation-based models across two different dialogue domains. However, the introduction of the expensive counter-fitting procedure breaks the end-to-end learning pipeline of Henderson's word-based model, motivating the development of an alternative framework which can make use of pre-trained word vectors directly.

Data-Driven Language Understanding using Word Vectors The focus of the thesis next shifts to showing that word vectors can be used as sole building blocks for language understanding models. The presented Neural Belief Tracking (NBT) models introduce a novel data-driven framework for language understanding. The data flows from user and system input to the predicted belief state, starting with *fixed* word vector representations for words in the system and user utterance, and using a complex hierarchical neural network model to predict the new belief state. The model makes use of the semantics embedded in pre-trained word vector collections to resolve lexical ambiguity, with the internal neural network parameters capturing domain-specific rephrasings which users employ in the given dialogue domain. The proposed framework achieves performance superior to the delexicalisation-based models and removes the need for hand-crafted system components.

Semantic Specialisation improves Downstream Performance The NBT models' performance showed that improving the semantic content of the underlying word vector collections led to improved performance in downstream language understanding tasks. This motivated the development of an improved *semantic specialisation* procedure, which could inject information about synonymous or antonymous word pairs to boost the semantic content of pre-trained word vector collections. The ATTRACT-REPEL procedure takes an approach different from counter-fitting, using a fine-grained tuning procedure which pulls synonyms close and pushes antonyms away until they are closer (or further) from each other than other randomly sampled word vectors. The ATTRACT-REPEL procedure can use existing lexical resources to produce word vector spaces which achieve state-of-the-art performance on word similarity tasks across different languages.

Cross-Lingual Vector Spaces Semantic specialisation can also use existing lexical corpora to induce high-quality *cross-lingual* vector spaces. If translation pairs extracted from cross-lingual semantic lexicons such as BabelNet (Navigli and Ponzetto, 2012) are used as synonyms, the ATTRACT-REPEL procedure can tie the distributional vector spaces of multiple languages into a single cross-lingual vector space. The unified vector space achieves

superior performance on word similarity tasks across all of its constituent languages, showing evidence of positive semantic transfer from high to lower-resource languages. Interestingly, for the investigated word similarity tasks, cross-lingual specialisation produces bilingual vector spaces which substantially outperform a representative suite of existing methods for inducing cross-lingual vector spaces. To show that gains in semantic content persist for lower-resource languages, new intrinsic evaluation datasets are collected for Hebrew and Croatian, two languages with very limited monolingual resources. BabelNet constraints are then used to tie these two languages into bilingual vector spaces with English, showing performance gains similar to higher-resource languages such as Italian or German.

Belief Tracking across Languages The thesis next explores the problem of deploying statistical dialogue systems to languages other than English. Two new datasets (for Italian and German) are created and used to show that these languages present challenges that are not investigated in existing English-centric spoken dialogue systems research. Neural Belief Tracking models are trained for these two languages, powered by different word vector collections. The models' performance reiterates the importance of semantic specialisation, with both Italian and German belief tracking performance benefiting from semantic specialisation more than the English DST performance did. Finally, the induced cross-lingual vector spaces are used to train multilingual belief tracking models which can perform language understanding across all three languages at once. This is enabled by the data-driven nature of NBT models, which use word embeddings of ontology labels in place of traditional language-specific class labels. Given high-quality cross-lingual vector spaces, the trained multilingual models outperform the monolingual ones, yielding an effective method for bootstrapping belief tracking models for lower-resource languages.

The Importance of Morphology Given the complex morphology of Italian and German (compared to English), their downstream performance shows a greater dependence on high-quality word vector estimates for low-frequency words. To validate this claim, the proposed *morph-fitting* procedure uses simple language-specific rules instead of lexicons such as BabelNet to supervise the ATTRACT-REPEL procedure. Morph-fitting brings the morphological variants of the same word (e.g., *mature*, *matured*, *matures* and *maturing*) closer together, while pushing derivational antonyms (e.g., *mature* and *immature*) further away. The morph-fitted vectors induce substantial gains in intrinsic word similarity and the downstream belief tracking performance. However, the scale of these improvements is very different from previous semantic specialisation experiments. Whereas morph-fitting induces relatively small gains in intrinsic SimLex-999 performance, it results in substantially larger

gains in downstream belief tracking performance. This reveals a disparity between intrinsic and extrinsic evaluation: large intrinsic gains do not always induce substantial improvements in dialogue state tracking, and vice versa.

8.2 Future Directions

The thesis concludes by presenting further research questions posed by the presented work.

Application to Real-World Domains The presented Neural Belief Tracking models do not rely on semantic lexicons, removing one important obstacle to scaling existing task-oriented dialogue systems to larger, more sophisticated dialogue domains. However, the real-world validation of the proposed paradigm can only come through successful application of this model to industry-scale domain ontologies and in evaluation with real users. As shown by the contrasting performance between DSTC 2 and WOZ 2.0 experiments, the experimental setup can greatly affect the way users speak to dialogue systems, with the quality of automated speech recognition and the systems' language understanding capability determining the way users choose to speak to spoken dialogue systems.

Robustness to Different Failure Modes Data-driven black-box neural networks (of which the NBT is an instance) can be hard to interpret. Their end-to-end learning nature often leads to surprising failure modes that can be difficult to handle in real-time applications. Despite their flaws and limitations, semantic lexicons provide a relatively elegant way for the system designer to circumvent frequently encountered failure modes. It remains to be seen how robust the NBT is when deployed in real-world systems, and how much harder it becomes to deal with unexpected failure modes now that all hand-crafted components have been surgically removed from the language understanding pipeline.

Data-Driven Understanding across Multiple Domains The word-based delexicalisation model can train using data from disparate dialogue domains, learning to transfer useful features from high-resource to resource-poor dialogue domains. This thesis has shown that the Neural Belief Tracking model outperforms delexicalisation-based models deprived of their hand-crafted semantic lexicons. Both the rule-based and the statistical update variants of the NBT model support the same kind of multi-domain training. If multi-domain NBT models outperform domain-specific ones, this would provide further evidence that the proposed data-driven framework could supersede existing delexicalisation-based models.

Uncovering Language-Specific Particularities The presented work on Italian and German revealed the dependency of language understanding performance on modelling morphology, not made apparent even by intrinsic evaluation of vector spaces in those languages. Morphologically rich languages such as Russian or Polish are likely to mirror the trends observed for German, and their language understanding performance is likely to improve when the underlying vector space is transformed to model language-specific morphology. Non Indo-European languages such as Arabic, Chinese or Swahili would present more substantial challenges to the proposed language understanding paradigm. For instance, Chinese and Japanese segment sentences, but not words, whereas written tokens in Vietnamese represent syllables. Extending the data-driven paradigm to these languages presents an interesting challenge, since the model would have to learn to segment sentences into words if it were to benefit from semantically specialised word vectors. Alternatively, the models could operate over pre-segmented sentences produced by third-party models, which would break the data-driven pipeline. Such third-party segmentation tools are typically language-specific, hindering the elegance of the proposed approach which could scale across Indo-European languages using word-level supervision from lexical resources such as BabelNet.

Disparity between Intrinsic and Downstream Performance Intrinsic word similarity datasets such as SimLex-999 (Hill et al., 2014) or SimVerb-3500 (Gerz et al., 2016) were developed to assign numeric estimates for the semantic quality of word vector collections. As such, it is expected that performance on intrinsic datasets could be used to predict downstream performance in tasks which make use of the semantics embedded in the employed word vectors. The experiments presented in this thesis showed that SimLex-999 performance does correlates with downstream language understanding performance. However, large gains in intrinsic performance do not always induce correspondingly large improvements in downstream belief tracking. An interesting direction for further work would be to investigate which aspects of pre-trained word vector collections are most important for extrinsic performance. One way to do this would be to incorporate the proposed semantically specialised word vector spaces into other language understanding architectures (e.g., those for *question answering* or *textual entailment*) and measure their impact on downstream performance. Another approach would be to develop alternative data-driven models for belief tracking to ensure that the discrepancies between intrinsic and extrinsic performance persist, and are not specific to the experimental setup used by the Neural Belief Tracker.

