INDUCTIVE BIASES FOR DEEP LINGUISTIC STRUCTURED PREDICTION VIA INDEPENDENT FACTORIZATION

by

Jie Cao

A dissertation submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

School of Computing

The University of Utah

June 2022

The University of Utah Graduate School

STATEMENT OF DISSERTATION APPROVAL

The dissertation of **Jie Cao**
has been approved by the following supervisory committee members:

Vivek Srikumar ,	Chair(s)	__ Date Approved
Ellen Riloff ,	Member	__ Date Approved
Qingyao Ai ,	Member	__ Date Approved
Zac Imel ,	Member	__ Date Approved
Yi Zhang ,	Member	__ Date Approved

by **xx**, Chair/Dean of
the Department/College/School of **Computing**
and by **xx**, Dean of The Graduate School.

ABSTRACT

Discovering the underlying structure of unstructured text can help make sense of the rapidly growing data. This thesis studies helpful inductive biases for designing deep learning models for natural language structured prediction. Towards generalization over new, previously unseen data, the search for appropriate inductive biases is necessary for any machine learning based natural language processing system. This is also true for deep learning models to predict complicated combinatory structures.

In this thesis, we primarily focus on studying deep linguistic structured prediction via independent factorization. We propose two kinds of generic inductive biases to enchance the independent factorization, including ***Structural Inductive Biases*** and ***Natural Language as Inductive Biases***. We ground our studies on both broad-coverage linguistic representations and application-specific representations.

Due to the compositionality of natural language, these language representations are defined to be compositional structures. We study structural inductive biases by designing factorization-oriented learning and reasoning mechanisms at the lexical, phrasal, and sentential levels. Furthermore, knowledge is often encoded as human language. Taking unannotated natural language as a source of supervision, we study task-oriented dialogue state tracking by describing the intents and their argument slots in natural language. We offer comparative studies showing how such inductive biases help generalize to new domains and APIs.

In all cases, based on independent factorization, the experimental results show our proposed inductive biases achieve competitive performance for each task. We expect that the structural and natural language inductive biases studied in this work can potentially help other linguistic structured prediction tasks via independent factorization.

For my parents, and my love.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Human language is essential for human intelligence, and arguably our most powerful tool for the learning and transmission of knowledge. With the advances of computers and the internet, most of the world's knowledge, such as conversations, scholarly research, factual news, online education, and private mental health records, is now easily accessible as digitized text. However, with limited ability of information processing, we cannot easily discover the knowledge hidden in the huge amount of unstructured text.

One classical way to study unstructured natural language is to represent the language via various structured symbolic representations in different levels (Smith, 2011). Before the revolution of representation learning with deep learning, the NLP community has put decades of effort into solving different lingustic structured prediction tasks to get various aspects of text understanding. Let us loot at some examples.

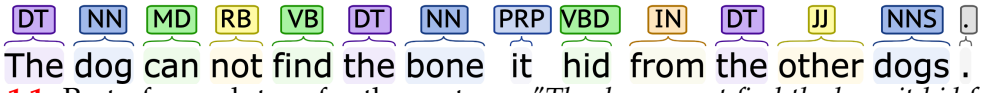

Figure 1.1: Part-of-speech tags for the sentence *"The dog cannot find the bone it hid from the other dogs."* This image shows the tag set used in Penn Treebank Marcus et al. (1994)

Example 1: Part-of-Speech Tagging, Constituency and Dependency Structure. We consider the sentence "The dog cannot find the bone it hid from the other dogs" as a running example. As shown in Figure 1.1, the part-of-speech (**POS**) tagging assigns each word in a sentence a part-of-speech tag, such as NOUN, VERB,ADJECTIVE, PRONOUN. How to capture the sequential correlations between consecutive tags is the key modelling challenge for this task. Figure 1.2 shows the **constituent tree** structure of the sentence. The constituent tree parsing requires recognizing the recursive phrase structure of a sentence, such as noun, verb, prepositional phrases, and their nesting in each other. Figure 1.3 shows the **dependency tree** structure of the sentence. Unlike the constituency structure, here the

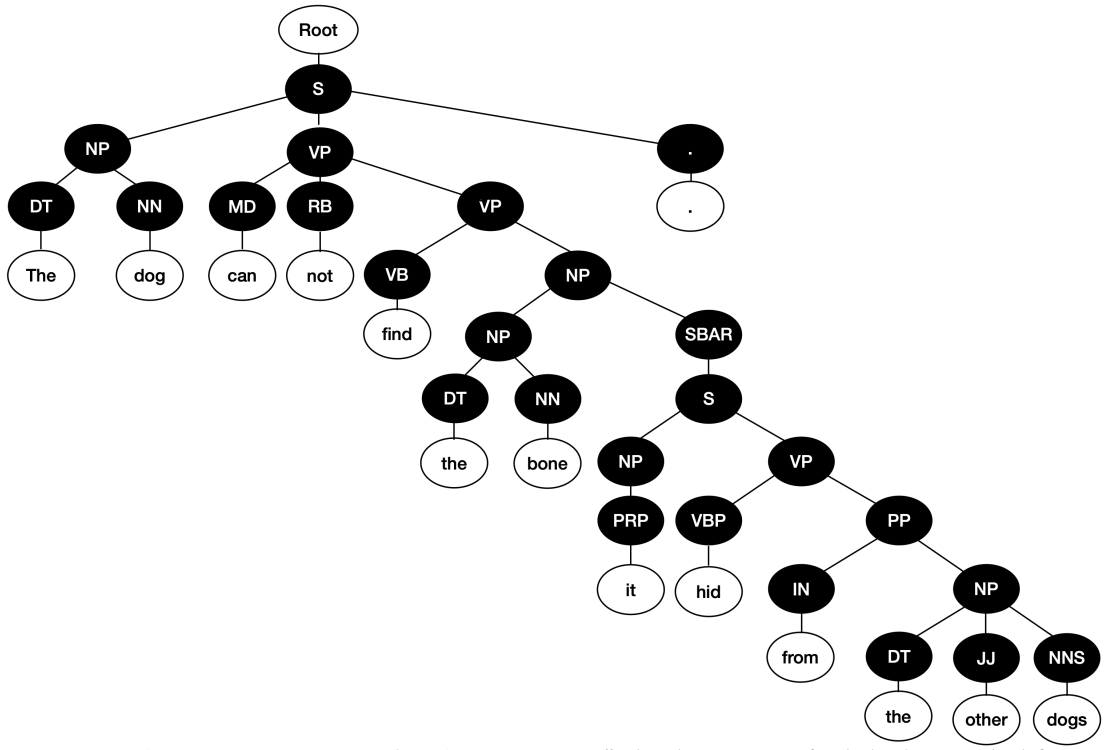

Figure 1.2: The constituent tree for the sentence *"The dog cannot find the bone it hid from the other dogs"*

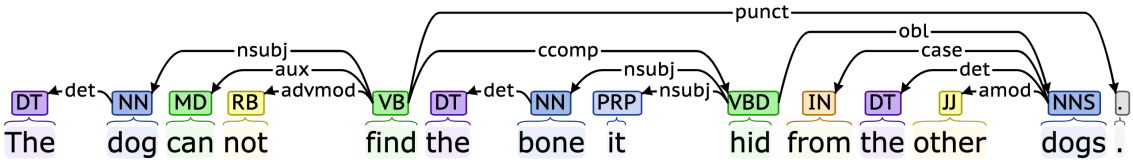

Figure 1.3: The dependency tree, for the sentence *"The dog cannot find the bone it hid from the other dogs"*

syntactic structure of a sentence is described in terms of the directed bi-lexical grammatical relations between words. Each labeled arc represents a directed relation from head words to their dependents. Besides the above lexical and syntactic structured information, as shown in the left part of Figure 1.4, natural language semantics is also widely studied as structured representations, via tasks such as **word sense diambiguation**, **semantic role labeling** and **co-reference resolution** and so on [1]. Such structured information is widely used in classical feature-engineering based NLP system (e.g., Johansson and Nugues, 2008; Hovy et al., 2010; Punyakanok et al., 2008), they are still helpful in deep learning based systems (Moosavi and Strube, 2018; Strubell et al., 2018; Bowman et al., 2016).

[1] More details about various semantic phenomenons will be introduced in Section 2.1.1

Example 2: Broad-coverage Meaning Representation. Besides the above structures capturing specific lexical, syntactic or semantic information, a broad-coverage semantic representation is a general-purpose meaning representation langauge aiming to represent the multiple phenomena in a single structure for broad-coverage text. Figure 1.4 shows the Abstract Meaning Representation (**AMR**, Banarescu et al., 2013a) of the example sentence. The node in the graph represent abstract concepts [2], and the labeled edges between the nodes represent the relations between those concepts. As shown in Figure 1.2, the node 'find–01' and 'hide-01' represent the word sense predefined in the Propbank Kingsbury and Palmer (2002); The connected edges ':ARG0' and ':ARG1' captures the semantic roles that can be derived from the semantic role labelling tasks; While the node 'dog/d1' means the subject for events 'find-01', 'hide-01' and 'possible-01' are the same dog, thus capturing the correference information. Figure 1.5 shows the foundational layer of Universal Conceptual Cognitive Annotation (**UCCA**, Abend and Rappoport, 2013a), which is a multi-layered framework for semantic representation that aims to accommodate the semantic distinctions in the sentence and support open-ended extensions. Different from AMR, this UCCA foundational layer mainly forms a tree-like structure, which focuses on argument structures of verbal, nominal and adjectival predicates and the inter-relations between them. Besides the above two broad-coverage meaning representations, we also studied the DELPH-IN MRS Bi-lexical Dependencies (DM, Ivanova et al., 2012a) and Prague Semantic Dependencies (PSD, Hajic et al., 2012; Miyao et al., 2014). More details about their captured semantic content and their structure properties will introduced comparatively in Section 2.1.1.

Example 3: Application-specific Symbolic Representation. Besides the above broad coverage syntactic and semantic structures in natural language, researchers have designed various symbolic representations for specific applications. Dialog acts are firstly designed to represent the speech act or intention of each utterance, in order to represent the functions of each utterance in the dialogue (Wittgenstein, 2010; Bunt et al., 2010). Then inspired by the case theory (Fillmore, 1968), frame-based representation in GUS (Bobrow et al., 1977) are introduced to represent the state of dialogue, which consists of a collection of slots and

[2] AMR concepts includes PropBank framesets, and other special date, and spatial entities, etc. More details about AMR will be introduced in Section 2.1.1

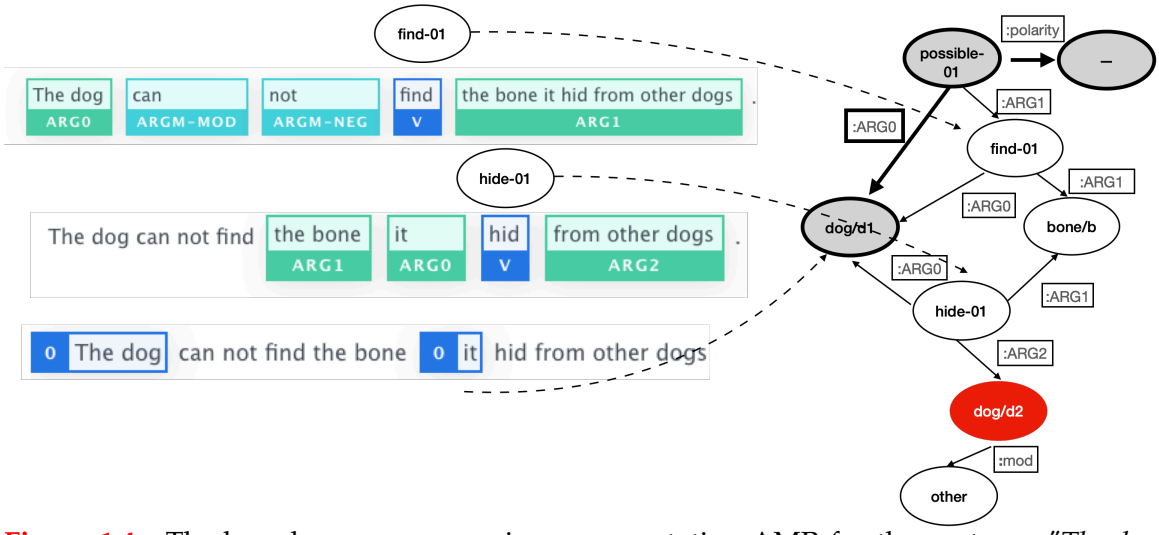

Figure 1.4: The broad coverage meaning representation AMR for the sentence *"The dog cannot find the bone it hid from the other dogs."*. It represent multiple phenomena in a single structure, inclduing the predicate-argument structure and word sense disambiguiation in semantic role labelling, coreference resolution, and so on.

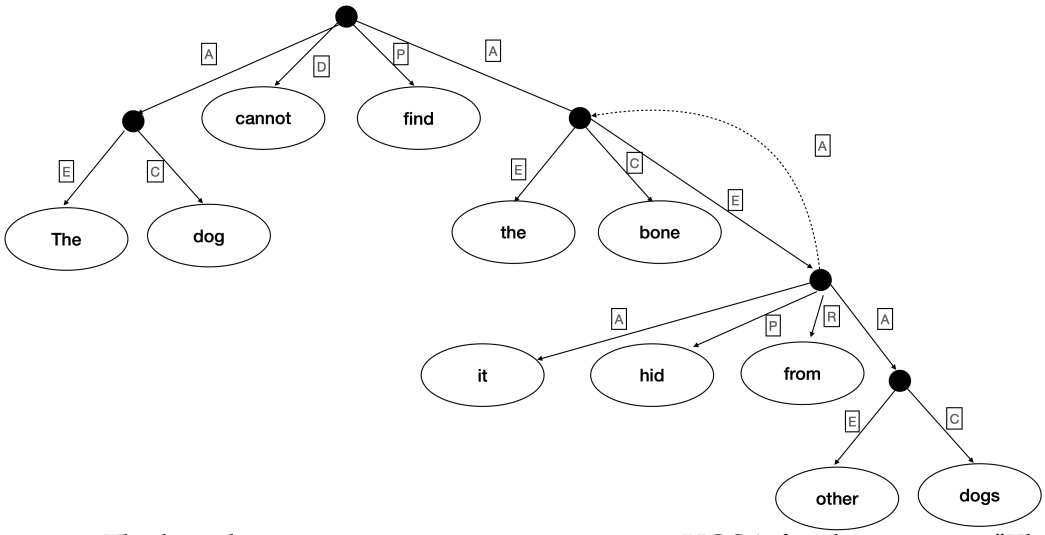

Figure 1.5: The broad coverage meaning representation UCCA for the sentence *"The dog cannot find the bone it hid from the other dogs."*

each with a set of possible values. Figure 1.6 shows an example of dialog state tracking, where each table filled with intent, slot and slot values, representing a dialog state for a user turn.

Lexical, syntactic structures, broad coverage semantic representations and application-specific representations, are interpretable to both human and computers. Such structured representations can enable rigorous document analysis, easier knowledge organization,

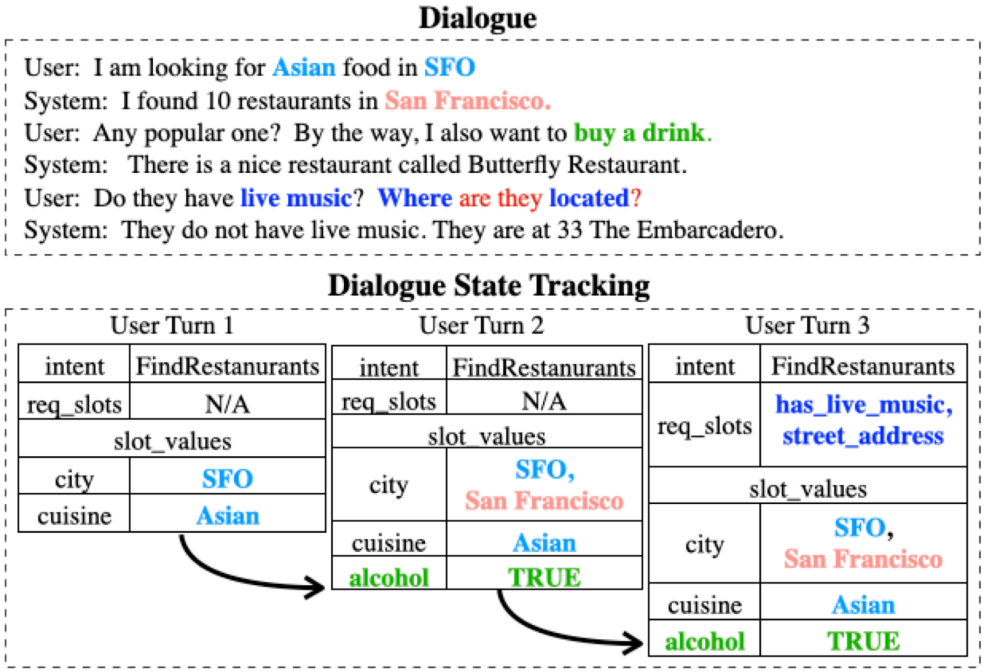

Figure 1.6: Example for dialog state tracking.

and programmable reasoning. Further more, they can be potentially helpful to offer actionable suggestions to guide human behavior, such as improving mental health counseling (Tanana et al., 2016), dialog state tracking (Budzianowski et al., 2018), scientific document analysis (Dernoncourt and Lee, 2017), and so on.

With the stunning rise of deep learning, modern NLP systems have achieved outstanding performance on many benchmark tasks, and offer helpful services, such as machine translation. Without any prior knowledge of the syntax or semantic structures for feature engineering, they simply feed the large amount of labeled raw data into an end-to-end deep learning model, and outperform many previous pipeline models built from handcrafted features. Recently, pretrained large language models even became the unified base model for many of the NLP tasks, which further boosts the performances.

However, recent research have shown that such end-to-end NLP systems often fail catastrophically when given unseen inputs from different sources or via adversarial attacks. The end-to-end black-box models lack inteperability and robustness, and they are fragile to maintain when deployed to real users. Using those large language models without any careful intervention will lead to fairness issues (Bommasani et al., 2021). Using interpretable symbolic representation in deep learning models can improve both

the efficiency and robustness of NLP systems. For example, combining the power of neural representation with symbolic AMR representation has shown great benefits to NLP applications like machine translation (Song et al., 2019), summarization (Liu et al., 2015), question answering (Kapanipathi et al., 2021) and so on.

Predicting structured representations of text is important for natural language processing, even in the deep learning era. In this thesis, we ground the studies of natural language structured prediction on both broad-coverage meaning representations and application-specific representations. Beyond pure data-driven methods, we primarily focus on studing deep linguistic structured prediction via independent factorization. We propose two kinds of generic inductive biases to support the independent factorization for each task, including Structural Inductive Biases and Natural Language as Inductive Biases.