Bibliography

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Al-Rfou, R., Alain, G., Almahairi, A., Angermueller, C., Bahdanau, D., Ballas, N., Bastien, F., Bayer, J., Belikov, A., Belopolsky, A., Bengio, Y., Bergeron, A., Bergstra, J., Bisson, V., Bleecher Snyder, J., Bouchard, N., Boulanger-Lewandowski, N., Bouthillier, X., de Brébisson, A., Breuleux, O., Carrier, P.-L., Cho, K., Chorowski, J., Christiano, P., Cooijmans, T., Côté, M.-A., Côté, M., Courville, A., Dauphin, Y. N., Delalleau, O., Demouth, J., Desjardins, G., Dieleman, S., Dinh, L., Ducoffe, M., Dumoulin, V., Ebrahimi Kahou, S., Erhan, D., Fan, Z., Firat, O., Germain, M., Glorot, X., Goodfellow, I., Graham, M., Gulcehre, C., Hamel, P., Harlouchet, I., Heng, J.-P., Hidasi, B., Honari, S., Jain, A., Jean, S., Jia, K., Korobov, M., Kulkarni, V., Lamb, A., Lamblin, P., Larsen, E., Laurent, C., Lee, S., Lefrancois, S., Lemieux, S., Léonard, N., Lin, Z., Livezey, J. A., Lorenz, C., Lowin, J., Ma, Q., Manzagol, P.-A., Mastropietro, O., McGibbon, R. T., Memisevic, R., van Merriënboer, B., Michalski, V., Mirza, M., Orlandi, A., Pal, C., Pascanu, R., Pezeshki, M., Raffel, C., Renshaw, D., Rocklin, M., Romero, A., Roth, M., Sadowski, P., Salvatier, J., Savard, F., Schlüter, J., Schulman, J., Schwartz, G., Serban, I. V., Serdyuk, D., Shabanian, S., Simon, E., Spieckermann, S., Subramanyam, S. R., Sygnowski, J., Tanguay, J., van Tulder, G., Turian, J., Urban, S., Vincent, P., Visin, F., de Vries, H., Warde-Farley, D., Webb, D. J., Willson, M., Xu, K., Xue, L., Yao, L., Zhang, S., and Zhang, Y. (2016). Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688.

Alam, J., Gupta, V., Kenny, P., and Dumouchel, P. (2015). Speech recognition in reverberant and noisy environments employing multiple feature extractors and i-vector speaker adaptation. *EURASIP Journal on Advances in Signal Processing*, 2015(1):50.

Aletras, N. and Stevenson, M. (2015). A hybrid distributional and knowledge-based model of lexical semantics. In *Proceedings of the Fourth Joint Conference on Lexical and Computational Semantics, *SEM*.

Ammar, W., Mulcaire, G., Ballesteros, M., Dyer, C., and Smith, N. (2016). Many languages, one parser. *Transactions of the Association for Computational Linguistics*, 4:431–444.

Avramidis, E. and Koehn, P. (2008). Enriching morphologically poor languages for statistical machine translation. In *Proceedings of ACL*.

Baayen, H. R., Piepenbrock, R., and van Rijn, H. (1995). The CELEX lexical data base on CD-ROM.

Baker, C. F., Fillmore, C. J., and Lowe, J. B. (1998). The Berkeley FrameNet project. In *Proceedings of ACL*.

Bansal, M., Gimpel, K., and Livescu, K. (2014). Tailoring continuous word representations for dependency parsing. In *Proceedings of ACL*.

Baroni, M., Dinu, G., and Kruszewski, G. (2014). Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of ACL*.

Bengio, Y., Courville, A. C., and Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

Bhatia, P., Guthrie, R., and Eisenstein, J. (2016). Morphological priors for probabilistic neural word embeddings. In *Proceedings of EMNLP*.

Bian, J., Gao, B., and Liu, T. (2014). Knowledge-powered deep learning for word embedding. In *Proceedings of ECML-PKDD*.

Biermann, A. and Long, P. M. (1996). The composition of messages in speech-graphics interactive systems. In *Proceedings of the International Symposium on Spoken Dialogue*.

Black, A. and Lenzo, K. (2001). FLite: A small fast run-time synthesis. In *ISCA Tutorial and Research Workshop on Speech Synthesis*.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Blitzer, J., Dredze, M., and Pereira, F. (2007). Biographies, Bollywood, Boom-boxes and Blenders: Domain Adaptation for Sentiment Classification. In *Proceedings of ACL*.

Bohus, D. and Rudnicky, A. (2005). Sorry, I Didn't Catch That! – An Investigation of Non-understanding Errors and Recovery Strategies. In *Proceedings of SIGDIAL*.

Bohus, D. and Rudnicky, A. (2006). A "k hypotheses + other" belief updating model. In *Proceedings of the AAAI Workshop on Statistical and Empirical Methods in Spoken Dialogue Systems*.

Botha, J. A. and Blunsom, P. (2014). Compositional morphology for word representations and language modelling. In *Proceedings of ICML*.

Bourlard, H. A. and Morgan, N. (1993). *Connectionist Speech Recognition: A Hybrid Approach*. Kluwer Academic Publishers, Norwell, MA, USA.

Bruni, E., Tran, N., and Baroni, M. (2014). Multimodal distributional semantics. *Journal of Artificial Intelligence Research*, 49:1–47.

Bui, T., Poel, M., Nijholt, A., and Zwiers, J. (2009). A Tractable Hybrid DDN–POMDP Approach to Affective Dialogue Modelling for Probabilistic Frame-based Dialogue Systems. *Natural Language Engineering*, 15(2):273–307.

Casanueva, I., Hain, T., Christensen, H., Marxer, R., and Green, P. (2015). Knowledge transfer between speakers for personalised dialogue management. In *Proceedings of SIGDIAL*.

Casanueva, I., Hain, T., and Green, P. (2016a). Improving generalisation to new speakers in spoken dialogue state tracking. In *Proceedings of Interspeech*.

Casanueva, I., Hain, T., Nicolao, M., and Green, P. (2016b). Using phone features to improve dialogue state tracking generalisation to unseen states. In *Proceedings of SIGDIAL*.

Celikyilmaz, A. and Hakkani-Tur, D. (2015). Convolutional Neural Network Based Semantic Tagging with Entity Embeddings. In *Proceedings of the NIPS Workshop on Machine Learning for Spoken Language Understanding and Interaction*.

Chandar, S. A., Lauly, S., Larochelle, H., Khapra, M. M., Ravindran, B., Raykar, V. C., and Saha, A. (2014). An autoencoder approach to learning bilingual word representations. In *Proceedings of NIPS*.

Chen, D. and Manning, C. D. (2014). A fast and accurate dependency parser using neural networks. In *Proceedings of EMNLP*.

Chen, M., Dorer, K., Foroughi, E., Heintz, F., Huang, Z., Kapetanakis, S., Kostiadis, K., Kummeneje, J., Murray, J., Noda, I., Obst, O., Riley, P., Steffens, T., Wang, Y., and Yin, X. (2003). *Users Manual: RoboCup Soccer Server — for Soccer Server Version 7.07 and Later*. The RoboCup Federation.