1.1 Motivation

In this section, we first study the need for inductive biases in machine learning. Then we analyze where current deep learning models can get inductive biases from, and finally we highlight some problems that this thesis addresses about inductive biases for deep linguistic structured prediction.

1.1.1 Generalization: The Need for Inductive Bias

Any system (natural or artificial) that makes general inferences on the basis of particular and limited data must constrain its hypotheses in some way. With limited observations and resources (time, memory, energy), our human intelligence of generalizing to new environments makes us efficiently learn when interacting with the world and other human beings. This efficiency largely depends on many **inductive biases** from human intelligence (Gershman, 2021), which can potentially be helpful for machine intelligence. According to extensive cognitive science studies (Spelke, 1990; Bienenstock et al., 1996; Rehder, 2003; Harlow, 1949; Lake et al., 2016; Gershman, 2021), there are many inductive biases for human intelligence, such as compositionality, causality, learning to learn, and so on. We do not imply machine intelligence should mimic the human intellgence. Instead, we argue that those key human indutive biases help overcome limited observations and resources may inspire us to design the machine intelligence.

On the machine intelligence side, the no-free-lunch theorem for machine learning (Baxter, 2000; Wolpert et al., 1995) tells us that inductive biases that influence hypothesis selection is necessary to obtain generalization. Mitchell (1980) argues that inductive biases constitue the heart of generalization and indeed a key basis for learning itself.

Let us examine a concrete example: the popular supervised learning setting. We design algorithms that can learn from a set of supervised training examples to predict a certain target output for an input. The learning algorithm is presented some training examples that demonstrate the intended relation between the input and output values. Then the learner is supposed to learn a target function which captures the coorelations between the inputs and outputs. Furthermore, we hope that the learned target function can approximate the correct output, even for examples that have not been shown during training. We call the ability of generalizing to unseen data as generalization. Without any additional assumptions, this generalization problem cannot be solved since unseen situations might have an arbitrary output value. The kind of necessary assumptions are subsumed in the phrase inductive bias.

In this thesis, following the definition of ***bias*** in Mitchell (1980), we define ***indutive bias*** as:

> 'Any bias for choosing one generalization over another, other than strict consistency with the observed training instances'.

1.1.2 Beyond Training Data: The Origins of Inductive Biases

Inductive biases are widely studied in the history of machine learning. As the definition stated above, inductive biases can be any assumption beyond the observed training data. In this thesis, we focused on the supervised learning setting, where observed training data only means the annotated training data directly avaible to that task.

For the popular supervised learning setting, let $\mathcal{H}$ refer to machine learning model families, including deep learning models. The task of finding a target hypothesis h is reduced to estimating the model parameters by fitting the training data. Hence, preferences beyond training data can naturally organized into two goals: how to choose the hypothesis class $\mathcal{H}$ and how to find the h is necessary to generalize to new data. For example, different

model families can represent different hypothesis classes. For example, generalized linear models such as logistic regression and support vector machines, can only support linear decision boundaries. Secondly, inductive biases are also used in ***feature engineering***. For data that are not linear seperable, the choices of ***kernel designing*** also introduce inductive bias in kernel-based SVM models. For finding the specific hypothesis h, there are also many assumptions about optimization. For example, smoothness assumptions in the ***optimization*** method, such as Stochostic gradient decent, which was shown to have better generalization. ***Inference Algorithms***, such as combinatorial optimization approaches, such as graph cuts, partitions, bipartie mactching, and dynamic programming can also be involved during the hypothesis learning, which will also constrain the learning. In this dissertation, we mainly focus on representation learning. For inference, we use methods, such as greedy search, maximum spanning connected graph, and dynamic programming for CKY parsing. Finally, the biases in the ***Training Data*** will also influence the hypothesis finding. It is often the case that available datasets do not exactly represent the data distribution of interest. One particularly problematic case is when the dataset is biased in some way against a particular demographic group, which often leads to model predictions that unfairly disadvantage members of that group. Hence, ***Data manipulation*** can also help finding the desired hypoethsis by augmenting the original training data with inductive biases.

According to the universal approximation theorem (Hornik et al., 1989), properly parameterized neural network can represent any function. Further more, training data seems rich enough in many cases. It seems purely data-driven deep learning can learn any target function. Then, what kind of inductive biases we need in deep learning era? In the following, by two examples of inductive biases used in computer vision and natural language processing, we show what are the inductive biases used beyond the training data in the deep learning era.

For example, the translation invariance for convolutional neural networks (CNN) and pooling, and the recurrent assumption of recurrent neural networks (RNN), the equivariance over permutation for neural graph networks (GNN), the positional encoding for word orders in Transformer, they are all specific inductive biases.

In this thesis, we mainly study inductive biases from neural architectures and the

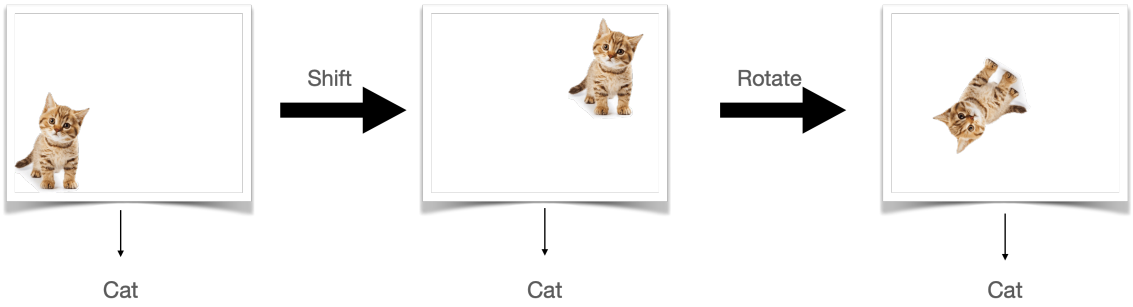

Figure 1.7: In object classication task, we hope the learned model can still recognize 'cat' for the unseen image with shifted or rotated cat.

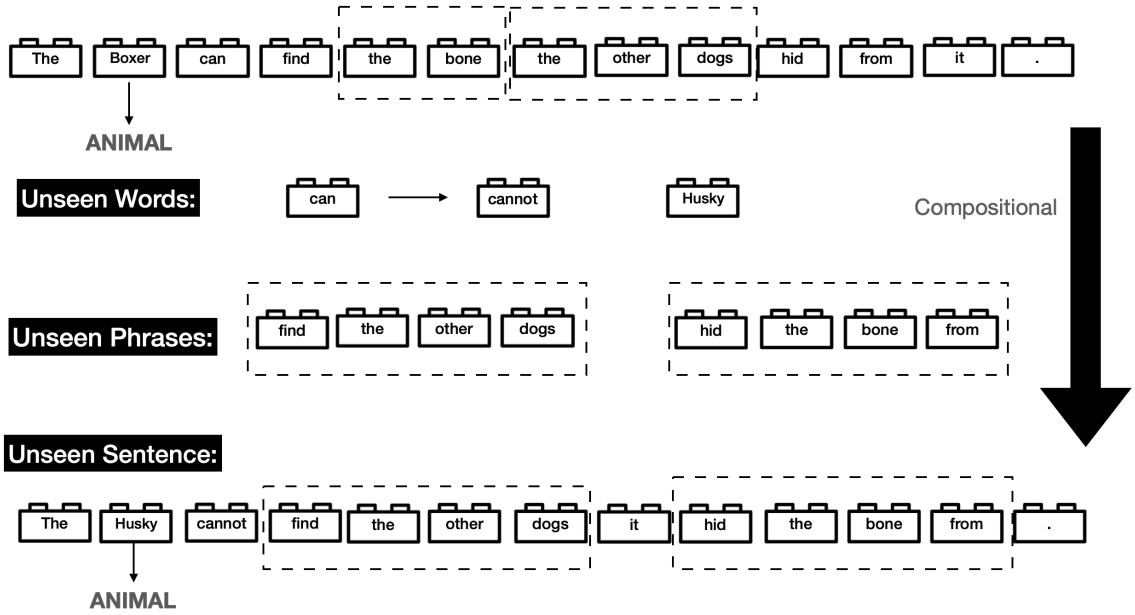

Figure 1.8: In named entity recognization task, we hope the learned model can still recognize 'dog' for new word 'Husky' in unseen context with newly composed words, phases and sentences.

corresponding representation learning.

1.1.3 Inductive Biases for Deep Lingusitic Structured Prediction

For systematic generalization, the search for appropriate inductive biases is necessary for deep linguistic structured prediction.

Let us denote an observation $x \in \mathcal{X}$. It can be any natural language text. Such as, the sentence "The dog cannot find the bone it hid from the other dogs" or a dialogue segment as shown in Figure 1.6. We define an output structured prediction for x by $y \in \mathcal{Y}(x)$. Here y is a structured symbolic representation for x. For exameple, y could be a sequence of part-of-speech tags in Figure 1.1, a constituent tree in Figure 1.2 or a dependency tree

in Figure 1.3. It can also be a broad-coverage meaning representation, like AMR, UCCA, or a dialogue state table. To represent the target function $y = f(x)$, we adopt the popular energy-minimization strategy by defining $f(x)$ as the minimizer of an auxiliary energy optimization problem.

$$f(x) = \arg\min_{y \in \mathcal{Y}(x)} E(x, y), \tag{1.1}$$

where $E(x, y)$ is a scoring function to represent the energy between x and a candidate output structure y.

In many NLP applications, the candidate output set $\mathcal{Y}(x)$ is finite but exponentially large, and its size may depend on the input x. For both exact and approximate optimization in Equation 1.1, the main challenges lie on how to model the representation of x and y, and the interactions between them. Practitioners typically employ energy functions with specific factorization structures to design efficient algorithms, by assuming the whole energy $E(x, y)$ can be decomposed as a sum of **factors** c, denoted by $E(x, y) = \sum_{c \in C} E(x, y_c)$.

A popular choice to represent the factorization is to index both x and y as a set of sub-components $x = (x_1, ..., x_i, ...x_N)$ and $y = (y_1, ...y_j, ...y_M)$. In AMR parsing as shown in Figure 1.4, x_i can be a word or multi-word expression in a sentence, while y_j can be a single AMR node and relation. For dialog state tracking in Figure 1.6, x_i is an utterance in the dialog, while y_i is the value for each intent and slot in the predicted frames. A factor c may depends on multiple subcomponents of x and y.

The interdependence assumptions between those sub-components in x and y are key in structured prediction model. In the Chapter 2, we show various representation formalism (such as graphical models), structured learning (max-margin framework) and inference approaches (dynamic programming, integer linear programming) to model the interdependence. Inductive biases can be designed in all the three key aspects.

In this thesis, we always assume the ***indenpendence factorization*** paradigm, where each factor c only depends on a well-segemented subset of subcomponents y_c and the aligned x components (anchors) $x_c = a(y_c)$. In other words, the output parts, once decomposed into mutually exclusive output segments, we consider each segment as an atomic output part, and each atomic part are independent from each other.

$$E(x, y) = \sum_{c \in C} E(x, y_c) = \sum_{c \in C} E(x, a(y_c), y_c) \tag{1.2}$$

Here, $a(y_c)$ is the alignment model to find how independent output parts y_c are ***anchored*** to the constituents of the observation x. Thus the prediction of each y_c are independent from each other, and can be locally decided by its aligned anchors.

Hence, this simple independent factorization can decompose the structured learning into decomposed local learning (constrained by some global constraints). More importantly, it also makes the inference tractable, thus can be easily employed in the end-to-end neural network training framework.

Using AMR parsing in Figure 1.4 as an example, the independent factorization will first segment the output y into small parts $y_c \in seg_{out}(y)$, then find the anchors x_c in the input sentence for each y_c from the candidate decomposition set $seg_{in}(x)$. For example, one of the segmented y_c in Figure 1.4 is a pre-categorized sub-graph '(possible-01 :polarity -)', and its anchor $a(y_c)$ is the anchor word 'cannot'. The words 'the', 'from' are mapped to empty nodes.

During inference, we can directly produce y_c for all candidates segments in a new sentence as shown in Figure 1.9. We first prepare a list of candidate anchors $seg_{in}(x) = \{$'The', 'dog', 'found', 'the', 'bone', 'it', 'hide'$\}$, then the model will produce the independent prediction y_c of each anchor as $\{\phi, $'dog', 'find-01', $\phi, $'bone', 'it', '*hide*'$\}$, then we assemble the non-empty y_c by predicing the relations between each other and finally forms y via postprocessing[3].

Hence, in the independence factorization setting, the problem is reduced into three challenges:

- How to decompose the output y into a set of independent parts y_c.
- How to decompose x and derive the aligned input $x_{a_{y_c}}$ at the index a_{y_c}.
- How to model the representation of x at the index a_{y_c} and y_c to compute the energy score $E(x, a(y_c), y_c)$.

[3]The post-processing include mergeing coreference nodes (as the 'dog' and 'it'), adding other attributes

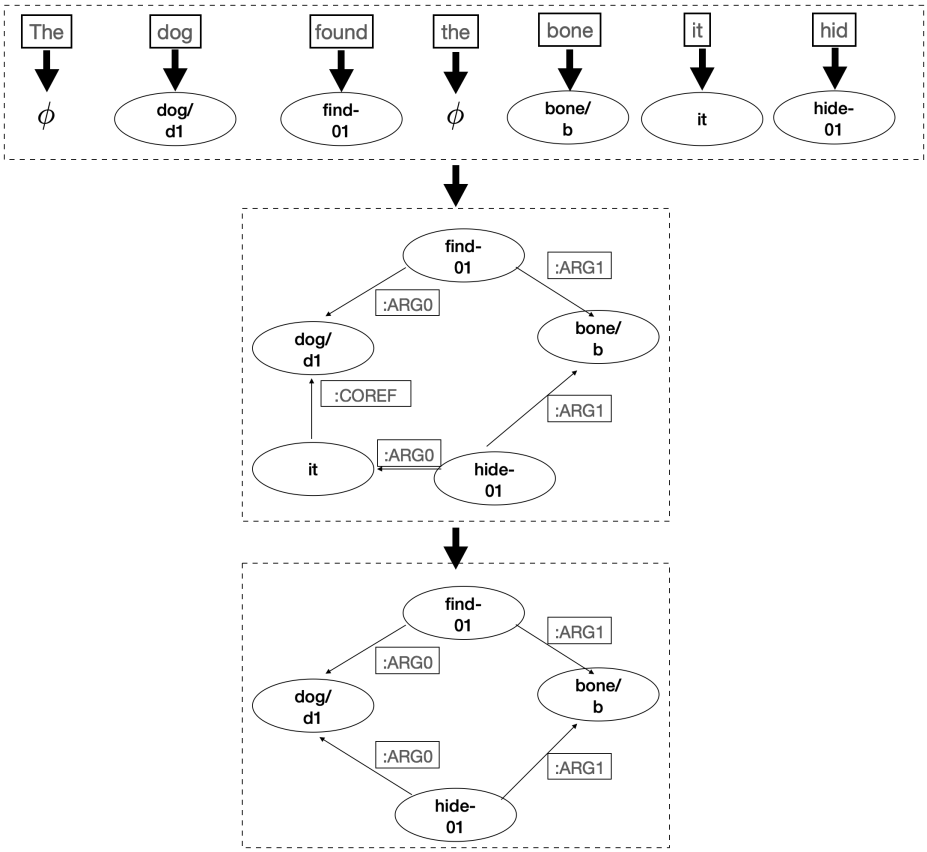

Figure 1.9: Independence Factorization for parsing a new sentence "The dog found the bone it hid" into an AMR graph

The first question on independently decomposing y is either straightforward or has been resolved by previously existing methods in our studied tasks. We mainly focus on the remaining challenges on modeling alignment and representation learning.

1.2 Contributions

Thesis Statement. Our claim is that by designing ***Structural Inductive Bias*** and ***Natural Language as Inductive Biases***, models with naive independent factorization can achieve strong performance at predicting the natural language structures across multiple broad-coverage meaning representations and application-specific representations.

Contributions. In this thesis, focusing on the independent factorization setting, we show that our proposed indutive biases can offer discriminative features to achieve competetive and generalizable performance on broad-coverage meaning representations and application-specific representations.

We next summarize the main contributions of this thesis, addressing some of the open problems mentioned in the previous section.

1. We proposed a unified parsing framework to support both **explicit lexical-anchoring** (including DELPH-IN MRS Bi-lexical Dependencies (DM, Ivanova et al., 2012a) and Prague Semantic Dependencies (PSD, Hajic et al., 2012; Miyao et al., 2014)), and **implicit lexical anchoring** (AMR). Over 16 teams in the shared tasks, my parser (Cao et al., 2019b) ***ranked 1st on AMR, 6th in DM, and 7th in PSD***. By combining Perturb-and-MAP sampling (Papandreou and Yuille, 2011) with differentiable Gumbel-Softmax Sinkhorn Networks (Mena et al., 2018), we can approximately infer the discrete latent-alignment variable in lexical-anchoring of the independent factorization setting. The **phrasal-anchoring** Universal Conceptual Cognitive Annotation (UCCA, Abend and Rappoport, 2013b) and Task-oriented Dialog Parsing (TOP, Gupta et al., 2018) are similar to constituency tree structure, except for unseen phenomena such as remote edges and discontinuous spans, we extend the existing algorithmic inductive bias for tree structure prediction and Cost-augmented CKY inferece to the new UCCA and TOP parsing tasks. Powered by strong span-representation learning method, my system (Cao et al., 2019b) ***ranked 5/16 on UCCA parsing***, and it can be reused for TOP parsing after a few preprocessing steps, and outperform serveral baseline models.

2. We address the problem of providing real-time guidance to therapists with a dialogue observer. It decompose the dialog structure analysis with two independent prediction tasks: (1) categorizing therapist and client MI behavioral codes and, (2) forcasting codes for upcoming utterances to help guide the conversation and potentially alert the therapist. For both tasks, I studied a hierarchical gated recurrent unit (HGRU) with the ***word-level attention*** and ***sentence-level attention*** to distinguish different importance of words and sentences (Cao et al., 2019a). Our experiments demonstrate that our models can outperform several baselines for both tasks. We also report the results of a careful analysis that reveals the impact of the various network design tradeoffs for modeling therapy dialogue.

3. Natural language can provide as inductive biases to describe the functions of the

intent/slot labels in task-oriented dialogue. We are among the first to use large pretrained language models for description-based dialog state tracking. We offer detailed comparative studies on how to transfer inductive biases to new domains and APIs with overlapping functions and task structures, including encoding strategies, supplementary pretraining, homogenuous and heterogeneous evalutions.

1.3 Thesis Outline

In the thesis, we discuss prior work related to an application in its own chapter, instead of putting them all in a single chapter. Therefore, we include a section of related work at the end of each application chapter. This thesis is divided into five parts, which we described below.