Cheyer, A. and Guzzoni, D. (2007). Method and apparatus for building an intelligent automated assistant. US Patent Application 11518292.

Clark, R., Richmond, K., and King, S. (2004). Festival 2-build your own general purpose unit selection speech synthesiser. In *Proceedings of the ISCA Workshop on Speech Synthesis*.

Collobert, R. and Weston, J. (2008). A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proceedings of ICML*.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Cotterell, R. and Schütze, H. (2015). Morphological word-embeddings. In *Proceedings of NAACL-HLT*.

Cotterell, R. and Schütze, H. (2017). Joint semantic synthesis and morphological analysis of the derived word. *Transactions of the Association for Computational Linguistics*, 5.

Cotterell, R., Schütze, H., and Eisner, J. (2016). Morphological smoothing and extrapolation of word embeddings. In *Proceedings of ACL*.

Coulmance, J., Marty, J.-M., Wenzek, G., and Benhalloum, A. (2015). Trans-gram, fast cross-lingual word embeddings. In *Proceedings of EMNLP*.

Creutz, M. and Lagus, K. (2007). Unsupervised models for morpheme segmentation and morphology learning. *TSLP*, 4(1):3:1–3:34.

Curran, J. (2003). *From Distributional to Semantic Similarity*. PhD thesis, School of Informatics, University of Edinburgh.

Dahl, G. E., Yu, D., Deng, L., and Acero, A. (2012). Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition. *Transactions of Audio, Speech and Language Processing*, 20(1):30–42.

Davidov, D. and Rappoport, A. (2006). Efficient unsupervised discovery of word categories using symmetric patterns and high frequency words. In *Proceedings of ACL*.

de Marneffe, M.-C. and Manning, C. D. (2008). The Stanford Typed Dependencies Representation. In *Proceedings of COLING*.

de Marneffe, M.-C., Rafferty, A. N., and Manning, C. D. (2008). Finding contradictions in text. In *Proceedings of ACL*.

Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407.

Deng, L., Li, J., Huang, J.-T., Yao, K., Yu, D., Seide, F., Seltzer, M. L., Zweig, G., He, X., Williams, J., Gong, Y., and Acero, A. (2013). Recent advances in deep learning for speech research at Microsoft. In *Proceedings of ICASSP*.

DeVault, D. and Stone, M. (2007). Managing ambiguities across utterances in dialogue. In *Proceedings of the Workshop on the Semantics and Pragmatics of Dialogue*.

Devlin, J., Zbib, R., Huang, Z., Lamar, T., Schwartz, R. M., and Makhoul, J. (2014). Fast and robust neural network joint models for statistical machine translation. In *Proceedings of ACL*.

Dhillon, P. S., Foster, D. P., and Ungar, L. H. (2015). Eigenwords: Spectral word embeddings. *Journal of Machine Learning Research*, 16:3035–3078.

Dinu, G., Lazaridou, A., and Baroni, M. (2015). Improving zero-shot learning by mitigating the hubness problem. In *Proceedings of ICLR: Workshop Papers*.

Duchi, J. C., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159.

Duong, L., Kanayama, H., Ma, T., Bird, S., and Cohn, T. (2016). Learning crosslingual word embeddings without bilingual corpora. In *Proceedings of EMNLP*.

Ehrmann, M., Cecconi, F., Vannella, D., Mccrae, J. P., Cimiano, P., and Navigli, R. (2014). Representing multilingual data as linked data: The case of BabelNet 2.0. In *Proceedings of LREC*.

Faruqui, M., Dodge, J., Jauhar, S. K., Dyer, C., Hovy, E., and Smith, N. A. (2015). Retrofitting word vectors to semantic lexicons. In *Proceedings of NAACL*.

Faruqui, M. and Dyer, C. (2014). Improving vector space word representations using multilingual correlation. In *Proceedings of EACL*.

Faruqui, M. and Dyer, C. (2015). Non-distributional word vector representations. In *Proceedings of ACL*.

Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., and Ruppin, E. (2002). Placing search in context: The concept revisited. *ACM Transactions on Information Systems*, 20(1):116–131.

Firth, J. R. (1957). A synopsis of linguistic theory 1930-55. 1952-59:1–32.

Fraser, N. M. and Gilbert, G. N. (1991). Simulating speech systems. *Computer Speech and Language*, 5(1):81–99.

Fromkin, V., Rodman, R., and Hyams, N. (2013). *An Introduction to Language, 10th Edition*.

Gales, M. and Young, S. (2007). The Application of Hidden Markov Models in Speech Recognition. *Foundations and Trends in Signal Processing*, 1(3).

Ganitkevitch, J. and Callison-Burch, C. (2014). The Multilingual Paraphrase Database. In *Proceedings of LREC*.

Ganitkevitch, J., Durme, B. V., and Callison-burch, C. (2013). PPDB: The Paraphrase Database. In *Proceedings of NAACL*.

Gašić, M., Breslin, C., Henderson, M., Kim, D., Szummer, M., Thomson, B., Tsiakoulis, P., and Young, S. (2013). POMDP-based dialogue manager adaptation to extended domains. In *Proceedings of SIGDIAL*.

Gašić, M., Henderson, M., Thomson, B., Tsiakoulis, P., and Young, S. (2010). Policy optimisation of pomdp-based dialogue systems without state space compression. In *IEEE Spoken Language Technology Workshop*.

Gašić, M., Kim, D., Tsiakoulis, P., Breslin, C., Henderson, M., Szummer, M., Thomson, B., and Young, S. (2014). Incremental on-line adaptation of POMDP-based dialogue managers to extended domains. In *Proceedings of Interspeech*.

Gašić, M. and Young, S. (2011). Effective Handling of Dialogue State in the Hidden Information State POMDP-based Dialogue Manager. *ACM Transactions on Speech and Language Processing (TSLP)*, 7(3):4:1–4:28.

Gerz, D., Vulić, I., Hill, F., Reichart, R., and Korhonen, A. (2016). SimVerb-3500: A large-scale evaluation set of verb similarity. In *Proceedings of EMNLP*.

Ghoshal, A., Swietojanski, P., and Renals, S. (2013). Multilingual training of deep neural networks. In *Proceedings of ICASSP*.

Glorot, X. and Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of AISTATS*.

Glorot, X., Bordes, A., and Bengio, Y. (2011). Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach. In *Proceedings of ICML*.

Goddeau, D., Meng, H., Polifroni, J., Seneff, S., and Busayapongchai, S. (1996). A form-based dialogue manager for spoken language applications. In *Proceedings of ICSLP*.

Goldberg, Y. (2015). A primer on neural network models for natural language processing. *CoRR*, abs/1510.00726.

Gouws, S., Bengio, Y., and Corrado, G. (2015). BilBOWA: Fast bilingual distributed representations without word alignments. In *Proceedings of ICML*.

Graves, A. (2012). *Supervised Sequence Labelling with Recurrent Neural Networks*. Springer, Berlin.

Graves, A. and Jaitly, N. (2014). Towards End-to-End Speech Recognition with Recurrent Neural Networks. In *Proceedings of ICML*.

Guo, J., Che, W., Wang, H., and Liu, T. (2014). Revisiting embedding features for simple semi-supervised learning. In *Proceedings of EMNLP*.

Guo, J., Che, W., Yarowsky, D., Wang, H., and Liu, T. (2015). Cross-lingual dependency parsing based on distributed representations. In *Proceedings of ACL*.

Han, K. J., Chandrashekaran, A., Kim, J., and Lane, I. R. (2018). The CAPIO 2017 conversational speech recognition system. *CoRR*, abs/1801.00059.