Chapter 2. Background. The first part systematizes the background of this thesis study in two sections:

- ***Structures in NLP.*** We first provide the necessary background about structures in natural languages for better highlighting our contributions in remaining chapters.

- ***Structured Prediction, Learning and Inference.*** We summarize the recent advances in deep structured prediction with respect to representational formaliam, learning and inference respectively. We overview the development of representation learning methods for natural language, from feature selection to deep learning based representation learning methods.

Chapter 3. Structural Inductive Biases for Lexical and Phrasal Anchoring. In this chapter, we introduce lexical and phrasal anchoring analysis to decompose the output structures into locally independent parts, where each part can be derived from its anchoring words or phrases in the input sentence. For lexical-anchoring, we propose a unified model to support both explicit and implicit alignment information between each input and output. For phrasal-anchoring, we compared different ways to learn the contextualized representation for the spans, and how they can bring discriminative features to our locally-dependent model. We show that with the above lexical and phrasal-anchoring based structural inductive biases for energey factorization and contextualized representation learning, our model

can learn efficient discriminative features for the anchor and achieve high performance in the locally-independent model.

Chapter 4. Structural Inductive Biases for Sentence and Dialog. In the following two chapters, we extend our study to structures beyond a single sentence. In this chapter, we study the sequential dialog flow structure in a style of therapy called Motivational Interviewing (MI, Miller and Rollnick, 2003, 2012), which is widely used for treating addiction-related problems. Sentence-level tags called Motivational Interview Skill Codes are designed to represent the intention of each utterance and the dialogue flow of the whole therapy session. By developing a modular family of neural networks categorizing and forcasting the dialog flow in the form of MISC codes, we show that the above mechanisms on dialogue representation can efficiently model the sequential structure of dialogue flow.

Chapter 5. Natural Language as Inductive Biases. In this chapter, we study using natural language descriptions to represent the meaning of output symbols (Intents and Slots) in task-oriented dialog state tracking, which helps to reduce the poor scalability to transfer to unseen domain and services. We study three main challenges of using natural language for label representation: schema encoding, supplementary training, and description styles.

Chapter 6. Conclusion and Future Work. This chapter concludes, by providing a summary of contributions and a discussion of possible directions of future work.

CHAPTER 2

BACKGROUND

In the following two sections, we will briefly review the background on symbolic representations for natural language including broad-coverage meaning representations and application-specific representations. Then, we will give an overview of the linguistic structure prediction and recent advances on deep structured prediction.

2.1 Symbolic Representations for Natural Language

In the Chapter 1, we have listed serveral lexical, syntactic and semantic structures in NLP. In this section, we will mainly introduce more detailed background on the broad-coverage semantic presentations §2.1.1 and application-specific representations on dialogue §2.1.2.

Before we introduce each semantic representation, we first define the term ***anchoring*** and ***anchors***, and then we use them to organize semantic representations in our dissertation.

Anchoring. As the same definition of ***anchoring*** used in the Meaning Representation Parsing (MRP) shared task (Oepen et al., 2019), we distinguish different flavors of semantic graphs based on the nature of the relationship they assume between the linguistic surface signal (typically a written sentence, i.e. a string) and the nodes of the graph. We refer to this relation as ***anchoring*** (of nodes onto sub-strings); other commonly used terms include alignment, correspondence, or lexicalization.

Anchor. Besides that, we also define the term ***anchor*** as the surface substring in the sentence, which is ***anchoring*** to the corresponding node in the graph. According to the type of the substring (lexicon, phrase, sentence and so on), we mainly classify the type of the ***anchors*** into ***lexical-anchoring***, ***phrasal-anchoring*** and ***sentential-anchoring***.

2.1.1 Broad-coverage Semantic Representation

For linguistic analysis, structures have been studied from subword-level morphology (Beesley and Karttunen, 2003), word-level lexicon semantics (Miller, 1998), to single sentence syntax/semantic representations (Baker et al., 1998; Palmer et al., 2005; Collins, 2003), and multi-sentences discourse analysis (Carlson et al., 2003; Wolf and Gibson, 2005; Prasad et al., 2008). This thesis covers broad-coverage graph-based semantic representations in a single sentence, which involves ***lexical-anchoring*** and ***phrasal-anchoring***. For lexical anchoring, we cover the DELPH-IN MRS Bi-lexical Dependencies (DM, Ivanova et al., 2012a) and Prague Semantic Dependencies (PSD, Hajic et al., 2012; Miyao et al., 2014), Abstract Meaning Representation (AMR, Banarescu et al., 2013a); While for phrasal-anchoring, we study Universal Conceptual Cognitive Annotation (UCCA, Abend and Rappoport, 2013a). The following lists a famous sentence (#20001001 in MRP Corpus), which is also the first sentence from WallStreetJournal (WSJ) Corpus from the Penn Treebank (Marcus et al., 1993).

`Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov.29.`

This sentence contains some interesting linguistic phenomes, such as morphlogy words, person and date named entities. Taking it as an example, we will introduce the detailed properties for each of the representations. [1]

2.1.1.1 Bi-lexical Semantic Dependencies

As shown in Figure 1.3, syntactic dependency structures captures the directed bi-lexical grammatical relations between words. Each labeled arc represents a directed relation from heads to dependents. However, different with the syntactic dependency tree, semantic dependency graphs aims to represent the semantic dependencies in the full sentence, inclduing word sense distinctions, the reentrances for correference, predicate-argument structures in semantic role labelling, namded entities representations and so on. Because of the reentrances, semantic representations can be a graph rather than a tree-like structures.

DM: DELPH-IN MRS-Derived Bi-Lexical Dependencies. It originates in a manual re-

[1] The summeriziation paper in MRP workshop (Marcus et al., 1993) introduces another example-based comparative studies for different meaning representations.

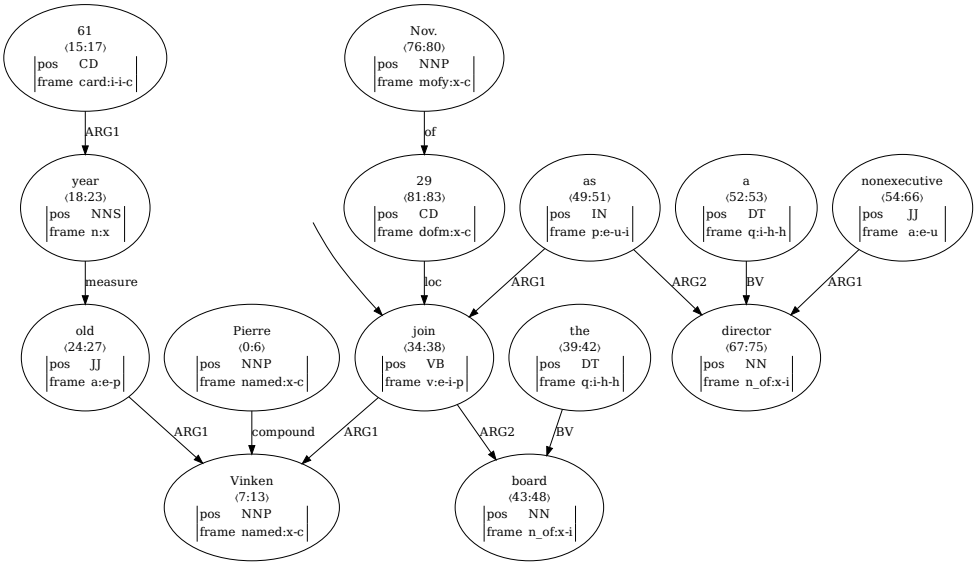

Figure 2.1: The DM representation for the sentence #20001001

annotation, dubbed DeepBank (Flickinger et al., 2012), with syntactico-semantic analyses of the LinGO English Resource Grammar (Oepen et al., 2004) in logical forms. These logical forms are often referred to as English Resrouce Semantics (ERS, Bender et al., 2015), and the underlying grammar is rooted in the general linguistic theory of Head-Driven Phrase Structure Grammar (HPSG, Pollard and Sag, 1994).

Then Ivanova et al. (2012b) propose a two-stage version to transform the ERS logical forms into bi-lexical semantic dependency graphs. As shown in the Figure 2.3, ERS logical forms are firstly tranformed into Elementary Dependency Structures (EDS, Oepen and Lønning, 2006), then EDS are simplified into pure bi-lexical dependencies, dubbed DELPH-IN MRS Bi-Lexical Dependencies (or DM). As shown in Figure 2.1, graph nodes in DM are anchoring to the surface tokens. However, the underlying sentence is fully covered in the graph. For example, the word “will" does not produce any node in the DM graph. Edges mainly indicate semantic argument roles (ARG1, ARG2, ...) into the relation corresponding to their source node [2], but there are some more specialized edge labels too. For example, it uses ***compound*** to reflect the name “Pierre Vinken" as a whole, and uses BV

[2]The annotation of predicate-arugment structures is based on the semantic inferface (SEM-I) in the ERG. Please refer to this introduction for more details.`https://github.com/delph-in/docs/wiki/ErgSemantics_Interface`

(bound variable, e.g., the word 'a') as a reflection of quantification in the underlying logic quantification.

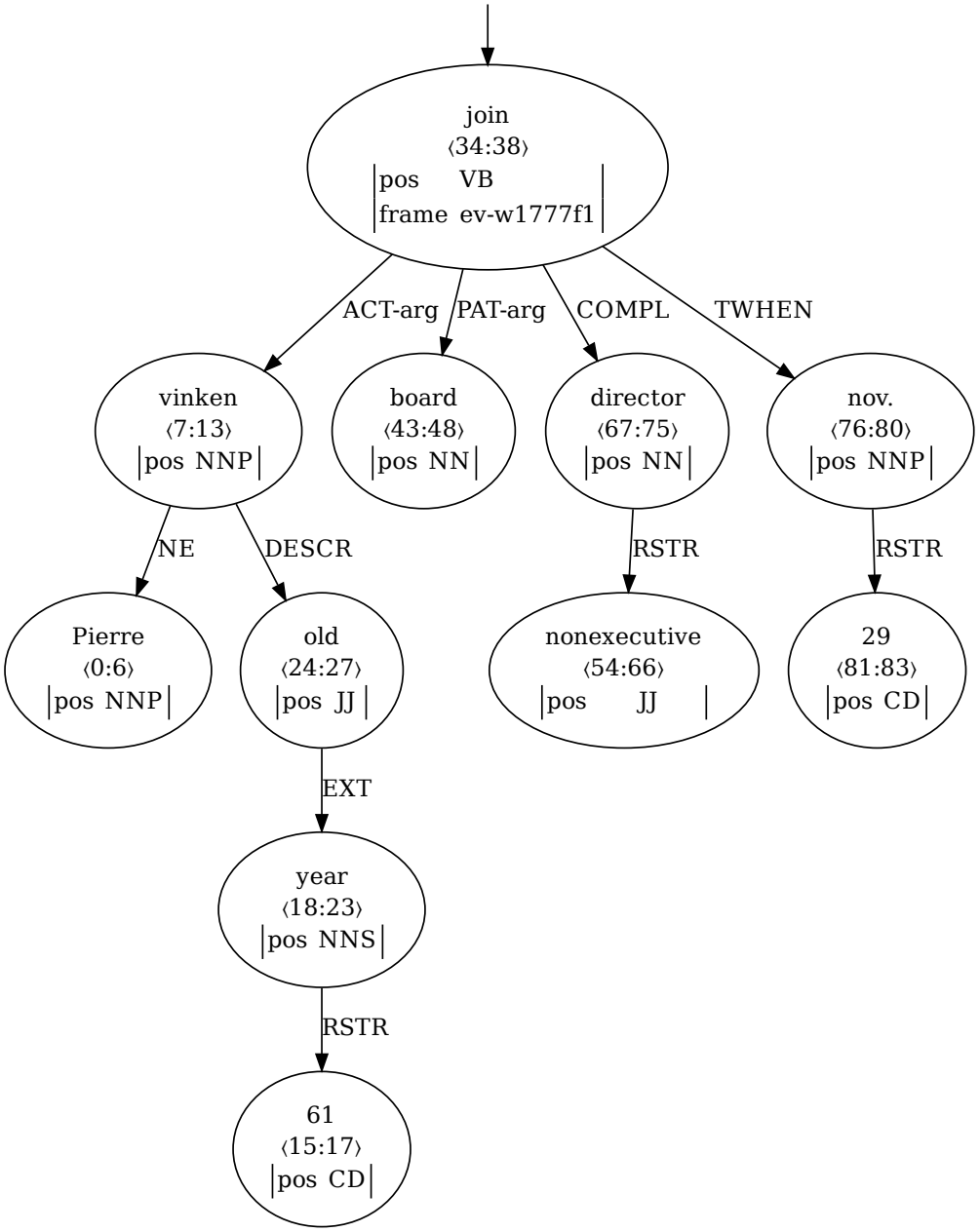

Figure 2.2: The PSD representation for the sentence #20001001

Prague Semantic Dependencies. Besides DM, there is another line of research to simplify the richer syntacticosemantic representations into bi-lexical semantic dependencies. It adopts the reduction of tectogrammatical trees (or t-trees) from the linguistic school of Functional Generative Description (FGD, Sgall et al., 1986; Hajic et al., 2012) into PSD. The Prague Czech-English Dependency Treebank (PCEDT, Hajic et al., 2012) is a set of parallel dependency trees over the WSJ texts from the PTB, and their Czech translations. The PSD bi-lexical dependencies are extracted from the tectogrammatical annotation layer. Top nodes are derived from t-tree roots; Especially, they mostly correspond to main verbs. In case of coordinate clauses, there are multiple top nodes per sentence. Figure 2.2 shows the PSD represention of our example sentence. One major difference are the role labels and verb frames, they are grounded in a machine-readable valency lexicon (Urešová et al., 2016), and the frame values on verbal nodes indicate specific verbal senses in the lexicon.

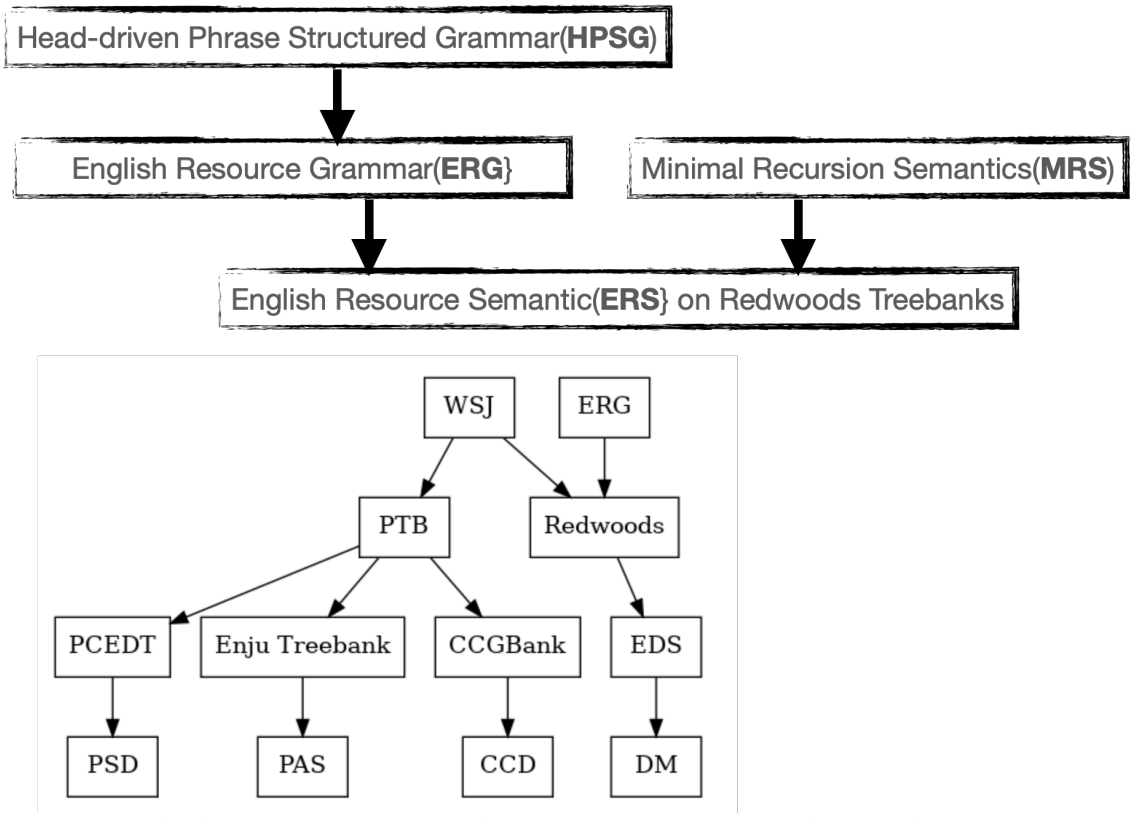

Figure 2.3: The hierarchical relations between DM, PSD, and their underlying grammar and semantics

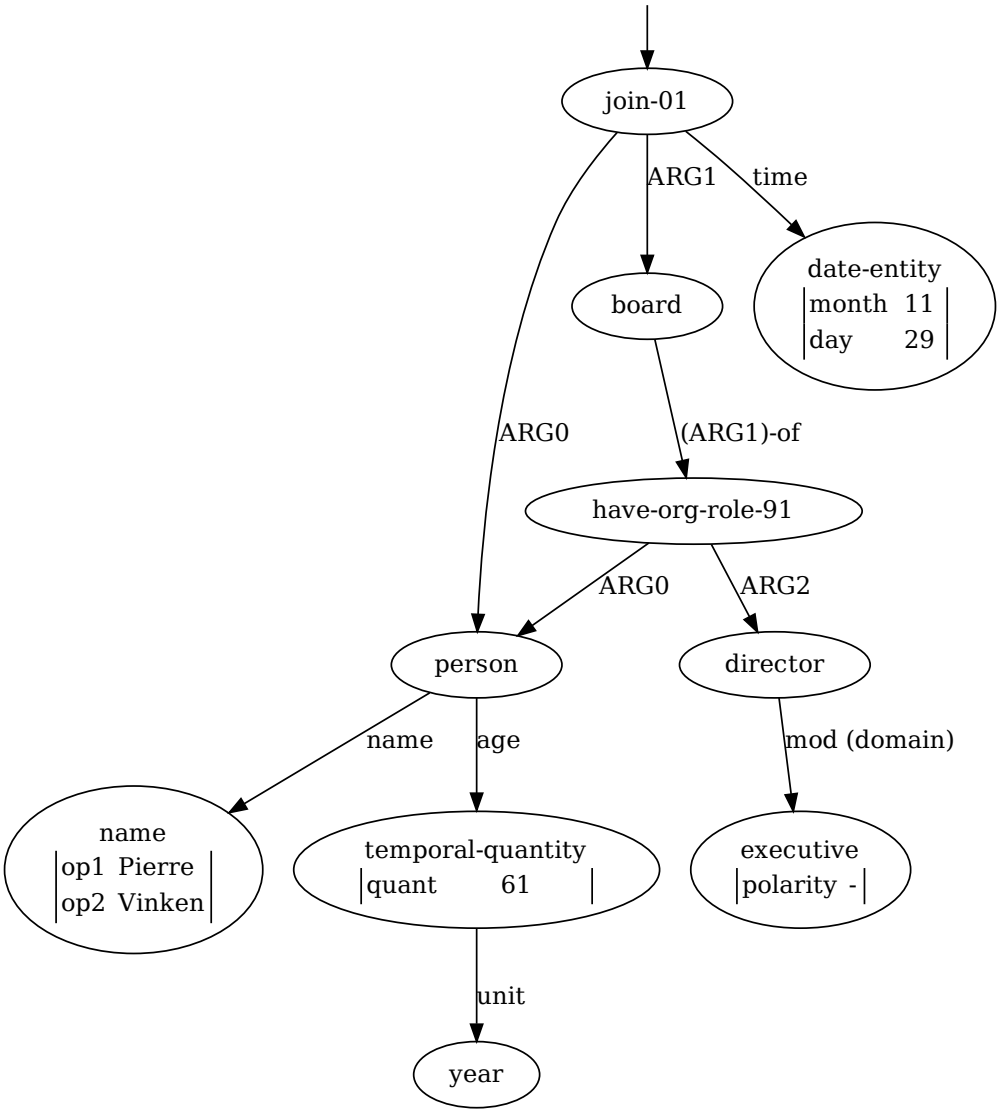

Figure 2.4: The AMR representation for the sentence #20001001

2.1.1.2 Abstract Meaning Representation

As shown in Figure 2.4, Abstract Meaning Representation represents sentence meaning as directed graphs with labeled nodes (concepts) and edges (relations). AMRs are rooted, labeled graphs that are easy for people to read, and easy for programs to traverse. AMR concepts are either English words (“board"), PropBank framesets (“join-01"), or special entity keywords (“date-entity", “person", “name", etc.), quantities (“temporal-quantity",

“distance-quantity", etc.), and logical conjunctions (“and", etc). AMR strives for a more logical, less syntactic representation. For example, it represent the word “nonexecutive" with a negation “(:polarity 0)" with the concept “executive". Further more, different from DM and PSD on predicate-argument representation, AMR makes extensive use of PropBank framesets (Kingsbury and Palmer, 2002; Palmer et al., 2005). For example, It represents the verb “join" using the frame âĂIJjoin-01âĂİ. At the mean time, AMR also newly designs special frames to reuse those core roles in Propbank. As shown in Figure 2.4, the word “board" have a role “ARG1-of" to a special frame “have-org-role-91".