Harris, Z. (1954). Distributional Structure. *Word*, 10(23):146–162.

Hashimoto, C., Torisawa, K., De Saeger, S., Oh, J.-H., and Kazama, J. (2012). Excitatory or inhibitory: A new semantic orientation extracts contradiction and causality from the Web. In *Proceedings of EMNLP-CoNLL*.

Haspelmath, M. and Sims, A. (2013). *Understanding morphology*.

He, Y. and Young, S. (2005). Semantic processing using the Hidden Vector State model. *Computer Speech and Language*, 19(1):85 – 106.

Hemphill, C. T., Godfrey, J. J., and Doddington, G. R. (1990). The ATIS Spoken Language Systems Pilot Corpus. In *Proceedings of the Workshop on Speech and Natural Language*.

Henderson, J. and Lemon, O. (2008). Mixture Model POMDPs for Efficient Handling of Uncertainty in Dialogue Management. In *Proceedings of ACL*.

Henderson, M. (2015a). *Discriminative Methods for Statistical Spoken Dialogue Systems*. PhD thesis, University of Cambridge.

Henderson, M. (2015b). Machine learning for dialog state tracking: A review. In *Proceedings of The First International Workshop on Machine Learning in Spoken Language Processing*.

Henderson, M. (2015c). Machine Learning for Dialog State Tracking: A Review. In *Proceedings of The First International Workshop on Machine Learning in Spoken Language Processing*.

Henderson, M., Gašić, M., Thomson, B., Tsiakoulis, P., Yu, K., and Young, S. (2012a). Discriminative Spoken Language Understanding Using Word Confusion Networks. In *IEEE Spoken Language Technology Workshop*.

Henderson, M., Gašić, M., Thomson, B., Tsiakoulis, P., Yu, K., and Young, S. (2012b). Discriminative Spoken Language Understanding Using Word Confusion Networks. In *IEEE Spoken Language Technology Workshop*.

Henderson, M., Thomson, B., and Wiliams, J. D. (2014a). The Second Dialog State Tracking Challenge. In *Proceedings of SIGDIAL*.

Henderson, M., Thomson, B., and Wiliams, J. D. (2014b). The Third Dialog State Tracking Challenge. In *IEEE Spoken Language Technology Workshop*.

Henderson, M., Thomson, B., and Young, S. (2013). Deep Neural Network Approach for the Dialog State Tracking Challenge. In *Proceedings of SIGDIAL*.

Henderson, M., Thomson, B., and Young, S. (2014c). Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised adaptation. In *IEEE Spoken Language Technology Workshop*.

Henderson, M., Thomson, B., and Young, S. (2014d). Word-based dialog state tracking with recurrent neural networks. In *Proceedings of SIGDIAL*.

Hermann, K. M. and Blunsom, P. (2014a). Multilingual Distributed Representations without Word Alignment. In *Proceedings of ICLR*.

Hermann, K. M. and Blunsom, P. (2014b). Multilingual models for compositional distributed semantics. In *Proceedings of ACL*.

Hill, F., Cho, K., Jean, S., Devin, C., and Bengio, Y. (2014). Embedding word similarity with neural machine translation. *CoRR*, abs/1412.6448.

Hill, F., Reichart, R., and Korhonen, A. (2015). SimLex-999: Evaluating semantic models with (genuine) similarity estimation. *Computational Linguistics*, 41(4):665–695.

Hinton, G., Deng, L., Yu, D., Dahl, G., rahman Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T., and Kingsbury, B. (2012). Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups. *IEEE Signal Processing Magazine*, 29(6):82–97.

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Hopfield, J. (1982). Neural Networks and Physical Systems with Emergent Collective Computational Abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558.

Huang, K., Gardner, M., Papalexakis, E., Faloutsos, C., Sidiropoulos, N., Mitchell, T., Talukdar, P. P., and Fu, X. (2015). Translation invariant word embeddings. In *Proceedings of EMNLP*.

Iyyer, M., Boyd-Graber, J., Claudino, L., Socher, R., and Daumé III, H. (2014). A Neural Network for Factoid Question Answering over Paragraphs. In *Proceedings of EMLNP*.

Jain, L. and Medsker, L. (1999). *Recurrent Neural Networks: Design and Applications*. CRC Press, Boca Raton, Florida, USA, 1st edition.

Jauhar, S. K., Dyer, C., and Hovy, E. H. (2015). Ontologically grounded multi-sense representation learning for semantic vector space models. In *Proceedings of NAACL*.

Johannsen, A., Martínez Alonso, H., and Søgaard, A. (2015). Any-language frame-semantic parsing. In *Proceedings of EMNLP*.

Jones, M. N. and Mewhort, D. J. K. (2007). Representing Word Meaning and Order Information in a Composite Holographic Lexicon. *Psychological Review*, 114(1):1–37.

Jurafsky, D. and Martin, J. H. (2009). *Speech and Language Processing (2nd Edition)*. Prentice-Hall, New Jersey, USA.

Jurčíček, F., Gašić, M., Keizer, S., Mairesse, F., Thomson, B., and Young, S. (2009). Transformation-based Learning for Semantic Parsing. In *Proceedings of Interspeech*.

Jurčíček, F., Thomson, B., and Young, S. (2011). Natural Actor and Belief Critic: Reinforcement Algorithm for Learning Parameters of Dialogue Systems Modelled As POMDPs. *ACM Transactions on Speech and Language Processing*, 7(3):6:1–6:26.

Kalchbrenner, N., Grefenstette, E., and Blunsom, P. (2014). A Convolutional Neural Network for Modelling Sentences. In *Proceedings of ACL*.

Kanerva, P., Kristoferson, J., and Holst, A. (2000). Random Indexing of Text Samples for Latent Semantic Analysis. In *Proceedings of the Annual Conference of the Cognitive Science Society*.

Kannan, A. and Vinyals, O. (2017). Adversarial Evaluation of Dialogue Models. *CoRR*, abs/1701.08198v1.

Kate, R. J. and Wong, Y. W. (2010). Semantic Parsing: The Task, the State of the Art and the Future. ACL 2010 Tutorial.

Kiela, D., Hill, F., and Clark, S. (2015). Specializing word embeddings for similarity or relatedness. In *Proceedings of EMNLP*.

Kim, J.-K., de Marneffe, M.-C., and Fosler-Lussier, E. (2016a). Adjusting word embeddings with semantic intensity orders. In *Proceedings of the 1st Workshop on Representation Learning for NLP*.

Kim, J.-K., Tur, G., Celikyilmaz, A., Cao, B., and Wang, Y.-Y. (2016b). Intent detection using semantically enriched word embeddings. In *IEEE Spoken Language Technology Workshop*.

Kim, S., D'Haro, L. F., Banchs, R. E., Williams, J., and Henderson, M. (2016c). The Fourth Dialog State Tracking Challenge. In *Proceedings of IWSDS*.

Kim, S., D'Haro, L. F., Banchs, R. E., Williams, J., Henderson, M., and Yoshino, K. (2016d). The Fifth Dialog State Tracking Challenge. In *IEEE Spoken Language Technology Workshop*.

Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of EMNLP*.

Kingma, D. P. and Ba, J. (2015). Adam: A Method for Stochastic Optimization. In *Proceedings of ICLR*.

Klementiev, A., Titov, I., and Bhattarai, B. (2012). Inducing crosslingual distributed representations of words. In *Proceedings COLING*, pages 1459–1474.

Koehn, P. (2005). Europarl: A parallel corpus for statistical machine translation. In *MT summit, volume 5*.

Kohonen, T., Schroeder, M. R., and Huang, T. S., editors (2001). *Self-Organizing Maps*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 3rd edition.