The above abstraction allows for concepts and relations not explicitly mentioned in the text, but leaves open the question of how these are derived from the text. This question is important because training statistical AMR parsers typically starts with a conjectured alignment between tokens and the graph elements. Most AMR parsers (*e.g.* Flanigan et al., 2014; Wang et al., 2015; Artzi et al., 2015; Pust et al., 2015; Peng et al., 2015; Konstas et al., 2017; Wang and Xue, 2017) use either the JAMR aligner Flanigan et al. (2014) or the ISI aligner Pourdamghani et al. (2014) for this purpose[3] We introduced more details about AMR alignment in Section 3.1.1.2.

2.1.1.3 Universal Conceptual Cognitive Anotation

Similar with AMR, UCCA is designed to abstract the semantic scheme away from its surface from and syntactic forms. UCCA uses directed acyclic graphs (DAGs) to represent its semantic structures. The atomic meaning-bearing units are placed at the leaves of the DAG and are called terminals. The nodes of the graph are called units. A unit may be either (i) a terminal or (ii) several elements jointly viewed as a single entity according to some semantic or cognitive consideration. In many cases, a non-terminal unit is comprised of a single relation and the units it applies to (its arguments), although in some cases it may also contain secondary relations. Hierarchy is formed by using units as arguments or relations in other units.

While different from previous DM, PSD and AMR, categories are not annotated on nodes, but on edges and represent the descendant unit’s role in forming the semantics of

[3]Other aligners exist – *e.g.*, Chen and Palmer (2017) uses dependencies, but raw text alignments are more prevalent.

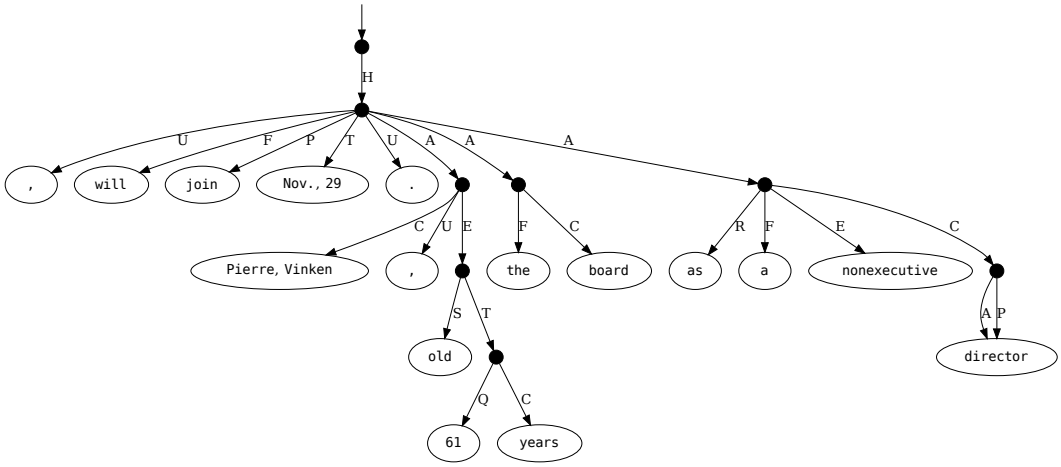

Figure 2.5: The UCCA representation for the sentence #20001001

the parent unit. The foundational layer is designed to cover the entire text, so that each word participates in at least one unit. It focuses on argument structures of verbal, nominal and adjectival predicates and the inter-relations between them. Argument structure phenomena are considered basic by many approaches to semantic and grammatical representation, and have a high applicative value, as demonstrated by their extensive use in NLP. The foundational layer views the text as a collection of Scenes. A Scene can describe some movement or action, or a temporally persistent state. As shown in Figure 2.5, for the event 'join the board', 'join' with an incoming edge 'P' denoting the category "Process", which is the main relation of a scene that evolves in time. While the edge 'A' linked to the non-terminal node corresponding to "Pierre, Vinken, 61 years old" means the unit "Pierre, Vinken" is the participant of the scene. For more detailed comparative studies over the state-of-art semantic representation, please refer to the paper Abend and Rappoport (2017).

2.1.2 Application-specific Representation on Dialogue

In this thesis, we ground the study on application-specific representation on dialogue. The following section will introduce the dialog rerepsentations via dialogue act, dialog state and richer conversational semantic representations.

2.1.2.1 Dialogue Act and MISC Codes

In utterance-level, dialogue acts are designed to represent the function of each utterance in the dialogue. The key insight behind dialgue act is that each utterance in a dialogue is a kind of action being performed by the speaker. The history of dialogue act can be derived back to the philosopher Wittgenstein (2010). A dialogue act has two main components: a communicative function and a semantic content. Bunt et al. (2010) provides an ISO project developing an international standard for annotating dialogue with semantic information, in particular concerning the communicative functions of the utterances, the kind of content they address, and the dependency relations to what was said and done earlier in the dialogue. Similarly, Motivational Interview Skill Codes (MISC, Miller and Rollnick, 2003, 2012) are also proposed to represent the functions of each client and therapist utterance in the psychotherapy dialogue. In this part, we will mainly introduce the MISC Codes for psychotherapy dialogue.

Code	Count	Description	Examples
		Client Behavioral Codes	
FN	47715	Follow/ Neutral: unrelated to changing or sustaining behavior.	"You know, I didn't smoke for a while." "I have smoked for forty years now."
CT	5099	Utterances about changing unhealthy behavior.	"I want to stop smoking."
ST	4378	Utterances about sustaining unhealthy behavior.	"I really don't think I smoke too much."
		Therapist Behavioral Codes	
FA	17468	Facilitate conversation	"Mm Hmm.", "OK.","Tell me more."
GI	15271	Give information or feedback.	"I'm Steve.", "Yes, alcohol is a depressant."
RES	6246	Simple reflection about the client's most recent utterance.	C: "I didn't smoke last week" T: "Cool, you avoided smoking last week."
REC	4651	Complex reflection based on a client's conversation history.	C: "I didn't smoke last week." T: "You mean things begin to change".
QUC	5218	Closed question	"Did you smoke this week?"
QUO	4509	Open question	"Tell me more about your week."
MIA	3869	MI adherent, *e.g.*, affirmation, advising with permission, etc.	"You've accomplished a difficult task." "Is it OK if I suggested something?"
MIN	1019	MI non-adherent, *e.g.*, confront, advising without permission, etc.	"You hurt the baby's health for cigarettes?" "You ask them not to drink at your house."

Table 2.1: Distribution, description and examples of MISC labels.

Motivational Interviewing (MI) is a style of psychotherapy that seeks to resolve a client's

ambivalence towards their problems, thereby motivating behavior change. Several meta-analyses and empirical studies have shown the high efficacy and success of MI in psychotherapy (Burke et al., 2004; Martins and McNeil, 2009; Lundahl et al., 2010). However, MI skills take practice to master and require ongoing coaching and feedback to sustain (Schwalbe et al., 2014). Given the emphasis on using specific types of linguistic behaviors in MI (*e.g.*, open questions and reflections), fine-grained behavioral coding plays an important role in MI theory and training.

Motivational Interviewing Skill Codes (MISC, table 4.1) is a framework for coding MI sessions. It facilitates evaluating therapy sessions via utterance-level labels that are akin to dialogue acts (Stolcke et al., 2000; Jurafsky and Martin, 2019), and are designed to examine therapist and client behavior in a therapy session.[4]

2.1.2.2 Dialog State Tracking

Figure 2.6: Different Flight Service Ontology for Dialogue State Tracking

From the simple GUS (Bobrow et al., 1977) to the modern task-based dialogue systems built in virtual assistants (Alexa, Siri, and Google Assistant *et al.*), they are all based around frames. Frame theory is derived from Fillmore's case theory (Fillmore, 1968). A frame is a kind of knowledge structure representing the kinds of intentions the system can extract from user sentences, and consists of a collection of slots, each of which can take a set of possible values. Together this set of frames is sometimes called a domain ontology.

[4]The original MISC description of Miller et al. (2003) included 28 labels (9 client, 19 therapist). Due to data scarcity and label confusion, various strategies are proposed to merge the labels into a coarser set. We adopt the grouping proposed by Xiao et al. (2016); the appendix Section A.1 gives more details.

As shown in Figure 2.6, two flight services present two different ontology for the same domain task of booking a flight. For the dialogue presented in the center of the figure, the two services will produce different dialogue state for each user dialogue turn, and request different commands for downsteam information retrieval components. Imaging another hotel booking tasks, then there will be new onotlogy for the new domain.

The intent classification task is to understand what the user trying to accomplish? Booking a flight, Finding a Movie, or booking a hotel. While the slot filling task to extract the particular slots and fillers that the user intends the system to understand from their utterance with respect to their intent. As shown in Figure 2.6, bolded text are the evidence for different slot values. For example, “economic" implies the value of the slot seat class, and “June 10" indicates the time of departure.

As the dialogue goes on, a dialogue state tracker maintains the current state of the dialogue (which include the user’s most recent dialogue intent, plus the entire set of slot-filler constraints the user has expressed so far). The dialogue policy decides what the system should do or say next. Finally, dialogue state systems have a natural language generation component to reply the utterance to users.

2.1.2.3 Conversational Semantic Representation

Most existing annotations for task oriented dialog systems have fallen on the extremes of non-recursive intent and slot tagging, such as in the MultiWOZ Budzianowski et al. (2018). Hence, the previous intent-slot dialog state representation have poor compositionality to represent complex conversational request, such multiple intents in the same utterance, and nested intent slot structures.

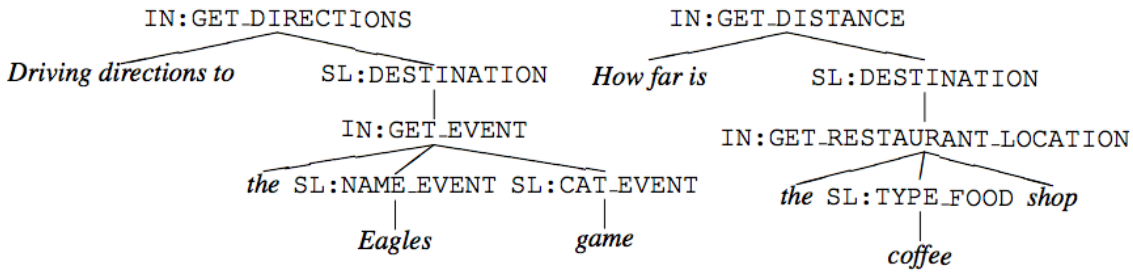

Figure 2.7: TOP examples for conversational semantic parsing (excerpted from the original paper (Gupta et al., 2018))

Recently, conversational parsing has attract much attention to represent the dialogue state in a more compositional way, such as the hierarchical tree structure in the Task-Oriented dialogue Parsing (TOP, Gupta et al., 2018; Aghajanyan et al., 2020), (TreeDST, Cheng et al., 2020), and (Dataflow, Andreas et al., 2020). In a summ, those conversation semantic representation offers richer intent slot compositions, and support complex conversational linguistic phenomenos, such as dialogue state revision and recovering.

In this thesis, we study the structures of single sentence representation TOP for dialogue representations. Figure 2.7 shows two examples of the nested structures of TOP structures. All intents and slots are non-teriminal nodes, and their labels are prefixed with ***IN:*** or ***SL:*** respectively.

The TOP tree structure shares a lot similaries with consituent tree shown in Figure 1.2. It has the following three structural constraints. (1) The top level node must be an intent. (2) An intent can have tokens and/or slots as children (3) A slot can have either tokens or intents as its children.

2.2 Deep Structured Prediction in NLP

Due to the power of representation learning, deep learning is widely used to extract sophisticated representations for the inputs in various NLP tasks. In this thesis, instead of focusing on a single task, we systematically study the representation learning challenges for multiple sets of tasks based on independent factorization assumption. In this section, We summerize the recent advances in deep structured prediction with respect to representational formalism §2.2.1, learning §2.2.2 and inference §2.2.3 respectively.

In this thesis, we study the specific inductive biases based on the principle of compositionality, which can be used for **different purposes more than the accuracy on a single task or a single domain**. For example, with simple independent factorization, we study a universal alignment-based model to support *cross-framework* meaning representation parsing. By defining two complementary dialogueue observers to sequentially predict the MISC code for both current and future utterance, our model emphasizes both *accurate and real-time* assistance to a therapist. We propose to represent the output labels with natural language descriptions for *zero-shot learning* on unseen labels §5. In our thesis, we show that during the rapid progress of representation learning methods, our proposed inductive

biases still can outperform the standard usage baselines.

2.2.1 Formulation of Structural Interdependence

Graphical Models. A graphical model is a probabilistic model for which a graph expresses the conditional dependence structure between random variables. Generally, probabilistic graphical models use a graph-based representation as the foundation for encoding a distribution over a multi-dimensional space, which represents a set of independences that hold in the specific distribution. Two branches of graphical representations of distributions are commonly used, namely, Bayesian networks and Markov random fields. Both families encompass the properties of factorization and independences, but they differ in the set of independences they can encode and the factorization of the distribution that they induce. In this thesis, we mainly use the undirected Markov random fileds to represent our independent factorization assumptions.

As shown in Figure 2.8, each circle represent a variable, while each rectangle shows a factor between the input sentence variable x and each decomposed segements of output structures y. The difference between left and right figure lies to the alignment variable a in the center. In the left figure, the shade cicle a means the alignment are explicitly observed after the decomposion. While in the right figure, the alignment variable a is not observed.

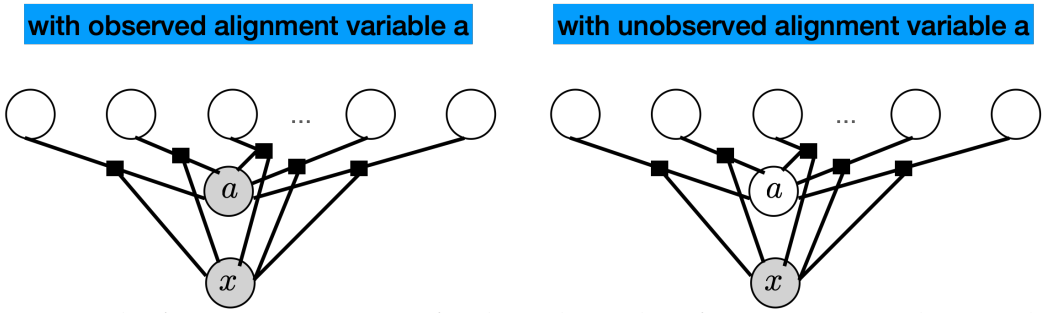

Figure 2.8: The factor representation for the independent factorization used in our thesis

Constrained Conditional Models. Besides using graphical models to declaritively represent the structural interdependence between variables, constrained conditional models (CCM, Chang et al., 2012) is another machine learning and inference framework for the same goals. More specifically, CCM emphasizes augmenting the learning of conditional models with declarative constraints. It aims to support constrained decisions in

an expressive output space while maintaining modularity and tractability of training and inference. These constraints can express either hard restrictions, completely prohibiting some assignments, or soft restrictions, penalizing unlikely assignments. One popular formalism to represent the constraints is to use an interger linear programming (ILP), which has been widely used to constrain learning in many NLP tasks (Roth and Yih, 2007). The declaritive linear objective functions and linear constraints, and the availibities of the off-the-shelf solvers make this formalism very easy to use.

Recently, to inject known hand-crafted constraints bewteen discrete variable assignments in the deep neural networks, one foundmental challange is how to represent the constraints in end-to-end differentiable ways Bach et al. (2017). For example, Li and Srikumar (2019) propose to use differentiable fuzzy logic operators to augment the neural networks with boolean logic. Pacheco and Goldwasser (2021) introduces a declarative Deep Relational Learning framework (DRAIL) via integrating neural representation learners with probabilistic logic.

Besides representing the constraints as logic forms, many recent work also studies representing constraints with discrete latent variable models, such as StructVAE for latent tree structured variables, Yin et al. (2018); Corro and Titov (2019). Our work on latent alignment models also falls into this category.