Kurose, J. F. and Ross, K. (2002). *Computer Networking: A Top-Down Approach Featuring the Internet, Second Edition*. Addison-Wesley Longman Publishing, Boston, MA, USA.

Kutuzov, A. and Andreev, I. (2015). Texts in, meaning out: neural language models in semantic similarity task for Russian. In *Proceedings of DIALOG*.

Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., and Steedman, M. (2011). Lexical Generalization in CCG Grammar Induction for Semantic Parsing. In *Proceedings of EMNLP*.

Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*.

Landauer, T., Foltz, P., and Laham, D. (1998). An introduction to latent semantic analysis. *Discourse Processes*, 25:259–284.

Larsson, S. and Traum, D. R. (2000). Information State and Dialogue Management in the TRINDI Dialogue Move Engine Toolkit. *Journal of Natural Language Engineering*, 6(3-4):323–340.

Lazaridou, A., Dinu, G., and Baroni, M. (2015). Hubness and pollution: Delving into cross-space mapping for zero-shot learning. In *Proceedings of ACL*.

Lazaridou, A., Marelli, M., Zamparelli, R., and Baroni, M. (2013). Compositionally derived representations of morphologically complex words in distributional semantics. In *Proceedings of ACL*.

Lee, B.-J. and Kim, K.-E. (2016). Dialog History Construction with Long-Short Term Memory for Robust Generative Dialog State Tracking. *Dialogue & Discourse*, 7(3):47–64.

Lee, B.-J., Lim, W., Kim, D., and Kim, K.-E. (2014). Optimizing Generative Dialog State Tracker via Cascading Gradient Descent. In *Proceedings of SIGDIAL*.

Lee, S. (2013). Structured Discriminative Model For Dialog State Tracking. In *Proceedings of SIGDIAL*.

Lee, S. and Eskenazi, M. (2013). Recipe For Building Robust Spoken Dialog State Trackers: Dialog State Tracking Challenge System Description. In *Proceedings of SIGDIAL*.

Leviant, I. and Reichart, R. (2015). Separated by an Un-common Language: Towards Judgment Language Informed Vector Space Modeling. *CoRR*, abs/1508.00106.

Levin, E. and Pieraccini, R. (1995). Chronus, the next generation. In *Proceedings of the ARPA Workshop on Spoken Language Technology*.

Levy, O. and Goldberg, Y. (2014). Dependency-based word embeddings. In *Proceedings of ACL*.

Levy, O., Goldberg, Y., and Dagan, I. (2015). Improving Distributional Similarity with Lessons Learned from Word Embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

Li, J. and Jurafsky, D. (2015). Do Multi-Sense Embeddings Improve Natural Language Understanding? In *Proceedings of EMNLP*.

Lin, D., Zhao, S., Qin, L., and Zhou, M. (2003). Identifying synonyms among distributionally similar words. In *Proceedings of IJCAI*.

Liu, B. and Lane, I. (2016a). Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling. In *Proceedings of Interspeech*.

Liu, B. and Lane, I. (2016b). Joint Online Spoken Language Understanding and Language Modeling with Recurrent Neural Networks. In *Proceedings of SIGDIAL*.

Liu, F. and Perez, J. (2017). Gated End-to-End Memory Networks. In *Proceedings of EACL*.

Liu, Q., Jiang, H., Wei, S., Ling, Z.-H., and Hu, Y. (2015). Learning semantic word embeddings based on ordinal knowledge constraints. In *Proceedings of ACL*.

Lowe, R., Pow, N., Serban, I. V., and Pineau, J. (2015). The Ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Proceedings of SIGDIAL*.

Lucas, B. (2000). "VoiceXML. In *Communications of the ACM*.

Lund, K. and Burgess, C. (1996). Producing High-Dimensional Semantic Spaces from Lexical Co-Occurrence. *Behavior Research Methods, Instruments, & Computers*, 28(2):203–208.

Luong, T., Pham, H., and Manning, C. D. (2015a). Bilingual word representations with monolingual quality in mind. In *Proceedings of the 1st Workshop on Vector Space Modeling for NLP*.

Luong, T., Pham, H., and Manning, C. D. (2015b). Bilingual word representations with monolingual quality in mind. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*.

Luong, T., Socher, R., and Manning, C. (2013). Better word representations with recursive neural networks for morphology. In *Proceedings of CoNLL*.

Ma, Y. and Fosler-Lussier, E. (2014). A discriminative sequence model for dialog state tracking using user goal change detection. In *IEEE Spoken Language Technology Workshop*.

Mairesse, F., Gašić, M., Jurčíček, F., Keizer, S., Thomson, B., Yu, K., and Young, S. (2009). Spoken Language Understanding from Unaligned Data using Discriminative Classification Models. In *Proceedings of ICASSP*.

Mairesse, F., Walker, M. A., Mehl, M. R., and Moore, R. K. (2007). Using Linguistic Cues for the Automatic Recognition of Personality in Conversation and Text. *Journal of Artificial Intelligence Research*, 30(1):457–500.

Mairesse, F. and Young, S. (2014). Stochastic Language Generation in Dialogue Using Factored Language Models. *Computational Linguistics*, 40(4):763–799.

Mangu, L., Brill, E., and Stolcke, A. (2000). Finding Consensus Among Words: Lattice-Based Word Error Minimization. In *Computer Speech and Language*.

Marcu, D. and Echihabi, A. (2002). An unsupervised approach to recognizing discourse relations. In *Proceedings of ACL*.

Margolis, A., Livescu, K., and Ostendorf, M. (2010). Domain Adaptation with Unlabeled Data for Dialog Act Tagging. In *Proceedings of the ACL Workshop on Domain Adaptation*.

McClosky, D., Charniak, E., and Johnson, M. (2006). Effective self-training for parsing. In *Proceedings of HLT-NAACL*.

McClosky, D., Charniak, E., and Johnson, M. (2010). Automatic domain adaptation for parsing. In *Proceedings of NAACL*.

McTear, M. F. (2002). Modelling spoken dialogues with state transition diagrams: Experiences with the CSLU toolkit. In *Proceedings of ICSLP*.

Mehta, N., Gupta, R., Raux, A., Ramachandran, D., and Krawczyk, S. (2010). Probabilistic Ontology Trees for Belief Tracking in Dialog Systems. In *Proceedings SIGDIAL*.

Mesnil, G., Dauphin, Y., Yao, K., Bengio, Y., Deng, L., Hakkani-Tur, D., He, X., Heck, L., Yu, D., and Zweig, G. (2015). Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(3):530–539.

Metallinou, A., Bohus, D., and Williams, J. (2013). Discriminative State Tracking For Spoken Dialog Systems. In *Proceedings of ACL*.

Miao, Y., Gowayyed, M., and Metze, F. (2015). EESEN: End-to-End Speech Recognition using Deep RNN Models and WFST-based Decoding. *CoRR*, abs/1507.08240.

Mikolov, T., Le, Q. V., and Sutskever, I. (2013a). Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*.

Miller, G. A. (1995). WordNet: A lexical database for English. *Communications of the ACM*, pages 39–41.

Mitra, B., Nalisnick, E. T., Craswell, N., and Caruana, R. (2016). A dual embedding space model for document ranking. *CoRR*, abs/1602.01137.

Mohammad, S., Dorr, B., and Hirst, G. (2008). Computing word-pair antonymy. In *Proceedings of EMNLP*.

Mohammad, S. M., Dorr, B. J., Hirst, G., and Turney, P. D. (2013). Computing lexical contrast. *Computational Linguistics*, 39(3):555–590.