Besides injecting declaritive constraints, recent research also learn the constraints in an end-to-end ways. Belanger and McCallum (SPEN, 2016) define energy functions that can learn the arbitrary dependencies among parts of structured outputs by relaxing the whole structured outputs into continuous vectors, Following this, inference network Tu and Gimpel (2018) was proposed to learn the constrained network for inference, which approximate the cost-augmented inference during training and then fine-tuning for test-time inference.

2.2.2 Neural Representation Learning

Structured prediction requires the representation learning can capture both the distriminitive interactions between x and y and also allow efficient combinatorial optimization over y. Ideally, we hope neural repsentation learning can handle all of this.

The key challenge of trying to apply analytical and computational methods to text data

is how to represent the text in a way that is amenable to operations such as similarity, composition, etc. Besides the early day one-hot representation and TF-IDF extensions, word embedding and nerual contextualized representation are widely used in modern deep learning based models. In this section, we review the recent advances from static word embedding based methods to attention-based dynamic features selection and contextualized representation. Finally, we also introduced the rapid progress in language encoding architectures, from recurrent neural network to transformer, and the corresponding pretrained language models ELMo , BERT, GPT3 etc.

Static Word Embedding. Word embeddings are commonly Baroni et al. (2014); Pennington et al. (2014); Li et al. (2015b) categorized into two types, depending upon the strategies used to induce them: (1) Prediction-based models, via local data in sentence (a word' context). (2) Count-based models, via the global corpus-wide statistics (such as word counts, co-occurrece). Skip-gram with negative sampling (SGNS, Mikolov et al., 2013) and GloVe Pennington et al. (2014) are among the best known models for the two type respectively. However, they create a single fixed representation for each word, a notable problem with static word embeddings is that all senses of a polysemous word must share a single vector.

Contetualized Representation and Contextualing Models. To resolve the issue of static word embedding, sequence encoders (such as LSTM (Hochreiter and Schmidhuber, 1997), Transformer (Vaswani et al., 2017)) can be used as contextualizing models to encode the whole context and produce a ***contexualized representation*** for each word, phrase or the whole sentence. In this way, the contextualized representation dynamicly depends on the entire sentence. Further more, based on the neural sequence encodingarchitectures, pretraining language models with a large amount of text can create more powerful representations. ELMo Peters et al. (2018a) creates contextualized representations of each token by concatenating the internal states of a 2-layer biLSTM trained on a bidirectional language modelling task. In contrast, BERT (Devlin et al., 2019) and GPT-2 (Radford et al., 2018) are bi-directional and uni-directional transformer-based language models respectively. Peters et al. (2019) shows that ELMo contextualized representation is more suitable to be used a fixed word embedding. In our thesis, we simply replace the original fixed word embed-

ding with ELMo, and then feed them into other nerual models. While for BERT and GPT-2, finetuning them on downsteam task will lead better performance.

2.2.3 Inference

Learning with structured data typically involves searching or summing over a set with an exponential number of structured elements, for example the set of all parse trees for a given sentence. In the deep learning community, it is common to fit models by computing point estimates, such as the MLE or MAP estimate. Such MAP inference approaches seem particularly appealing, since they are computationally fairly cheap, and can use the prior to reduce overfitting. In this way, the nerual models only learn a single set of parameter. However, the point estimation does not capture the associated uncertainties (Murphy, 2022; Wilson and Izmailov, 2020). Hence, in structured prediction research, we care about both MAP inference and marginal inference.

MAP Inference. Various exact inference methods are proposed for MAP inference in NLP tasks. Exact inference methods include dynamic programming based methods (such as viterbi (Viterbi, 1967) for hidden markov models, CKY for context-tree grammars Kasami (1966); Younger (1967); Cocke (1969) , Max Spanning Arborescence for spanning tree Chu (1965); Edmonds et al. (1967), and so on), and Integer Linear Programming (Roth and Yih, 2005, 2007; Berant et al., 2014). On the other side, approximate inference methods includes various sampling methods (Finkel et al., 2005; Singh et al., 2012), search-based methods (Daumé et al., 2009; Ross et al., 2011; Chang et al., 2015), and Linear Programming Relaxations (Rush and Collins, 2012; Werner and PruËĞsa, 2014).

Marginal Inference. Integration is at the heart of marginal inference, whereas differentiation is at the heart of optimization. Corresponding to the each of the above exact MAP inference algorithms, various methods are proposed for marginal inference. They compute marginal probabilities and partition functions which are central to many methods, such as EM (Baker, 1979; Weizenbaum, 1966), constrative estimation (Smith and Eisner, 2005), Conditional Random Field (CRF, Lafferty et al., 2001), max-margin training over all candidate targets Koller (2004). For linguistic structured prediction, exact marginal inference methods include forward-backward algorithm for HMM (Binder et al., 1997), Inside-outside (Baker, 1979), Matrix-Tree Theorem for non-projective dependency struc-

tures (Koo et al., 2007; Liu and Lapata, 2018).

2.3 Chapter Summary

This section introduces the background on symbolic representation for natural language including broad-coverage meaning representations and application-specific representations. Then, we also give an overview of the linguistic structure prediction and recent advances on deep structured prediction. In this thesis, instead of studying structured prediction for each lingusitic structured prediction tasks separately, based on independent factorization, we firstly categorize those symbolic representations according to the anchoring types. Then in the following chapters, we study on the structural inductive biases and natural language as inductive biases for each of them.

CHAPTER 3

PARSING MEANING REREPRESENTATIONS VIA LEXICAL AND PHARASAL ANCHORING

The design and implementation of broad-coverage and linguistically motivated meaning representation frameworks for natural language is attracting growing attention in recent years. With the advent of deep neural network-based machine learning techniques, we have made significant progress to automatically parse sentences intro structured meaning representation (Oepen et al., 2014, 2015; May, 2016; Hershcovich et al., 2019). Moreover, the differences between various representation frameworks has a significant impact on the design and performance of the parsing systems.

Due to the abstract nature of semantics, there is a diverse set of meaning representation frameworks in the literature (Abend and Rappoport, 2017). In some application scenario, tasks-specific formal representations such as database queries and arithmetic formula have also been proposed. However, primarily the study in computational semantics focuses on frameworks that are theoretically grounded on formal semantic theories, and sometimes also with assumptions on underlying syntactic structures.

Anchoring is crucial in graph-based meaning representation parsing. Training a statistical parser typically starts with a conjectured alignment between tokens/spans and the semantic graph nodes to help to factorize the supervision of graph structure into nodes and edges. In this chapter, with evidence from previous research on AMR alignments (Pourdamghani et al., 2014; Flanigan et al., 2014; Wang and Xue, 2017; Chen and Palmer, 2017; Szubert et al., 2018; Lyu and Titov, 2018), we propose a uniform handling of three meaning representations from `Flavor-0` (DM, PSD) and `Flavor-2` (AMR) into a new group referred to as the **lexical-anchoring** MRs. It supports both explicit and implicit anchoring of semantic concepts to tokens. The other two meaning representations from `Flavor-1` (EDS, UCCA) is referred to the group of **phrasal-anchoring** MRs where the

semantic concepts are anchored to phrases as well.

To support the simplified taxonomy, we named our parser as LAPA (**L**exical-**A**nchoring and **P**hrasal-**A**nchoring)[1]. We proposed a graph-based parsing framework with a latent-alignment mechanism to support both explicit and implicit lexicon anchoring. According to official evaluation results, our submission for this group ranked 1st in the AMR subtask, 6th on PSD, and 7th on DM respectively, among 16 participating teams. For phrasal-anchoring, we proposed a CKY-based constituent tree parsing algorithm to resolve the anchor in UCCA, and our post-evaluation submission ranked 5th on UCCAsubtask.

3.1 Related Work and Anchoring Analysis

The 2019 Conference on Computational Language Learning (CoNLL) hosted a shared task on Cross-Framework Meaning Representation Parsing (MRP 2019, Oepen et al., 2019), which encourage participants in building a parser for five different meaning representations in three distinct flavors. `Flavor-0` includes the DELPH-IN MRS Bi-lexical Dependencies (DM, Ivanova et al., 2012a) and Prague Semantic Dependencies (PSD, Hajic et al., 2012; Miyao et al., 2014). Both frameworks under this representation have a syntactic backbone that is (either natively or by-proxy) based on bi-lexical dependency structures. As a result, the semantic concepts in these meaning representations can be anchored to the individual lexical units of the sentence. `Flavor-1` includes Elementary Dependency Structures (EDS, Oepen and Lønning, 2006) and Universal Conceptual Cognitive Annotation framework (UCCA, Abend and Rappoport, 2013b), which shows an explicit, many-to-many anchoring of semantic concepts onto sub-strings of the underlying sentence. Finally, `Flavor-2` includes Abstract Meaning Representation (AMR, Banarescu et al., 2013b), which is designed to abstract the meaning representation away from its surface token. But it leaves open the question of how these are derived.

3.1.1 Lexical-Anchoring Analysis on DM, PSD and AMR

Previous studies have shown that the nodes in AMR graphs are predominantly aligned with the surface lexical units, although explicit anchoring is absent from the AMR representation. In this section, we review the related work supporting the claim of the implicit

[1]The code is available online at `https://github.com/utahnlp/lapa-mrp`

anchoring in AMR is actually lexical-anchoring, which can be merged into `Flavor-0` when we consider the parsing methods on it.

3.1.1.1 Explicit Alignments: DM, PSD

As discussed in Section 2.1.1.1, DM and PSD aims to represent all the semantic dependencies between words with fully covering the senentence. For each node in the graph, there will be an explicit words, multiword expression align to it. Hence, we call such1 kind of alignments as explicit alignments.

3.1.1.2 Implicit Anchoring in AMR

AMR tries to abstract the meaning representation away from the surface token. The absense of explicit anchoring can present difficulties for parsing. In this section, by extensive analysis on previous work AMR alignments, we show that AMR nodes can be implicitly aligned to the leixical tokens in a sentence.

AMR-to-String Alignments. A straightforward solution to find the missing anchoring in an AMR Graph is to align it with a sentence; We denote it as AMR-to-String alignment. ISI alignments Pourdamghani et al. (2014) first linearizes the AMR graph into a sequence, and then use IBM word alignment model Brown et al. (1993) to align the linearized sequence of concepts and relations with tokens in the sentence. According to the AMR annotation guidelines and error analysis of ISI aligner, some of the nodes or relations are evoked by subwords, e.g., the whole graph fragment `(p/possible-01 :polarity -)` is evoked by word “impossible", where the subword `"im-"` actually evoked the relation polarity and concept `"-"`; On the other side, sometimes concepts are evoked by multiple words, e.g., named entities, `(c/city :name (n/name :op1 "New":op2 "York"))`, which also happens in explict anchoring of DM and PSD. Hence, aligning and parsing with recategorized graph fragments are a natural solution in aligners and parsers. JAMR aligner Flanigan et al. (2014) uses a set of rules to greedily align single tokens, special entities and a set of multiple word expression to AMR graph fragments, which is widely used in previous AMR parsers (*e.g.* Flanigan et al., 2014; Wang et al., 2015; Artzi et al., 2015; Pust et al., 2015; Peng et al., 2015; Konstas et al., 2017; Wang and Xue, 2017). Other AMR-to-String Alignments exists, such as the extended HMM-based aligner. To consider more structure

info in the linearized AMR concepts, Wang and Xue (2017) proposed a Hidden Markov Model (HMM)-based alignment method with a novel graph distance. All of them report over 90% F-score on their own hand-aligned datasets, which shows that AMR-to-String alignments are almost token-level anchoring.

AMR-to-Dependency Alignments. Chen and Palmer (2017) first tries to align an AMR graph with a syntactic dependency tree. Szubert et al. (2018) conducted further analysis on dependency tree and AMR interface. It showed 97% of AMR edges can be evoked by words or the syntactic dependency edges between words. Those nodes in the dependency graph are anchored to each lexical token in the original sentence. Hence, this observation indirectly shows that AMR nodes can be aligned to the lexical tokens in the sentence.

Both AMR-to-String and AMR-to-dependency alignments shows that AMR nodes, including recategorized AMR graph fragements, do have implicit lexical anchoring. Based on this, Lyu and Titov (2018) propose to treat token-node alignments as discrete and exclusive alignment matrix and learn the latent alignment jointly with parsing. Recently, attention-based seq2graph model also achieved the state-of-the-art accuracy on AMR parsing Zhang et al. (2019b). However, whether the attention weights can be explained as AMR alignments needs more investigation in future.

3.1.1.3 Lexical-Anchoring

According to the bi-lexical dependency structures of DM and PSD, and implicit lexical token anchoring on AMR, the nodes/categorized graph fragments of DM, PSD, and AMR are anchored to surface lexical units in an explicit or implict way. Especially, those lexical units do not overlap with each other, and most of them are just single tokens, multiple word expression, or named entities. In other words, when parsing a sentence into DM, PSD, AMR graphs, tokens in the original sentence can be merged by looking up a lexicon dict when preprocessing and then may be considered as a single token for aligning or parsing.

3.1.2 Analysis on Phrasal Anchoring in UCCA and TOP

As a result, the semantic concepts in these meaning representations can be anchored to the individual lexical units of the sentence. `Flavor-1` includes Elementary Dependency

Structures (EDS, Oepen and Lønning, 2006) and Universal Conceptual Cognitive Annotation framework (UCCA, Abend and Rappoport, 2013b), which shows an explicit, many-to-many anchoring of semantic concepts onto sub-strings of the underlying sentence.

However, different from the lexical anchoring without overlapping, nodes in EDS and UCCA may align to larger overlapped word spans which involves syntactic or semantic pharsal structure. Nodes in UCCA do not have node labels or node properties, but all the nodes are anchored to the spans of the underlying sentence. Furthermore, the nodes in UCCA are linked into a hierarchical structure, with edges going between parent and child nodes. With certain exceptions (e.g. remote edges), the majority of the UCCA graphs are tree-like structures. According to the position as well as the anchoring style, nodes in UCCA can be classified into the following two types:

1. **Terminal nodes** are the leaf semantic concepts anchored to individual lexical units in the sentence

2. **Non-terminal nodes** are usually anchored to a span with more than one lexical units, thus usually overlapped with the anchoring of terminal nodes.

The similar classification of anchoring nodes also applies to the nodes in EDS, although they do not regularly form a recursive tree like UCCA. As the running example in Figure **??**, most of the nodes belongs to terminal nodes, which can be explicitly anchored to a single token in the original sentence. However, those bold non-teriminal nodes are anchored to a large span of words. For example, the node `"undef_q"` with span `<53:100>` is aligned to the whole substring starting from “other crops" to the end; The abstract node with label `imp_conj` are corresponding to the whole coordinate structure between `soybeans and rice`

3.1.3 Summary of Anchoring Analysis

In summary, by treating AMR as an implicitly lexically anchored MR, we propose a simplified taxonomy for parsing the five meaning representations.

`Lexical-anchoring`**: DM, PSD, AMR.** In Section 3.2, we proposed an unified graph-based parsing framework for `lexical-anchoring`, which use a latent alignment model to support both the explicit and implicit alignments.

`Phrasal-anchoring`**: UCCA, TOP.** In Section 3.3, we proposed an unified span-based CKY

parsing for both UCCA and TOP.

3.2 Latent Alignment Model for Lexical-Anchoring Graph-based Parsing

In this section, we will introduce the two-stage framework for parsing the DM, PSD, and AMR graphs §3.2.1. Then we resolve the alignment problem with a latent alignment model §**??**

3.2.1 Two-stage Graph-Based Parsing

Before formulating the graph-based model into a probabilistic model as Equation 3.1, we denote some notations: C, R are sets of concepts (nodes) and relations (edges) in the graph, and w is a sequence of tokens. $a \in \mathbb{Z}^m$ as the alignment matrix, each a_i is the index of aligned token where ith node aligned to. When modeling the negative log likelihood loss (NLL), with independence assumption between each node and edge, we decompose it into node- and edge-identification pipelines.

$$
\begin{aligned}
& NLL(P(C, R \mid w)) \\
& = -\log(P(C, R \mid w)) \\
& = -\log(\sum_a P(a) P(C, R \mid w, a)) \\
& = -\log\left(\sum_a P(a) P(R \mid w, a, C) P(C \mid w, a)\right) \\
& = -\log\left(\sum_a P(a) \prod_i^m P(c_i \mid h_{a_i}) \right. \\
& \qquad \left. \cdot \prod_{i,j=1}^m P(r_{ij} \mid h_{a_i}, c_i, h_{a_j}, c_j)\right)
\end{aligned}
\tag{3.1}
$$

In DM, PSD, and AMR, every token will only be aligned once. Hence, we train a joint model to maximize the above probability for both node identification $P(c_i \mid h_{a_i})$ §**??** and edge identification $P(r_{ij} \mid h_{a_i, c_i, h_{a_j}, c_j})$ §**??**. The alignment information is mainly used for training. We need to marginalize out the discrete alignment variable a to jointly learning the parameters in node identification and relation identification networks. We will introduce the latent alignment model in Section 3.2.2. Figure 3.1 summerize the unifed two-stage graph based parsing framework. In the following subsections, we will explain

the framework in more details.