Mrkšić, N., Ó Séaghdha, D., Thomson, B., Gašić, M., Rojas-Barahona, L., Su, P.-H., Vandyke, D., Wen, T.-H., and Young, S. (2016). Counter-fitting word vectors to linguistic constraints. In *Proceedings of NAACL*.

Mrkšić, N., Ó Séaghdha, D., Thomson, B., Gašić, M., Su, P.-H., Vandyke, D., Wen, T.-H., and Young, S. (2015). Multi-domain dialog state tracking using recurrent neural networks. In *Proceedings of ACL*.

Mrkšić, N., Ó Séaghdha, D., Thomson, B., Wen, T.-H., and Young, S. (2017a). Neural Belief Tracker: Data-driven dialogue state tracking. In *Proceedings of ACL*.

Mrkšić, N. and Vulić, I. (2018). Fully statistical neural belief tracking. In *Proceedings of ACL*.

Mrkšić, N., Vulić, I., Ó Séaghdha, D., Leviant, I., Reichart, R., Gašić, M., Korhonen, A., and Young, S. (2017b). Semantic specialisation of distributional word vector spaces using monolingual and cross-lingual constraints. *Transactions of the Association for Computational Linguistics*, 5:309–324.

Murveit, H., Butzberger, J., Digalakis, V., and Weintraub, M. (1993). Large-vocabulary Dictation Using SRI's DECIPHERTM Speech Recognition System: Progressive Search Techniques. In *Proceedings of ICASSP*.

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted Boltzmann machines. In *Proceedings of ICML*.

Navigli, R. and Ponzetto, S. P. (2012). BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

Nguyen, K. A., Schulte im Walde, S., and Vu, N. T. (2016). Integrating distributional lexical contrast into word embeddings for antonym-synonym distinction. In *Proceedings of ACL*.

Nivre, J., de Marneffe, M.-C., Ginter, F., Goldberg, Y., Hajic, J., Manning, C. D., McDonald, R., Petrov, S., Pyysalo, S., Silveira, N., Tsarfaty, R., and Zeman, D. (2016). Universal Dependencies v1: A Multilingual Treebank Collection. In *Proceedings of LREC*.

Ono, M., Miwa, M., and Sasaki, Y. (2015). Word Embedding-based Antonym Detection using Thesauri and Distributional Information. In *Proceedings of NAACL*.

Osborne, D., Narayan, S., and Cohen, S. (2016). Encoding prior knowledge with eigenword embeddings. *Transactions of the Association for Computational Linguistics*, 4:417–430.

Ó Séaghdha, D. and Korhonen, A. (2014). Probabilistic distributional semantics. *Computational Linguistics*, 40(3):587–631.

Pavlick, E., Rastogi, P., Ganitkevich, J., Durme, B. V., and Callison-Burch, C. (2015). PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of ACL*.

Peng, B., Yao, K., Jing, L., and Wong, K.-F. (2015). Recurrent Neural Networks with External Memory for Language Understanding. In *Proceedings of the National CCF Conference on Natural Language Processing and Chinese Computing*.

Pennington, J., Socher, R., and Manning, C. (2014). GloVe: Global vectors for word representation. In *Proceedings of EMNLP*.

Perez, J. and Liu, F. (2017). Dialog state tracking, a machine reading approach using Memory Network. In *Proceedings of EACL*.

Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., Silovsky, J., Stemmer, G., and Vesely, K. (2011). The Kaldi Speech Recognition Toolkit. In *Proceedings of ASRU*.

Qiu, S., Cui, Q., Bian, J., Gao, B., and Liu, T.-Y. (2014). Co-learning of word representations and morpheme representations. In *Proceedings of COLING*.

Rastogi, P., Durme, B. V., and Arora, R. (2015). Multiview LSA: Representation learning via generalized CCA. In *Proceedings of NAACL*.

Raux, A. and Ma, Y. (2011). Efficient Probabilistic Tracking of User Goal and Dialog History for Spoken Dialog Systems. In *Proceedings of Interspeech*.

Raymond, C. and Ricardi, G. (2007). Generative and discriminative algorithms for spoken language understanding. In *Proceedings of Interspeech*.

Recski, G., Iklódi, E., Pajkossy, K., and Kornai, A. (2016). Measuring Semantic Similarity of Words Using Concept Networks. In *Proceedings of the 1st Workshop on Representation Learning for NLP*.

Reddy, S., Täckström, O., Collins, M., Kwiatkowski, T., Das, D., Steedman, M., and Lapata, M. (2016). Transforming Dependency Structures to Logical Forms for Semantic Parsing. *Transactions of the Association for Computational Linguistics*, 4.

Reiter, E. and Dale, R. (2000). *Building Natural Language Generation Systems*. Cambridge University Press, New York, NY, USA.

Ren, H., Xu, W., and Yan, Y. (2014a). Markovian discriminative modeling for cross-domain dialog state tracking. In *IEEE Spoken Language Technology Workshop*.

Ren, H., Xu, W., and Yan, Y. (2014b). Markovian discriminative modeling for cross-domain dialog state tracking. In *IEEE Spoken Language Technology Workshop*.

Ren, H., Xu, W., Zhang, Y., and Yan, Y. (2013). Dialog State Tracking using Conditional Random Fields. In *Proceedings of SIGDIAL*.

Rieser, V. and Lemon, O. (2010). Natural language generation as planning under uncertainty for spoken dialogue systems. In *Proceedings of EMNLP*.

Rocktäschel, T., Grefenstette, E., Hermann, K. M., Kocisky, T., and Blunsom, P. (2016). Reasoning about entailment with neural attention. In *Proceedings of ICLR*.

Rocktäschel, T., Grefenstette, E., Hermann, K. M., Kočiský, T., and Blunsom, P. (2016). Reasoning about Entailment with Neural Attention. In *Proceedings of ICLR*.

Rothe, S. and Schütze, H. (2015). AutoExtend: Extending word embeddings to embeddings for synsets and lexemes. In *Proceedings of ACL*.

Roy, N., Pineau, J., and Thrun, S. (2000). Spoken dialogue management using probabilistic reasoning. In *Proceedings of ACL*.

Saleh, I., Joty, S., Màrquez, L., Moschitti, A., Nakov, P., Cyphers, S., and Glass, J. (2014). A study of using syntactic and semantic structures for concept segmentation and labeling. In *Proceedings of COLING*.

Saon, G., Kurata, G., Sercu, T., Audhkhasi, K., Thomas, S., Dimitriadis, D., Cui, X., Ramabhadran, B., Picheny, M., Lim, L., Roomi, B., and Hall, P. (2017). English Conversational Telephone Speech Recognition by Humans and Machines, booktitle = Proceedings of Interspeech.

Schone, P. and Jurafsky, D. (2001). Knowledge-free induction of inflectional morphologies. In *Proceedings of NAACL*.

Schütze, H. (1993). Word Space. In *Proceedings of NIPS*.

Schwartz, R., Reichart, R., and Rappoport, A. (2015). Symmetric pattern based word embeddings for improved word similarity prediction. In *Proceedings of CoNLL*.

Serban, I. V., Sordoni, A., Bengio, Y., Courville, A., and Pineau, J. (2016a). Building End-to-end Dialogue Systems Using Generative Hierarchical Neural Network Models. In *Proceedings of AAAI*.

Serban, I. V., Sordoni, A., Lowe, R., Charlin, L., Pineau, J., Courville, A., and Bengio, Y. (2016b). A Hierarchical Latent Variable Encoder-Decoder Model for Generating Dialogues. *CoRR*, abs/1605.06069.

Singh, S., Kearns, M., Litman, D., and Walker, M. (1999). Reinforcement learning for spoken dialogue systems. In *Advances in Neural Information Processing Systems*.