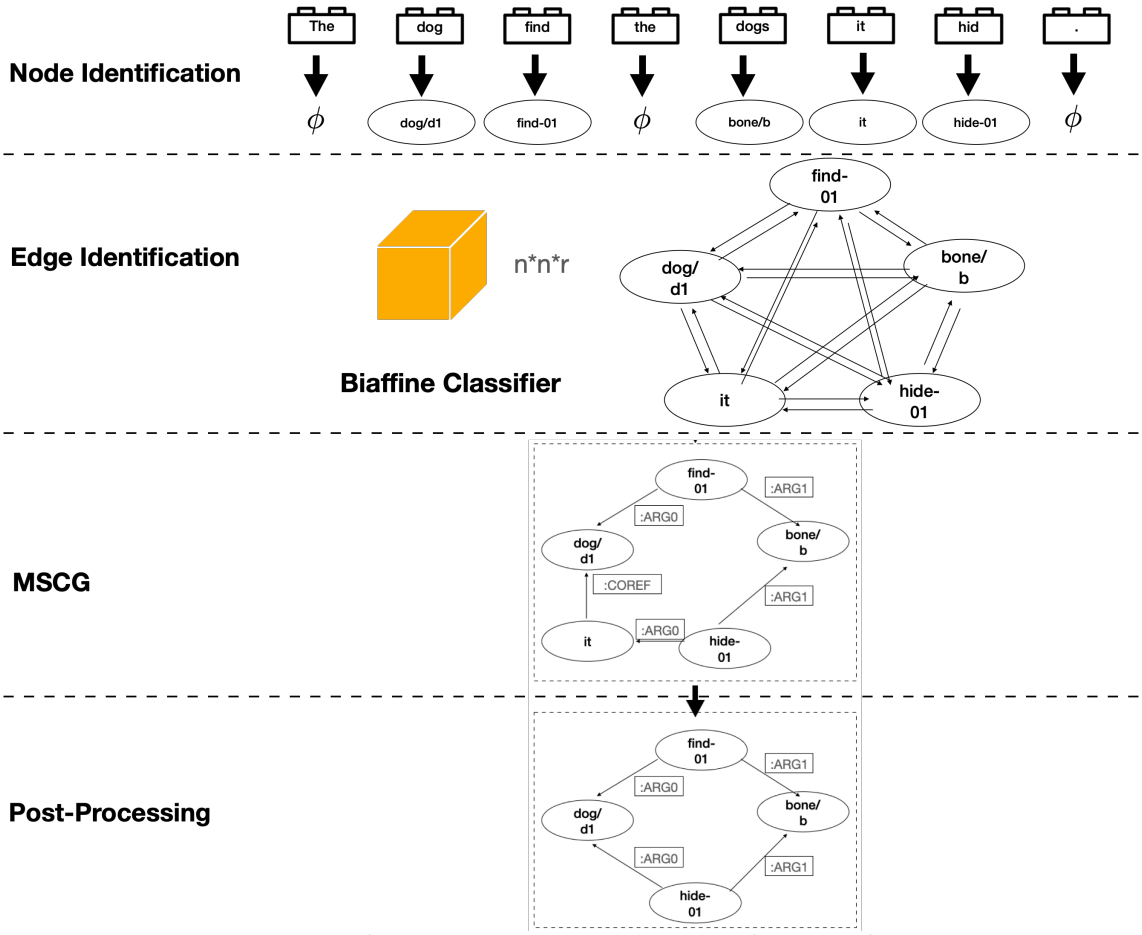

Figure 3.1: Architecture of graph-based model and inference, for running exmaple [wsj#0209013]

3.2.1.1 Node Identification

Node Identification predicts a concept c given a word. A concept can be either *NULL* (when there is no semantic node anchoring to that word, e.g., the word is dropped), or a node label (e.g., lemma, sense, POS, name value in AMR, frame value in PSD), or other node properties. One challenge in node identification is the data sparsity issue. Many of the labels are from open sets derived from the input token, e.g., its lemma. Moreover, some labels are constrained by a deterministic label set given the word. Hence, we designed a copy mechanism (Luong et al., 2014) in our neural network architecture to decide whether to copying deterministic label given a word or estimate a classification probability from a

fixed label set.

3.2.1.2 Edge Identification

By assuming the independence of each edge, we model the edges probabilites independently. Given two nodes and their underlying tokens, we predict the edge label as the semantic relation between the two concepts with a bi-affine classifier Dozat and Manning (2016).

3.2.1.3 Inference

In our two-stage graph-based parsing, after nodes are identified, edge identification only output a probility distribution over all the relations between identified nodes. However, we need to an inference algorithm to search for the ***maximum spanning connected graph*** from all the relations. We use Flanigan et al. (MSCG, 2014) to greedily select the most valuable edges from the identified nodes and their relations connecting them. As shown in Figure 3.1, an input sentence goes through preprocessing, node identification, edge identification, root identification, and MCSG to generate a final connected graph as structured output.

3.2.2 Latent Alignment Model

As the two-stage probablistic model shown in **??**, we need to marginalize all the alignment information a to learn the above two-stage nerual networks for node and edge identification. We do the following computing for explicit and implicit alignments respectively.

Explicit Alignments. For DM, PSD, with explicit alignments $a*$, we can simply use $P(a^*) = 1.0$ and other alignments $P(a|a \neq a^*) = 0.0$. In this case, with known alignment information, we don't need to worry the marginalization problem.

Implicit Alignments. However, For AMR, without gold alignments, one requires to compute all the valid alignments and then condition the node- and edge-identification methods on the alignments. However, it is computationally intractable to enumerate all combinatoral values for the discrete alignment variable. Hence, we estimate the latent alignments via variational inferece, which has been initialially used in Lyu and Titov (2018). In the following section, we firstly introduce the details of latent alignment model via contin-

uous relaxiation §3.2.2.1 and then we descibe the details of variational inference §3.2.2.2, and we propose a error fix for computing KL-divergence with implicit Gumbel-Sinkhorn distribution.

$$\begin{aligned} & NLL(P(C,R \mid w)) \\ & = -\log\left(\sum_{a} P(a) \prod_{i}^{m} P(c_i \mid h_{a_i}) \right. \\ & \left. \cdot \prod_{i,j=1}^{m} P(r_{ij} \mid h_{a_i}, c_i, h_{a_j}, c_j)\right) \end{aligned} \tag{3.2}$$

3.2.2.1 Continuous Relaxation for Discrete Alignments

$$\begin{aligned} & \log(P(C,R \mid w)) \geq \\ & E_Q[\log(P_\theta(c \mid w,a) P_\Phi(R \mid w,a,c))] \\ & - D_{KL}(Q_\Psi(a \mid c,R,w) \mid\mid P(a)) \end{aligned} \tag{3.3}$$

3.2.2.2 VAE, Perturb-and-Map, Gumble Sinkhorn

- Applying variational inference to reduce it into Evidence Lower Bound (ELBO, Kingma and Welling, 2013)

- The denominator Z_Ψ in Q can be estimated by Perturb-and-Max(MAP) Papandreou and Yuille (2011)

$$Q_\Psi(a \mid c,R,w) = \frac{\exp(\sum_{i=1}^{n} \phi(g_i, h_{a_i}))}{Z_\Psi(c,w)} \tag{3.4}$$

  Where $\phi(g_i, h_{a_i})$ score each alignment link between node i and the corresponding words, g_i is node encoding, and h_{a_i} is encoding for the aligned token.

- Discrete `argmax` of a permutation can be estimated by Gumbel-Softmax Sinkhorn Networks Mena et al. (2018); Lyu and Titov (2018)

3.3 Minimal Span-based CKY Parsing Framework

Let us now see our phrasal-anchoring parser for UCCA and TOP. We introduce the transformation we used to reduce UCCA and TOP parsing into a unified consituent tree parsing task, and finally introduce the detailed CKY model for the constituent parsing.

3.3.1 Graph to Constituent Tree Transformation

UCCA to Consistuent Tree. We propose to transform a graph into a constituent tree structure for parsing, which is also used in recent work Jiang et al. (2019). Figure 3.2 shows an example of transforming a UCCA graph into a constituent tree. The primary transformation assigns the original label of an edge to its child node. Then to make it compatible with parsers for standard PennTree Bank format, we add some auxiliary nodes such as special non-terminal nodes, TOP, HEAD, and special terminal nodes TOKEN and MWE. We remove all the "remote" annotation in UCCA since the constituent tree structure does not support reentrance. A fully compatible transformation should support both graph-to-tree and tree-to-graph transformation.

In our case, due to time constraints, we remove those remote edges and reentrance edges during training. Besides that, we also noticed that for multi-word expressions, the children of a parent node might not be in a continuous span (i.e., discontinuous constituent), which is also not supported by our constituent tree parser. Hence, when training the tree parser, by reattaching the discontinuous tokens to its nearest continuous parent nodes, we force every sub span are continuous in the transformed trees. We leave the postprocessing to recover those discontinuous as future work.

For inference, given an input sentence, we first use the trained constituent tree parsing model to parse it into a tree, and then we transform a tree back into a directed graph by assigning the edge label as its child's node label, and deleting those auxiliary labels, adding anchors to every remaining node.

TOP to Consistuent Tree. For the hierarchical dialog representation TOP, we also can tranform it to a constituent tree structure. Figure 3.3 shows the transformation process for the utterance "Driving directions to the Eagles game". In TOP tree shown in the up side of the figure, the leaf nodes are not single words as in consituent tree, and there are no other non-terminal nodes (such as part-of-speech tags) other than the intents and slots nodes. Hence, we decompose the original teriminal nodes in TOP into seperate tokens, and add a special parent node as `TOK` to each of the ternimal token node. Finally, it forms the constituent tree as shown in the bottom.

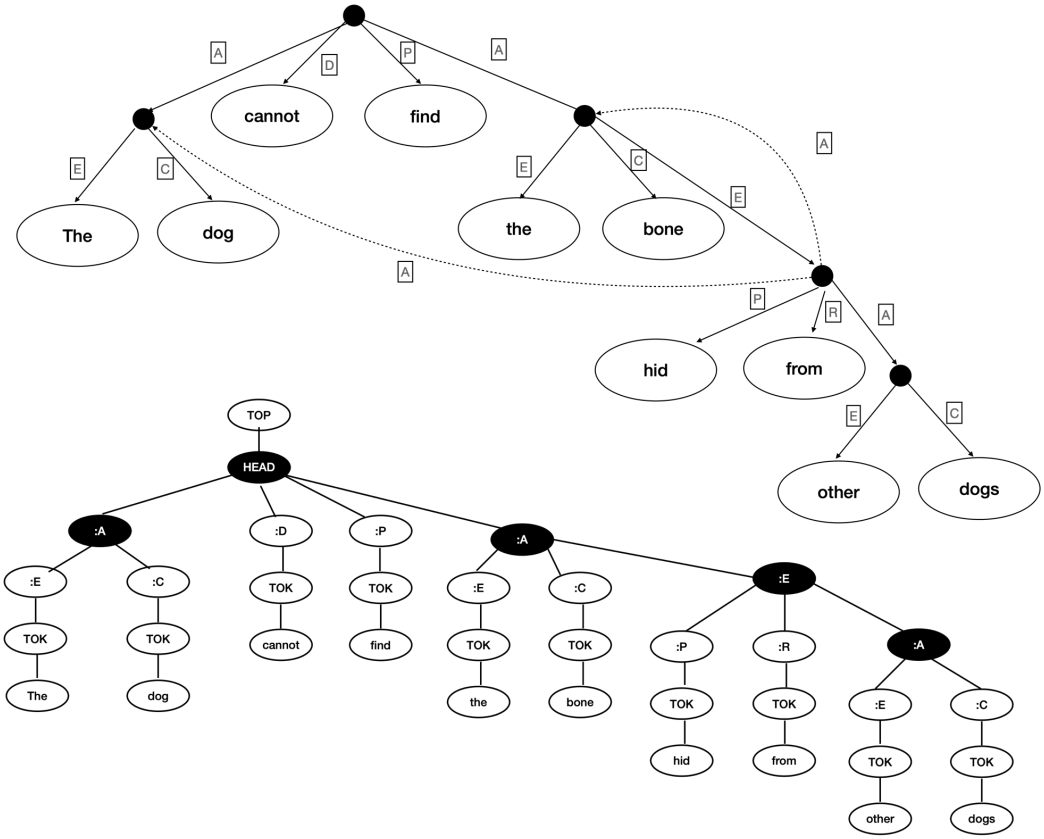

Figure 3.2: UCCA to Constituent Tree Transformation for [wsj#0209013]

3.3.2 A Unified Span-based Model for CKY Parsing

After transforming the UCCA graph into a constituent tree, we reduce the UCCA parsing into a constituent tree parsing problem. Similar to the previous work on UCCA constituent tree parsing Jiang et al. (2019), we use a minimal span-based CKY parser for constituent tree parsing. The intuition is to use dynamic programming to recursively split the span of a sentence recursively, as shown in Figure 3.2. The entire sentence can be splitted from top to bottom until each span is a single unsplittable tokens. For each node, we also need to assign a label. Two simplified assumptions are made when predicting the hole tree given a sentence. However, different with previous work, we use 8-layers with 8 heads transformer encoder, which shows better performance than LSTM in Kitaev and Klein (2018).

3.3.2.0.1 Tree Factorization In the graph-to-tree transformation, we move the edge label to its child node. By assuming the labels for each node are independent, we factorize the tree structure prediction as independent span-label prediction as Equation 3.5. How-

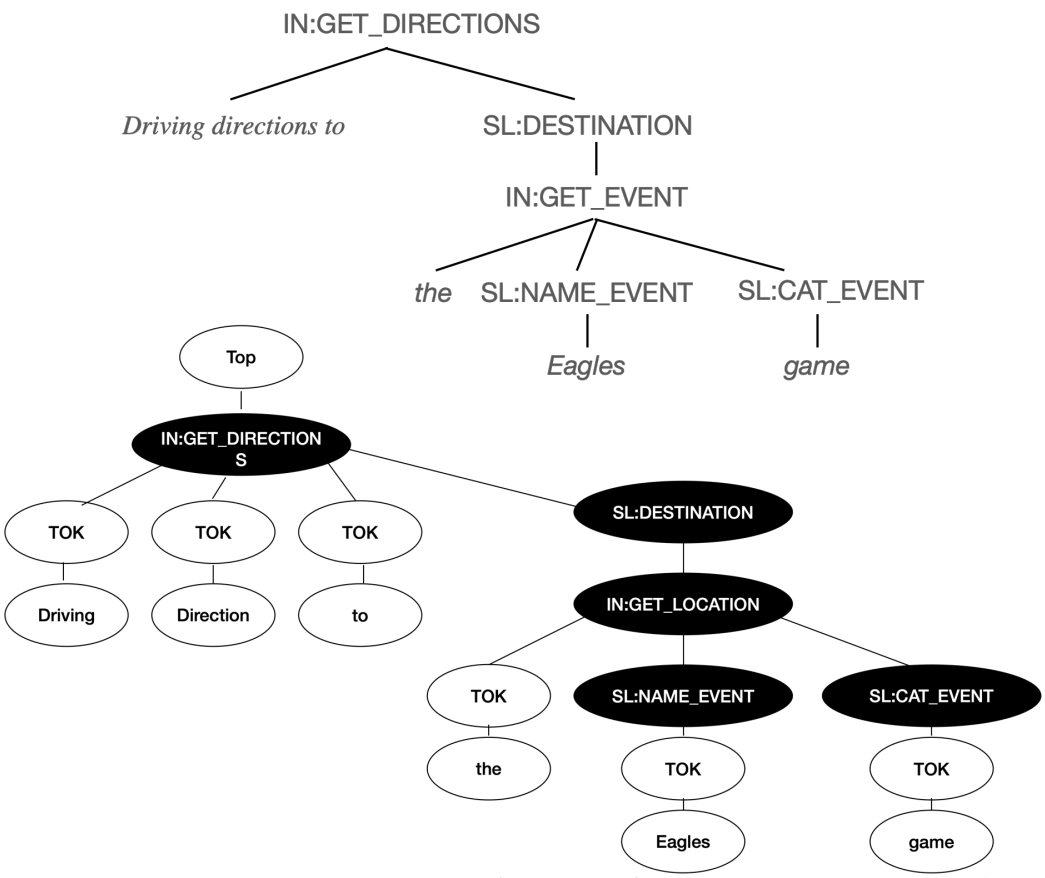

Figure 3.3: TOP to Constituent Tree Transformation for the utterance "Driving directions to the Eagles game"

ever, this assumption does not hold for UCCA. Please see more error analysis in §3.4.5

$$
\begin{aligned}
T^* &= \arg\max_T s(T) \\
s(T) &= \sum_{(i,j,l)\in T} s(i,j,l)
\end{aligned}
\tag{3.5}
$$

3.3.2.0.2 CKY Parsing By assuming the label prediction is independent of the splitting point, we can further factorize the whole tree as the following dynamic programming in Equation 3.6.

$$
\begin{aligned}
s_{\text{best}}(i,i+1) &= \max_l s(i,i+1,l) \\
s_{\text{best}}(i,j) &= \max_l s(i,j,l) \\
&\quad + \max_k [s_{\text{best}}(i,k) + s_{\text{best}}(k,j)]
\end{aligned}
\tag{3.6}
$$

3.3.3 Span Encoding

For each span (i, j), we represent the span encoding vector $v_{(i,j)} = [\vec{y}_j - \vec{y}_i] \oplus [\overleftarrow{y}_{j+1} - \overleftarrow{y}_{i+1}]$. $\oplus$ denotes vector concatenation. Assuming a bidirectional sentence encoder, we use the forward and backward encodings $\vec{y}_i$ and $\overleftarrow{y}_i$ of i_{th} word. Following the previous work, and we also use the loss augmented inference training. More details about the network architecture are in the Section 3.4.3

3.4 Experiments and Results

3.4.1 Dataset and Evaluation

For DM, PSD, EDS, we split the training set by taking WSJ section (00-19) as training, and section 20 as dev set. For other datasets, when developing and parameter tuning, we use splits with a ratio of 25:1:1. In our submitted model, we did not use multitask learning for training. Following the unified MRP metrics in the shared tasks, we train our model based on the development set and finally evaluate on the private test set. For more details of the metrics, please refer to the summarization of the MRP 2019 task Oepen et al. (2019),

	Lexicon Anchoring		
	DM	PSD	AMR
Top	1	$\geq$ 1 (11.56%)	1
Node Label	Lemma	Lemma(*)	Lemma(*) + NeType(143+)
Node Properties	POS semi(160*)_args(25)	POS wordid_sense(25)	constant values polarity, Named entity
Edge Label	(45)	(91)	(94+)
Edge Properties	N/A	N/A	N/A
Connectivity	True	True	True
Training Data	35656	35656	57885
Test Data	3269	3269	1998

Table 3.1: Detailed classifiers in our model, round bracket means the number of ouput classes of our classify, * means copy mechanism is used in our classifier. At the end of shared task, EDS are not fully supported to get an official results, we leave it as our future work.

3.4.2 Summary of Implementation

We summarize our implementation for five meaning representations as Table **??**. As we mentioned in the previous sections, we use latent-alignment graph-based parsing for lexical anchoring MRs (DM, PSD, AMR), and use CKY-based constituent parsing phrasal

	Phrase Anchoring	
	EDS	UCCA
Top	1	1
Node Label	_lemma(*)_semi_sense	N/A
Node Properties	 carg: constant value	N/A N/A
Edge Label	(45)	(15)
Edge Properties	N/A	“remote"
Connectivity	True	True
Training Data	35656	6485
Test Data	3269	1131

Table 3.2: Detailed classifiers in our model, round bracket means the number of ouput classes of our classify, * means copy mechanism is used in our classifier. At the end of shared task, EDS are not fully supported to get an official results, we leave it as our future work.

anchoring in MRs (UCCA, EDS). This section gives information about various decision for our models.

3.4.2.1 Top

The first row “Top" shows the numbers of root nodes in the graph. We can see that for PSD, 11.56% of graphs with more than 1 top nodes. In our system, we only predict one top node with a N (N is size of identified nodes) way classifier, and then fix this with a post-processing strategy. When our model predicts one node as the top node, and if we find additional coordination nodes with it, we add the coordination node also as the top node.

3.4.2.2 Node

Except for UCCA, all other four MRs have labeled nodes, the row “Node Label" shows the templates of a node label. For DM and PSD, the node label is usually the lemma of its underlying token. But the lemma is neither the same as one in the given companion data nor the predicted by Stanford Lemma Annotators. One common challenge for predicting the node labels is the open label set problem. Usually, the lemma is one of the morphology derivations of the original word. But the derivation rule is not easy to create manually. In our experiment, we found that handcrafted rules for lemma prediction only works worse than classification with copy mechanism, except for DM.

For AMR and EDS, there are other components in the node labels beyond the lemma.

Especially, the node label for AMR also contains more than 143 fine-grained named entity types; for EDS, it uses the full SEM-I entry as its node label, which requires extra classifiers for predicting the corresponding sense. In addition to the node label, the properties of the label also need to be predicted. Among them, node properties of DM are from the SEMI sense and arguments handler, while for PSD, senses are constrained the senses in the predefined the vallex lexicon.

3.4.2.3 Edge

Edge predication is another challenge in our task because of its large label set (from 45 to 94) as shown in row “Edge Label", the round bracket means the number of output classes of our classifiers. For Lexical anchoring MRs, edges are usually connected between two tokens, while phrasal anchoring needs extra effort to figure out the corresponding span with that node. For example, in UCCA parsing, To predict edge labels, we first predicted the node spans, and then node labels based that span, and finally we transform back the node label into edge label.

3.4.2.4 Connectivity

Beside the local label classification for nodes and edges, there are other global structure constraints for all five MRs: All the nodes and edges should eventually form a connected graph. For lexical anchoring, we use MSCG algorithm to find the maximum connected graph greedily; For phrasal anchoring, we use dynamic programming to decoding the constituent tree then deterministically transforming back to a connected UCCA Graph [2]

3.4.3 Model Setup

For lexical-anchoring model setup, our network mainly consists of node and edge prediction model. For AMR, DM, and PSD, they all use one layer Bi-directional LSTM for input sentence encoder, and two layers Bi-directional LSTM for head or dependent node encoder in the bi-affine classifier. For every sentence encoder, it takes a sequence of word embedding as input (We use 300 dimension Glove here), and then their output will pass a softmax layer to predicting output distribution. For the latent AMR model, to

[2]Due to time constraint, we ignored all the discontinuous span and remote edges in UCCA

model the posterior alignment, we use another Bi-LSTM for node sequence encoding. For phrasal-anchoring model setup, we follow the original model set up in Kitaev and Klein (2018), and we use 8-layers 8-headers transformer with position encoding to encode the input sentence.

For all sentence encoders, we also use the character-level CNN model as character-level embedding without any pre-trained deep contextualized embedding model. Equipping our model with Bert or multi-task learning is promising to get further improvement. We leave this as our future work.

Our models are trained with Adam Kingma and Ba (2015), using a batch size 64 for a graph-based model, and 250 for CKY-based model. Hyper-parameters were tuned on the development set, based on labeled F1 between two graphs. We exploit early-stopping to avoid over-fitting.

3.4.4 Results

At the time of official evaluation, we submitted three lexical anchoring parser, and then we submitted another phrasal-anchoring model for UCCA parsing during post-evaluation stage, and we leave EDS parsing as future work. The following sections are the official results and error breakdowns for lexical-anchoring and phrasal-anchoring respectively.

Official Results on Lexical Anchoring. Table 3.3 shows the official results for our lexical-anchoring models on AMR, DM, PSD. By using our latent alignment based AMR parser, our system ranked top 1 in the AMR subtask, and outperformed the top 5 models in large margin. Our parser on PSD ranked 6, but only 0.02% worse then the top 5 model. However, official results on DM and PSD shows that there is still around 2.5 points performance gap between our model and the top 1 model.

MR	Ours (P/R/F1)	Top 1/3/5 (F1)
AMR(1)	**75/71/73.38**	**73.38/71.97/71.72**
PSD(6)	89/89/**88.75**	90.76/89.91/**88.77**
DM(7)	93/92/92.14	94.76/94.32/93.74

Table 3.3: Official results overview on unified MRP metric, we selected the performance from top 1/3/5 system(s) for comparison

Official Results on Phrasal Anchoring. Table 3.4 shows that our span-based CKY model

for UCCA can achieve 74.00 F1 score on official test set, and ranked 5th. When adding ELMo Peters et al. (2018b) into our model, it can further improve almost 3 points on it.

MR	Ours (P/R/F1)	Top 1/3/5 (F1)
UCCA(5)	80.83/73.42/**76.94**	81.67/77.80/73.22
EDS	N/A	94.47/90.75/89.10

Table 3.4: Official results overview on unified MRP metric, we selected the performance from top 1/3/5 system(s) for comparison. It shows our UCCA model for post-evluation can rank 5th

3.4.5 Error Breakdown

Table 3.5, 3.6, 3.7 and 3.8 shows the detailed error breakdown of AMR, DM, PSD and UCCA respectively. Each column in the table shows the F1 score of each subcomponent in a graph: top nodes, node lables, node properties, node anchors, edge labels, and overall F1 score. No anchors for AMR, and no node label and propertis for UCCA. We show the results of MRP metric on two datasets. "all" denotes all the examples for that specific MR, while lpps are a set of 100 sentences from `The Little Prince`, and annotated in all five meaning representations. To better understand the performance, we also reported the official results from two baseline models TUPA Hershcovich and Arviv (2019) and ERG Oepen and Flickinger (2019).

	data	tops	labels	prop	edges	all
TUPA	all	63.95	57.20	22.31	36.41	44.73
single	lpps	71.96	55.52	26.42	36.38	47.04
TUPA	all	61.30	39.80	27.70	27.35	33.75
multi	lpps	72.63	50.11	20.25	33.12	43.38
Ours(1)	all	65.92	82.86	**77.26**	63.57	**73.38**
	lpps	72.00	78.71	58.93	63.96	71.11
Top 2	all	78.15	82.51	71.33	63.21	72.94
	lpps	83.00	76.24	51.79	60.43	69.03

Table 3.5: Our parser on AMR ranked 1st. This table shows the error breakdown when comparing to the baseline TUPA model and top 2 Che et al. (2019) in official results

3.4.5.1 Error Analysis on Lexical-Anchoring

As shown in Table 3.5, our AMR parser is good at predicting node properties and consistently perform better than other models in all subcomponent, except for top prediction.

	data	tops	labels	prop	anchors	edges	all
ERG	all	91.83	98.22	95.25	98.82	90.76	95.65
	lpps	95.00	97.32	97.75	99.46	92.71	97.03
Top 1	all	93.23	94.14	94.83	98.40	91.55	94.76
	lpps	96.48	91.85	94.36	99.04	93.28	94.64
Ours(7)	all	70.95	93.96	92.13	97.25	86.45	92.14
	lpps	84.00	90.55	91.91	97.96	87.24	91.82

Table 3.6: Our parser on DM ranked 7th. This table shows the error breakdown when comparing to the model ranked Top 1 Li et al. (2019) in official results

	data	tops	labels	prop	anchors	edges	all
Top 1	all	93.45	94.68	91.78	98.35	77.79	90.76
	lpps	93.33	91.73	84.37	98.40	77.63	88.34
Ours(6)	all	82.01	94.18	91.28	96.94	72.40	88.75
	lpps	85.85	90.48	82.63	95.97	73.60	85.83

Table 3.7: Our parser on PSD ranked 6th. This table shows the error breakdown when comparing to the model ranked top 1 Donatelli et al. (2019) in official results

Node properties in AMR are usually named entities, negation, and some other quantity entities. In our system, we recategorize the graph fragements into a single node, which helps for both alignments and structured inference for those special graph fragments. We see that all our 3 models perform almost as good as the top 1 model of each subtask on node label prediction, but they perform worse on top and edge prediction. It indicates that our bi-affine relation classifier are main bottleneck to improve. Moreover, we found the performance gap between node labels and node anchors are almost consistent, it indicates that improving our model on predicting `NULL` nodes may further improve node label prediction as well. Moreover, we believe that multi-task learning and pre-trained deep models such as BERT Devlin et al. (2018) may also boost the performance of our paser in future.

3.4.5.2 Error Analysis on Phrasal-Anchoring

According to Table 3.8, our model with ELMo works slightly better than the top 1 model on anchors prediction. It means our model is good at predicting the nodes in UCCA and we belive that it is also helpful for prediction phrasal anchoring nodes in EDS.

However, when predicting the edge and edge attributes, our model performs 7-8 points worse than the top 1 model. In UCCA, an edge label means the relation between a parent nodes and its children. In our UCCA transformation, we assign edge label as the node

	data	tops	anchors	edge	attr	all
TUPA single	all	78.73	69.17	16.96	15.18	27.56
	lpps	86.03	76.26	28.32	24.00	40.06
TUPA multi	all	84.92	65.74	12.99	9.07	23.65
	lpps	88.89	77.76	26.45	18.32	41.04
Che et al. (2019)	all	1.00	95.36	72.66	61.98	81.67
	lpps	1.00	96.99	73.08	48.37	82.61
Ours(*5)	all	98.85	94.92	60.17	0.00	**74.00**
	lpps	96.00	96.75	60.20	0.00	75.17
Ours + ELMo	all	99.38	95.70	64.88	0.00	**76.94**
	lpps	98.00	96.84	66.63	0.00	78.77

Table 3.8: Our UCCA parser in post-evaluation ranked 5th according to the original official evaluation results. This table shows the error breakdown when comparing to the model ranked top 1 Che et al. (2019) in official results. * denotes the ranking of post-evaluation results

label of its child and then predict with only child span encoding. Thus it actually misses important information from the parent node. Hence, in future, more improvement can be done to use both child and parent span encoding for label prediction, or even using another span-based bi-affine classifier for edge prediction, or remote edge recovering.

3.5 Chapter Summary

In summary, by analyzing the AMR alignments, we show that implicit AMR anchoring is actually lexical-anchoring based. Thus we propose to regroup five meaning representations as two groups: lexical-anchoring and phrasal-anchoring. For lexical anchoring, we suggest to parse DM, PSD, and AMR in a unified latent-alignment based parsing framework. Our submission ranked top 1 in AMR sub-task, ranked 6th and 7th in PSD and DM tasks. For phrasal anchoring, by reducing UCCA graph into a constituent tree-like structure, and then use the span-based CKY parsing to parse their tree structure, our method would rank 5th in the original official evaluation results.

CHAPTER 4

MODELING SENTENTIAL-ANCHORING FOR DIALOGUE IN THERAPY

Conversational agents have long been studied in the context of psychotherapy, going back to chatbots such as ELIZA (Weizenbaum, 1966) and PARRY (Colby, 1975). Research in modeling such dialogue has largely sought to simulate a participant in the conversation.

In this chapter, we argue for modeling dialogue ***observers*** instead of participants, and focus on psychotherapy. An observer could help an ***ongoing*** therapy session in several ways. First, by monitoring fidelity to therapy standards, a helper could guide both veteran and novice therapists towards better patient outcomes. Second, rather than generating therapist utterances, it could suggest the type of response that is appropriate. Third, it could alert a therapist about potentially important cues from a patient. Such assistance would be especially helpful in the increasingly prevalent online or text-based counseling services. [1]

We ground our study in a style of therapy called Motivational Interviewing (MI, Miller and Rollnick, 2003, 2012), which is widely used for treating addiction-related problems. To help train therapists, and also to monitor therapy quality, utterances in sessions are annotated using a set of behavioral codes called Motivational Interviewing Skill Codes (MISC, Miller et al., 2003). Table 4.1 shows standard therapist and patient (*i.e.*, client) codes with examples. Recent NLP work (Tanana et al., 2016; Xiao et al., 2016; Pérez-Rosas et al., 2017; Huang et al., 2018, *inter alia)* has studied the problem of using MISC to assess *completed* sessions. Despite its usefulness, automated post hoc MISC labeling does not address the desiderata for ongoing sessions identified above; such models use information from utterances yet to be said. To provide real-time feedback to therapists, we define two

[1] For example, Crisis Text Line (`https://www.crisistextline.org`), 7 Cups (`https://www.7cups.com`), etc.

Code	Count	Description	Examples
		Client Behavioral Codes	
FN	47715	Follow/ Neutral: unrelated to changing or sustaining behavior.	"You know, I didn't smoke for a while." "I have smoked for forty years now."
CT	5099	Utterances about changing unhealthy behavior.	"I want to stop smoking."
ST	4378	Utterances about sustaining unhealthy behavior.	"I really don't think I smoke too much."
		Therapist Behavioral Codes	
FA	17468	Facilitate conversation	"Mm Hmm.", "OK.","Tell me more."
GI	15271	Give information or feedback.	"I'm Steve.", "Yes, alcohol is a depressant."
RES	6246	Simple reflection about the client's most recent utterance.	C: "I didn't smoke last week" T: "Cool, you avoided smoking last week."
REC	4651	Complex reflection based on a client's conversation history.	C: "I didn't smoke last week." T: "You mean things begin to change".
QUC	5218	Closed question	"Did you smoke this week?"
QUO	4509	Open question	"Tell me more about your week."
MIA	3869	MI adherent, *e.g.*, affirmation, advising with permission, etc.	"You've accomplished a difficult task." "Is it OK if I suggested something?"
MIN	1019	MI non-adherent, *e.g.*, confront, advising without permission, etc.	"You hurt the baby's health for cigarettes?" "You ask them not to drink at your house."

Table 4.1: Distribution, description and examples of MISC labels.

complementary dialogue observers:

1. **Categorization**: Monitoring an ongoing session by predicting MISC labels for therapist and client utterances as they are made.
2. **Forecasting**: Given a dialogue history, forecasting the MISC label for the next utterance, thereby both alerting or guiding therapists.

Via these tasks, we envision a helper that offers assistance to a therapist in the form of MISC labels.

We study modeling challenges associated with these tasks related to: (1) representing words and utterances in therapy dialogue, (2) ascertaining relevant aspects of utterances and the dialogue history, and (3) handling label imbalance (as evidenced in Table 4.1). We develop neural models that address these challenges in this domain.

Experiments show that our proposed models outperform baselines by a large margin. For the categorization task, our models even outperform previous session-informed approaches that use information from future utterances. For the more difficult forecasting task, we show that even without having access to an utterance, the dialogue history pro-

vides information about its MISC label. We also report the results of an ablation study that shows the impact of the various design choices. [2].

In summary, in this chapter, we (1) factorize the dialog sequential structure via defining the tasks of categorizing and forecasting Motivational Interviewing Skill Codes to provide real-time assistance to therapists, (2) propose neural models for both tasks that outperform several baselines, and (3) show the impact of various modeling choices via extensive analysis.

4.1 Background and Motivation

As Table 4.1 shows, client labels mark utterances as discussing changing or sustaining problematic behavior (CT and ST, respectively) or being neutral (FN). Therapist utterances are grouped into eight labels, some of which (RES, REC) correlate with improved outcomes, while MI non-adherent (MIN) utterances are to be avoided. MISC labeling was originally done by trained annotators performing multiple passes over a session recording or a transcript. Recent NLP work speeds up this process by automatically annotating a completed MI session (*e.g.*, Tanana et al., 2016; Xiao et al., 2016; Pérez-Rosas et al., 2017).

Instead of providing feedback to a therapist after the completion of a session, can a dialogue observer provide online feedback? While past work has shown the helpfulness of post hoc evaluations of a session, prompt feedback would be more helpful, especially for MI non-adherent responses. Such feedback opens up the possibility of the dialogue observer influencing the therapy session. It could serve as an assistant that offers suggestions to a therapist (novice or veteran) about how to respond to a client utterance. Moreover, it could help alert the therapist to potentially important cues from the client (specifically, CT or ST).

4.2 Task Definitions and Structural Factorization

In this section, we will formally define the two NLP tasks corresponding to the vision in §4.1 using the conversation in Table 4.2 as a running example. Suppose we have an ongoing MI session with utterances $u_1, u_2, \cdots, u_n$: together, the dialogue history H_n. Each utterance u_i is associated with its speaker s_i, either C (client) or T (therapist). Each utter-

[2]The code is available online at `https://github.com/utahnlp/therapist-observer`.

i	s_i	u_i	l_i
1	T:	Have you used drugs recently?	QUC
2	C:	I stopped for a year, but relapsed.	FN
3	T:	You will suffer if you keep using.	MIN
4	C:	Sorry, I just want to quit.	CT
...		...	...

Table 4.2: An example of ongoing therapy session

ance is also associated with the MISC label l_i, which is the object of study. We will refer to the last utterance u_n as the *anchor*.

We will define two classification tasks over a fixed dialogue history with n elements — *categorization* §4.2.1 and *forecasting* §4.2.2. As the conversation progresses, the history will be updated with a sliding window. Since the therapist and client codes share no overlap, we will design separate models for the two speakers, giving us four settings in all.

4.2.1 Task 1: Categorization

The goal of this task is to provide real-time feedback to a therapist during an ongoing MI session. In the running example, the therapist's confrontational response in the third utterance is not MI adherent (MIN); an observer should flag it as such to bring the therapist back on track. The client's response, however, shows an inclination to change their behavior (CT). Alerting a therapist (especially a novice) can help guide the conversation in a direction that encourages it.

In essence, we have the following real-time classification task: *Given the dialogue history H_n which includes the speaker information, predict the MISC label l_n for the last utterance u_n.*

The key difference from previous work in predicting MISC labels is that we are restricting the input to the real-time setting. As a result, models can only use the dialogue history to predict the label, and in particular, we can not use models such as a conditional random field or a bi-directional LSTM that need both past and future inputs.

4.2.2 Task 2: Forecasting

A real-time therapy observer may be thought of as an expert therapist who guides a session with suggestions to the therapist. For example, after a client discloses their recent drug use relapse, a novice therapist may respond in a confrontational manner (which is

not recommended, and hence coded MIN). On the other hand, a seasoned therapist may respond with a complex reflection (REC) such as *"Sounds like you really wanted to give up and you're unhappy about the relapse."* Such an expert may also anticipate important cues from the client.

The forecasting task seeks to mimic the intent of such a seasoned therapist: *Given a dialogue history H_n and the next speaker's identity s_{n+1}, predict the MISC code l_{n+1} of the yet unknown next utterance u_{n+1}.*

The MISC forecasting task is a previously unstudied problem. We argue that forecasting the type of the next utterance, rather than selecting or generating its text as has been the focus of several recent lines of work (*e.g.*, Schatzmann et al., 2005; Lowe et al., 2015b; Yoshino et al., 2018), allows the human in the loop (the therapist) the freedom to creatively participate in the conversation within the parameters defined by the seasoned observer, and perhaps even rejecting suggestions. Such an observer could be especially helpful for training therapists (Imel et al., 2017). The forecasting task is also related to recent work on detecting antisocial comments in online conversations (Zhang et al., 2018) whose goal is to provide an early warning for such events.