Socher, R., Bauer, J., Manning, C. D., and Ng, A. Y. (2013a). Parsing with compositional vector grammars. In *Proceedings of ACL*.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. (2013b). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP*.

Søgaard, A., Agić, v., Martínez Alonso, H., Plank, B., Bohnet, B., and Johannsen, A. (2015). Inverted indexing for cross-lingual NLP. In *Proceedings ACL*.

Soyer, H., Stenetorp, P., and Aizawa, A. (2015). Leveraging monolingual data for crosslingual compositional word representations. In *Proceedings of ICLR*.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Stent, A., Prassad, R., and Walker, M. (2004). Trainable sentence planning for complex information presentations in spoken dialog systems. In *Proceedings of ACL*.

Su, P.-H., Budzianowski, P., Ultes, S., Gašić, M., and Young, S. (2017). Sample-efficient Actor-Critic Reinforcement Learning with Supervised Data for Dialogue Management. In *Proceedings of SIGDIAL*.

Su, P.-H., Gašić, M., Mrkšić, N., Rojas-Barahona, L., Ultes, S., Vandyke, D., Wen, T.-H., and Young, S. (2016a). Continuously Learning Neural Dialogue Management. *CoRR*, abs/1606.02689.

Su, P.-H., Gašić, M., Mrkšić, N., Rojas-Barahona, L., Ultes, S., Vandyke, D., Wen, T.-H., and Young, S. (2016b). On-line Active Reward Learning for Policy Optimisation in Spoken Dialogue Systems. In *Proceedings of ACL*.

Sukhbaatar, S., Szlam, A., Weston, J., and Fergus, R. (2015). End-to-end Memory Networks. In *Proceedings of NIPS*.

Sun, K., Chen, L., Zhu, S., and Yu, K. (2014). The SJTU System for Dialog State Tracking Challenge 2. In *Proceedings of SIGDIAL*.

Sun, K., Xie, Q., and Yu, K. (2016). Recurrent Polynomial Network for Dialogue State Tracking. *Dialogue & Discourse*, 7(3):65–88.

Sutton, S., Novick, D., Cole, R., Vermeulen, P., de Villiers, J., Schalkwyk, J., and Fanty, M. (1996). Building 10,000 spoken dialogue systems. In *Proceedings of ICSLP*.

Syed, U. and Williams, J. D. (2008). Using Automatically Transcribed Dialogs to Learn User Models in a Spoken Dialog System. In *Proceedings of ACL*.

Sylak-Glassman, J., Kirov, C., Yarowsky, D., and Que, R. (2015). A language-independent feature schema for inflectional morphology. In *Proceedings of ACL*.

Taylor, P., Black, A. W., and Caley, R. (1998). The architecture of the festival speech synthesis system. In *ESCA Wokshop in Speech Synthesis*.

Thomson, B. (2009). *Statistical methods for spoken dialogue management*. PhD thesis, University of Cambridge.

Thomson, B., Jurčíčcek, F., Gašić, M., Keizer, S., Mairesse, F., Yu, K., and Young, S. (2010). Parameter Learning for POMDP Spoken Dialogue Models. In *IEEE Spoken Language Technology Workshop*.

Thomson, B. and Young, S. (2010). Bayesian Update of Dialogue State: A POMDP Framework for Spoken Dialogue Systems. *Computer Speech and Language*, 24(4):562–588.

Tsarfaty, R., Seddah, D., Goldberg, Y., Kuebler, S., Versley, Y., Candito, M., Foster, J., Rehbein, I., and Tounsi, L. (2010). Statistical parsing of morphologically rich languages (SPMRL) What, how and whither. In *Proceedings of the NAACL Workshop on Statistical Parsing of Morphologically-Rich Languages*.

Tsiakoulis, P., Breslin, C., Gašić, M., Henderson, M., Kim, D., Szummer, M., Thomson, B., and Young, S. (2014). Dialogue context sensitive HMM-based speech synthesis. In *Proceedings of ICASSP*.

Tur, G., Deoras, A., and Hakkani-Tur, D. (2013). Semantic Parsing Using Word Confusion Networks With Conditional Random Fields. In *Proceedings of Interspeech*.

Tur, G., Guz, U., and Hakkani-Tür, D. (2007). Model adaptation for dialog act tagging. In *IEEE Spoken Language Technology Workshop*.

Turian, J., Ratinov, L.-A., and Bengio, Y. (2010). Word representations: A simple and general method for semi-supervised learning. In *Proceedings of ACL*.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, 59:433–460.

Turney, P. D. (2008). A uniform approach to analogies, synonyms, antonyms, and associations. In *Proceedings of COLING*.

Upadhyay, S., Faruqui, M., Dyer, C., and Roth, D. (2016). Cross-lingual models of word embeddings: An empirical comparison. In *Proceedings of ACL*.

Van Deemter, K., Krahmer, E., and Theune, M. (2005). Real Versus Template-Based Natural Language Generation: A False Opposition? *Computational Linguistics*, 31(1):15–24.

Vandyke, D., Su, P., Gašić, M., Mrkšić, N., Wen, T., and Young, S. (2015). Multi-Domain Dialogue Success Classifiers for Policy Training. In *Proceedings of ASRU*.

Vinyals, O. and Le, Q. (2015). A Neural Conversational Model. In *ICML Deep Learning Workshop*.

Vlachos, A. and Clark, S. (2014). A new corpus and imitation learning framework for context-dependent semantic parsing. *Transactions of the Association for Computational Linguistics*, 2:547–559.

Vodolán, M., Kadlec, R., and Kleindienst, J. (2017). Hybrid Dialog State Tracker with ASR Features. In *Proceedings of EACL*.

Vu, N. T., Gupta, P., Adel, H., and Schütze, H. (2016). Bi-directional recurrent neural network with ranking loss for spoken language understanding. In *Proceedings of ICASSP*.

Vulić, I., Gerz, D., Kiela, D., Hill, F., and Korhonen, A. (2016). Hyperlex: A large-scale evaluation of graded lexical entailment. *CoRR*, abs/1608.02117.

Vulić, I. and Korhonen, A. (2016a). Is "universal syntax" universally useful for learning distributed representations? In *Proceedings of ACL*.

Vulić, I. and Korhonen, A. (2016b). On the role of seed lexicons in learning bilingual word embeddings. In *Proceedings of ACL*.

Vulić, I. and Moens, M. (2016). Bilingual distributed word representations from document-aligned comparable data. *Journal of Artificial Intelligence Research*, 55:953–994.

Vulić, I. and Moens, M.-F. (2015). Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. In *Proceedings of SIGIR*.

Vulić, I., Mrkšić, N., Reichart, R., Ó Séaghdha, D., Young, S., and Korhonen, A. (2017). Morph-fitting: Fine-tuning word vector spaces with simple language-specific rules. In *Proceedings of ACL*.

Žilka, L. and Jurčíček, F. (2015). Incremental LSTM-based dialog state tracker. In *Proceedings of ASRU*.

Walker, M., Rambow, O., and Rogati, M. (2001). Spot: A trainable sentence planner. In *Proceedings of NAACL*.

Walker, M., Stent, A., Mairesse, F., and Prasad, R. (2007). Individual and Domain Adaptation in Sentence Planning for Dialogue. *Journal of Artificial Intelligence Research*, 30(1):413–456.

Walker, W., Lamere, P., Kwok, P., Raj, B., Singh, R., Gouvea, E., Wolf, P., and Woelfel, J. (2004). Sphinx-4: A Flexible Open Source Framework for Speech Recognition. Technical report.

Wang, W. (1994). Extracting Information From Spontaneous Speech. In *Proceedings of Interspeech*.

Wang, Z. and Lemon, O. (2013). A Simple and Generic Belief Tracking Mechanism for the Dialog State Tracking Challenge: On the believability of observed information. In *Proceedings of SIGDIAL*.

Ward, W. and Issar, S. (1994). Recent Improvements in the CMU Spoken Language Understanding System. In *Proceedings of the Workshop on Human Language Technology*.

Weizenbaum, J. (1966). ELIZA - a Computer Program for the Study of Natural Language Communication Between Man and Machine. *Communications of the ACM*, 9(1):36–45.

Wen, T.-H., Gašić, M., Kim, D., Mrkšić, N., Su, P.-H., Vandyke, D., and Young, S. (2015a). Stochastic Language Generation in Dialogue using Recurrent Neural Networks with Convolutional Sentence Reranking. In *Proceedings of SIGDIAL*.

Wen, T.-H., Gašić, M., Mrkšić, N., , M. Rojas-Barahona, L., Su, P.-H., Vandyke, D., and Young, S. (2016). Multi-domain Neural Network Language Generation for Spoken Dialogue Systems. In *Proceedings of NAACL-HLT*.

Wen, T.-H., Gašić, M., Mrkšić, N., Su, P.-H., Vandyke, D., and Young, S. (2015b). Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In *Proceedings of EMNLP*.

Wen, T.-H., Vandyke, D., Mrkšić, N., Gašić, M., M. Rojas-Barahona, L., Su, P.-H., Ultes, S., and Young, S. (2017). A Network-based End-to-End Trainable Task-oriented Dialogue System. In *Proceedings of EACL*.

Weston, J., Chopra, S., and Bordes, A. (2014). Memory Networks. *CoRR*, abs/1410.3916.

Wieting, J., Bansal, M., Gimpel, K., and Livescu, K. (2015). From paraphrase database to compositional paraphrase model and back. *Transactions of the Association for Computational Linguistics*, 3:345–358.

Wieting, J., Bansal, M., Gimpel, K., and Livescu, K. (2016). Charagram: Embedding words and sentences via character n-grams. In *Proceedings of EMNLP*.

Williams, J. (2012a). A critical analysis of two statistical spoken dialog systems in public use. In *IEEE Spoken Language Technology Workshop*.

Williams, J., Asadi, K., and Zweig, G. (2017). Hybrid Code Networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. In *Proceedings of ACL*.

Williams, J. D. (2012b). A belief tracking challenge task for spoken dialog systems. In *NAACL-HLT Workshop on Future Directions and Needs in the Spoken Dialog Community: Tools and Data*.

Williams, J. D. (2013). Multi-domain learning and generalization in dialog state tracking. In *Proceedings of SIGDIAL*.

Williams, J. D. (2014). Web-style ranking and SLU combination for dialog state tracking. In *Proceedings of SIGDIAL*.

Williams, J. D., Raux, A., and Henderson, M. (2016). The Dialog State Tracking Challenge series: A review. *Dialogue & Discourse*, 7(3):4–33.

Williams, J. D., Raux, A., Ramachandran, D., and Black, A. W. (2013). The Dialogue State Tracking Challenge. In *Proceedings of SIGDIAL*.

Williams, J. D. and Young, S. (2007). Partially Observable Markov Decision Processes for Spoken Dialog Systems. *Computer Speech and Language*, 21:393–422.

Xiong, W., Wu, L., Alleva, F., Droppo, J., Huang, X., and Stolcke, A. (2017). The Microsoft 2017 Conversational Speech Recognition System. Technical report.

Xu, C., Bai, Y., Bian, J., Gao, B., Wang, G., Liu, X., and Liu, T.-Y. (2014). RC-NET: A general framework for incorporating knowledge into word representations. In *Proceedings of CIKM*.

Yao, K., Peng, B., Zhang, Y., Yu, D., Zweig, G., and Shi, Y. (2014). Spoken language understanding using long short-term memory neural networks. In *Proceedings of ASRU*.

Yih, W., Zweig, G., and Platt, J. C. (2012). Polarity inducing Latent Semantic Analysis. In *Proceedings of ACL*.

Young, S. (2010a). Cognitive User Interfaces. *IEEE Signal Processing Magazine*, 27(3):128–140.

Young, S. (2010b). Still talking to machines (cognitively speaking). In *Proceedings of Interspeech*.

Young, S., Evermann, G., Gales, M., Hain, T., Kershaw, D., Liu, X., Moore, G., Odell, J., Ollason, D., Povey, D., Valtchev, V., and Woodland, P. (2002). The HTK book. *Cambridge University Engineering Department*, 3.

Young, S., Gašić, M., Keizer, S., Mairesse, F., Schatzmann, J., Thomson, B., and Yu, K. (2010). The Hidden Information State Model: A Practical Framework for POMDP-based Spoken Dialogue Management. *Computer Speech and Language*, 24:150–174.

Young, S., Gašić, M., Thomson, B., and Williams, J. D. (2013). POMDP-Based Statistical Spoken Dialog Systems: A Review. *Proceedings of the IEEE*, 101(5):1160–1179.

Young, S. and Proctor, C. (1989). The design and implementation of dialogue control in voice operated database inquiry systems. *Computer Speech and Language*, 3(4):329 – 353.

Young, S., Schatzmann, J., Weilhammer, K., and Ye, H. (2007). The Hidden Information State Approach to Dialog Management. In *Proceedings of ICASSP*.

Yu, M. and Dredze, M. (2014). Improving lexical embeddings with semantic knowledge. In *Proceedings of ACL*.

Zanzotto, F. M., Pennachiotti, M., and Moschitti, A. (2009). A machine learning approach to textual entailment recognition. *Journal of Natural Language Engineering*, 15(4):551–582.

Zeiler, M. D. (2012). ADADELTA: An Adaptive Learning Rate Method. *CoRR*, abs/1212.5701.

Zelle, J. M. and Mooney, R. J. (1996a). Learning to Parse Database Queries Using Inductive Logic Programming. In *Proceedings of AAAI*.

Zelle, J. M. and Mooney, R. J. (1996b). Learning to Parse Database Queries using Inductive Logic Programming. In *Proceedings of AAAI*.

Zeller, B., Šnajder, J., and Padó, S. (2013). DErivBase: Inducing and evaluating a derivational morphology resource for German. In *Proceedings of ACL*.

Zen, H., Nose, T., Yamagishi, J., Sako, S., Masuko, T., Black, A. W., and Tokuda, K. (2007). HMM-based speech synthesis system (HTS) version 2.0. In *Sixth ISCA Workshop on Speech Synthesis*.

Zen, H., Sak, H., Graves, A., and Senior, A. (2014). Statistical Parametric Speech Synthesis based on Recurrent Neural Networks. In *Proceedings of UKSpeech*.

Zen, H., Senior, A., and Schuster, M. (2013). Statistical parametric speech synthesis using deep neural networks. In *Proceedings of ICASSP*.

Zettlemoyer, L. S. and Collins, M. (2007). Online learning of relaxed CCG grammars for parsing to logical form. In *In Proceedings of EMNLP*.

Zhang, X. and Wang, H. (2016). A Joint Model of Intent Determination and Slot Filling for Spoken Language Understanding. In *Proceedings of IJCAI*.

Zou, W. Y., Socher, R., Cer, D., and Manning, C. D. (2013). Bilingual word embeddings for phrase-based machine translation. In *Proceedings of EMNLP*.

Zue, V., Seneff, S., Glass, J., Polifroni, J., Pao, C., Hazen, T. J., and Hetherington, L. (2000). Jupiter: A Telephone-Based Conversational Interface for Weather Information. *IEEE Transactions on Speech and Audio Processing*, 8:85–96.