4.2.3 Comparing Categorization and Forcasting Task

Take the dialogue segment in Table 4.2 as a running example, we compare the categorization and forcaseting task for each turn, with respect to the dialogue history and predicting target. Table 4.3 shows the detailed differences when we choose dialogue history window size as 3.

When the dialogue goes to the turn 3, both the categorization and forcasting task will the observe the current dialog history window as input $X = H_n$. However, the key difference is as follows: the categorization task is to predict the MISC code l_n for the last seen utterance u_n, while the forcasting task is to guess the future MISC code l_{n+1} for the unseen utterance u_{n+1}.

Worth to mention, when the dialogue goes to the turn 4, the dialog history will slide to the next window of size 3, as $H_n = \{u_2, u_3, u_4\}$. The u_1 will be truncated due the sliding window. We limit the dialogue window because the whole therapy dialogue session may last for 500 utterances, where current neural models such as RNN and transformer can not

handle the long context well. In this thesis study Section 4.5.2, we compare the window size as 8 and 16 for our models. We leave the extrem long dialogue context encoding as the future work.

Turn	$X = H_n$	Categorization $Y = l_n$	Forcasting $Y = l_{n+1}$
1	$\{u_1\}$	QUC	FN
2	$\{u_1, u_2\}$	FN	MIN
3	$\{u_1, u_2, u_3\}$	MIN	CT
4	$\{u_2, u_3, u_4\}$	CT	RES

Table 4.3: Differences between the categorization task and the forcasting task, when choosing a window size as 3 to factorize the dialog sequential flow

4.3 Models for MISC Prediction

Modeling the categorization and forcasting tasks defined in §4.2 requires addressing four questions: (1) How do we encode a dialogue and its utterances? (2) Can we discover discriminative words in each utterance? (3) Can we discover which of the previous utterances are relevant? (4) How do we handle label imbalance in our data? Many recent advances in neural networks can be seen as plug-and-play components. To facilitate the comparative study of models, we will describe components that address the above questions. In the rest of the chapter, we will use **boldfaced** terms to denote vectors and matrices and SMALL CAPS to denote component names.

4.3.1 Encoding Dialogue

Since both our tasks are classification tasks over a dialogue history, our goal is to convert the sequence of utterences into a single vector that serves as input to the final classifier.

We will use a hierarchical recurrent encoder (Li et al., 2015a; Sordoni et al., 2015; Serban et al., 2016, and others) to encode dialogues, specifically a hierarchical gated recurrent unit (HGRU) with an utterance and a dialogue encoder. We use a bidirectional GRU over word embeddings to encode utterances. As is standard, we represent an utterance u_i by concatenating the final forward and reverse hidden states. We will refer to this utterance vector as v_i. Also, we will use the hidden states of each word as inputs to the attention

components in §4.3.2. We will refer to such contextual word encoding of the j^{th} word as $\boldsymbol{v}_{ij}$. The dialogue encoder is a unidirectional GRU that operates on a concatenation of utterance vectors $\boldsymbol{v}_i$ and a trainable vector representing the speaker s_i.[3] The final state of the GRU aggregates the entire dialogue history into a vector $\boldsymbol{H}_n$.

The HGRU skeleton can be optionally augmented with the word and dialogue attention described next. All the models we will study are two-layer MLPs over the vector $\boldsymbol{H}_n$ that use a ReLU hidden layer and a softmax layer for the outputs.

4.3.2 Word-level Attention

Certain words in the utterance history are important to categorize or forecast MISC labels. The identification of these words may depend on the utterances in the dialogue. For example, to identify that an utterance is a simple reflection (RES) we may need to discover that the therapist is mirroring a recent client utterance; the example in table 4.1 illustrates this. Word attention offers a natural mechanism for discovering such patterns.

We can unify a broad collection of attention mechanisms in NLP under a single high level architecture Galassi et al. (2019). We seek to define attention over the word encodings $\boldsymbol{v}_{ij}$ in the history (called queries), guided by the word encodings in the anchor $\boldsymbol{v}_{nk}$ (called keys). The output is a sequence of attention-weighted vectors, one for each word in the i^{th} utterance. The j^{th} output vector $\boldsymbol{a}_j$ is computed as a weighted sum of the keys:

$$\boldsymbol{a}_{ij} = \sum_k \alpha_j^k \boldsymbol{v}_{nk} \tag{4.1}$$

The weighting factor α_j^k is the attention weight between the j^{th} query and the k^{th} key, computed as

$$\alpha_j^k = \frac{\exp\left(f_m(\boldsymbol{v}_{nk}, \boldsymbol{v}_{ij})\right)}{\sum_{j'} \exp\left(f_m(\boldsymbol{v}_{nk}, \boldsymbol{v}_{ij'})\right)} \tag{4.2}$$

Here, f_m is a match scoring function between the corresponding words, and different choices give us different attention mechanisms.

Finally, a combining function f_c combines the original word encoding $\boldsymbol{v}_{ij}$ and the above attention-weighted word vector $\boldsymbol{a}_{ij}$ into a new vector representation $\boldsymbol{z}_{ij}$ as the final representation of the query word encoding:

[3] For the dialogue encoder, we use a unidirectional GRU because the dialogue is incomplete. For words, since the utterances are completed, we can use a BiGRU.

Method	f_m	f_c
BiDAF	$\boldsymbol{v}_{nk}\boldsymbol{v}_{ij}^T$	$[\boldsymbol{v}_{ij};\ \boldsymbol{a}_{ij};$ $\boldsymbol{v}_{ij} \odot \boldsymbol{a}_{ij};\ \boldsymbol{v}_{ij} \odot \boldsymbol{a}']$
GMGRU	$\boldsymbol{w}^e \tanh(\boldsymbol{W}^k\boldsymbol{v}_{nk}$ $+ \boldsymbol{W}^q[\boldsymbol{v}_{ij}; \boldsymbol{h}_{j-1}])$	$[\boldsymbol{v}_{ij}; \boldsymbol{a}_{ij}]$

Table 4.4: Summary of word attention mechanisms. We simplify BiDAF with multiplicative attention between word pairs for f_m, while GMGRU uses additive attention influenced by the GRU hidden state. The vector $\boldsymbol{w}_e \in \mathbb{R}^d$, and matrices $\boldsymbol{W}^k \in \mathbb{R}^{d\times d}$ and $\boldsymbol{W}^q \in \mathbb{R}^{2d\times 2d}$ are parameters of the BiGRU. The vector $\boldsymbol{h}_{j-1}$ is the hidden state from the BiGRU in GMGRU at previous position $j-1$. For combination function, BiDAF concatenates bidirectional attention information from both the key-aware query vector $\boldsymbol{a}_{ij}$ and a similarly defined query-aware key vector $\boldsymbol{a}'$. GMGRU uses simple concatenation for f_c.

$$\boldsymbol{z}_{ij} = f_c(\boldsymbol{v}_{ij}, \boldsymbol{a}_{ij}) \tag{4.3}$$

The attention module, identified by the choice of the functions f_m and f_c, converts word encodings in each utterance $\boldsymbol{v}_{ij}$ into attended word encodings $\boldsymbol{z}_{ij}$. To use them in the HGRU skeleton, we will encode them a second time using a BiGRU to produce attention-enhanced utterance vectors. For brevity, we will refer to these vectors as $\boldsymbol{v}_i$ for the utterance u_i. If word attention is used, these attended vectors will be treated as word encodings.

To complete this discussion, we need to instantiate the two functions. We use two commonly used attention mechanisms: BiDAF Seo et al. (2016) and gated matchLSTM Wang et al. (2017). For simplicity, we replace the sequence encoder in the latter with a BiGRU and refer to it as GMGRU. Table 4.4 shows the corresponding definitions of f_c and f_m. We refer the reader to the original papers for further details. In subsequent sections, we will refer to the two attended versions of the HGRU as BIDAF^H and GMGRU^H.

4.3.3 Utterance-level Attention

While we assume that the history of utterances is available for both our tasks, not every utterance is relevant to decide a MISC label. For categorization, the relevance of an utterance to the anchor may be important. For example, a complex reflection (REC) may depend on the relationship of the current therapist utterance to one or more of the previous client utterances. For forecasting, since we do not have an utterance to label, several previous utterances may be relevant. For example, in the conversation in Table **??**, both u_2 and u_4 may be used to forecast a complex reflection.

To model such utterance-level attention, we will employ the multi-head, multi-hop attention mechanism used in Transformer networks Vaswani et al. (2017). As before, due to space constraints, we refer the reader to the original work for details. We will use the $(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V})$ notation from the original paper here. These matrices represent a query, key and value respectively. The multi-head attention is defined as:

$$\text{Multihead}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = [\text{head}_1; \cdots; \text{head}_h]\boldsymbol{W}^O \tag{4.4}$$

$$\text{head}_i = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{W}_i^Q \left(\boldsymbol{K}\boldsymbol{W}_i^K\right)^T}{\sqrt{d_k}}\right)\boldsymbol{V}\boldsymbol{W}_i^V$$

The $\boldsymbol{W}_i$'s refer to projection matrices for the three inputs, and the final $\boldsymbol{W}^o$ projects the concatenated heads into a single vector.

The choices of the query, key and value defines the attention mechanism. In our work, we compare two variants: *anchor-based attention*, and *self-attention*. The anchor-based attention is defined by $Q = [\boldsymbol{v}_n]$ and $K = V = [\boldsymbol{v}_1 \cdots \boldsymbol{v}_n]$. Self-attention is defined by setting all three matrices to $[\boldsymbol{v}_1 \cdots \boldsymbol{v}_n]$. For both settings, we use four heads and stacking them for two hops, and refer to them as SELF$_{42}$ and ANCHOR$_{42}$.

4.3.4 Predicting and Training

Predicting. From up to the bottom, every component will produce some of useful representation for inferences in our tasks. The dialogue encoding vector $\boldsymbol{H}_n$, as the final of the unidirectional GRU, it is also the contextual utterance encoding $\boldsymbol{h}_{u_n}$ of u_n. Hence H_n can be directly used as a representaion of u_n for classification in annotating tasks, also can be used as a represention of whole dialogue for forecasting task. Hence in HGRU setting, we always use $\boldsymbol{H}_n$ as the base option as input for inference.

However, for CON skeleton, the final state C_n doest not exactly represent the segement u_n in the whole concatnated dalogue. Hence, we concatenate the hidden state of the start position (0) and end position (T) of u_n into $\boldsymbol{v}_n^{seg} = [h_{u_n^0}; h_{u_n^T}]$, which is contextual utterance encoding in CON mode.

Beside the above $\boldsymbol{H}_n$ and $\boldsymbol{C}_n$ contextual utterance encoding in dialogue level, our components also produced the original utterance encoding $\boldsymbol{v}_n$ from utterance encoder. Whats'more, in **CON** mode, we can use history-aware utterance encoding $\boldsymbol{v}_n^{wordatt}$ While in HGRU, it produced a self-attentive utterance encoding. We denote it as $\boldsymbol{v}_n^{selfatt}$.

Skeleton	Annotating	Forecasting
CON	$\boldsymbol{v}_n^{seg}$,$\boldsymbol{v}^{wordatt_n}$, $\boldsymbol{v}_n$	$\boldsymbol{C}_n$
HGRU	$\boldsymbol{H}_n$,$\boldsymbol{v}_n^{selfatt}$,$\boldsymbol{v}_n$	$\boldsymbol{H}_n$

Table 4.5: Input options for annotating and forecasting tasks based on CON and HGRU skeletons.

We summarize the option input encodings for inference in Table **??**. There are two ways to scoring withthese inputs, one is to score the concatenated those vectors together, denoted as $\text{concat}(A, B) = \text{MLP}([A; B)$; The other one is scoring each of them first and then add the scores up as the final scores, such as $\text{add}(A, B) = \text{MLP}(A) + \text{MLP}(B)$.

4.3.5 Addressing Label Imbalance

From Table 4.1, we see that both client and therapist labels are imbalanced. Moreover, rarer labels are more important in both tasks. For example, it is important to identify CT and ST utterances. For therapists, it is crucial to flag MI non-adherent (MIN) utterances; seasoned therapists are trained to avoid them because they correlate negatively with patient improvements. If not explicitly addressed, the frequent but less useful labels can dominate predictions.

To address this, we extend the focal loss (FL Lin et al., 2017) to the multiclass case. For a label l with probability produced by a model p_t, the loss is defined as

$$\text{FL}(p_t) = -\alpha_t(1 - p_t)^{\gamma} \log(p_t) \tag{4.5}$$

In addition to using a label-specific balance weight α_t, the loss also includes a modulating factor $(1 - p_t)^{\gamma}$ to dynamically downweight well-classified examples with $p_t \gg 0.5$. Here, the α_t's and the γ are hyperparameters. We use FL as the default loss function for all our models.

4.4 Experiments

The original psychotherapy sessions were collected for both clinical trials and Motivational Interviewing dissemination studies including hospital settings (Roy-Byrne et al., 2014), outpatient clinics (Baer et al., 2009), college alcohol interventions (Tollison et al., 2008; Neighbors et al., 2012; Lee et al., 2013, 2014). All sessions were annotated with the

Motivational Interviewing Skills Codes (MISC) Atkins et al. (2014). We use the train/test split of Can et al. (2015); Tanana et al. (2016) to give 243 training MI sessions and 110 testing sessions. We used 24 training sessions for development. As mentioned in §4.1, all our experiments are based on the MISC codes grouped by Xiao et al. (2016).

4.4.1 Preprocessing and Model Setup

Preprocessing. An MI session contains about 500 utterances on average. We use a sliding window of size $N = 8$ utterances with padding for the initial ones. We assume that we always know the identity of the speaker for all utterances. Based on this, we split the sliding windows into a client and therapist windows to train separate models. We tokenized and lower-cased utterances using spaCy Honnibal and Montani (2017). To embed words, we concatenated 300-dimensional Glove embeddings Pennington et al. (2014) with ELMo vectors Peters et al. (2018c).

Model Setup. We use 300-dimensional Glove embeddings pre-trained on 840B tokens from Common Crawl Pennington et al. (2014). We do not update the embedding during training. Tokens not covered by Glove are using a randomly initialized UNK embedding. We also use character-level deep contextualized embedding ELMo 5.5B model by concatenating the corresponding ELMo word encoding after the word embedding vector. For speaker information, we randomly initialize them with 8 dimensional vectors and update them during training. We used a dropout rate of 0.3 for the embedding layers.

We trained all models using Adam Kingma and Ba (2015) with learning rate chosen by cross validation between $[1e^{-4}, 5 * 1e^{-4}]$, gradient norms clipping from at $[1.0, 5.0]$, and minibatch sizes of 32 or 64. We use the same hidden size for both utterance encoder, dialogue encoder and other attention memory hidden size; it has been selected from $\{64, 128, 256, 512\}$. We set a smaller dropout 0.2 for the final two fully connected layers. All the models are trained for 100 epochs with early-stoping based on macro F_1 over development results.

4.4.2 Results

Our goal is to discover the best client and therapist models for the two tasks. We first summarize the best configuration and the corresponding performance of our best models

for both categorzing and forecasting MISC codes in Table 4.6 with precision, recall and F_1 for each codes. Then, we also show the performance of these models against various baselines.

Label	Categorizing			Forecasting		
	P	R	F_1	P	R	F_1
FN	92.5	86.8	89.6	90.8	80.3	85.2
CT	34.8	44.7	39.1	18.9	28.6	22.7
ST	28.2	39.9	33.1	19.5	33.7	24.7
FA	95.1	94.7	94.9	70.7	73.2	71.9
RES	50.3	61.3	55.2	20.1	18.8	19.5
REC	52.8	55.5	54.1	19.2	34.7	24.7
GI	74.6	75.1	74.8	52.8	67.5	59.2
QUC	80.6	70.4	75.1	36.2	24.3	29.1
QUO	85.3	81.2	83.2	27.0	11.8	16.4
MIA	61.8	52.4	56.7	27.0	10.6	15.2
MIN	27.7	28.5	28.1	17.2	10.2	12.8

Table 4.6: Performance of our proposed models with respect to precision, recall and F_1 on categorizing and forecasting tasks for client and therapist codes

Best Models. We identified the following best configurations using F_1 score on the development set:

1. **Categorization**: For client, the best model does not need any word or utterance attention. For the therapist, it uses GMGRU^H for word attention and ANCHOR_{42} for utterance attention. We refer to these models as $\mathcal{C}_C$ and $\mathcal{C}_T$ respectively
2. **Forecasting**: For both client and therapist, the best model uses no word attention, and uses SELF_{42} utterance attention. We refer to these models as $\mathcal{F}_C$ and $\mathcal{F}_T$ respectively.

Results on Categorization. Tables 4.7 and 4.8 show the performance of the $\mathcal{C}_C$ and $\mathcal{C}_T$ models and the baselines. For both therapist and client categorization, we compare the best models against the same set of baselines. The majority baseline illustrates the severity of the label imbalance problem. Xiao et al. (2016), $\text{BiGRU}_{\text{generic}}$, Can et al. (2015) and Tanana et al. (2016) are the previous published baselines. The best results of previous published baselines are underlined. The last row Δ in each table lists the changes of our best model

from them. $\text{BiGRU}_{\text{ELMo}}$, CONCAT^{C}, GMGRU^{H} and BiDAF^{H} are new baselines we define below.

Method	macro	FN	CT	ST
Majority	30.6	**91.7**	0.0	0.0
Xiao et al. (2016)	50.0	87.9	32.8	29.3
$\text{BiGRU}_{\text{generic}}$	50.2	87.0	35.2	28.4
$\text{BiGRU}_{\text{ELMo}}$	52.9	87.6	**39.2**	32.0
Can et al. (2015)	44.0	91.0	20.0	21.0
Tanana et al. (2016)	48.3	89.0	29.0	27.0
CONCAT^{C}	51.8	86.5	38.8	30.2
GMGRU^{H}	52.6	89.5	37.1	31.1
BiDAF^{H}	50.4	87.6	36.5	27.1
$\mathcal{C}_C$	**53.9**	89.6	39.1	**33.1**
$\Delta = \mathcal{C}_C - \underline{\text{score}}$	+3.5	-2.1	+3.9	+3.8

Table 4.7: Main results on categorizing client codes, in terms of macro F_1, and F_1 for each client code. Our model $\mathcal{C}_C$ uses final dialogue vector H_n and current utterance vector v_n as input of MLP for final prediction. We found that predicting using $\text{MLP}(H_n) + \text{MLP}(v_n)$ performs better than just $\text{MLP}(H_n)$.

Method	macro	FA	RES	REC	GI	QUC	QUO	MIA	MIN
Majority	5.87	47.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Xiao et al. (2016)	59.3	94.7	50.2	48.3	71.9	68.7	80.1	54.0	6.5
$\text{BiGRU}_{\text{generic}}$	60.2	94.5	50.5	49.3	72.0	70.7	80.1	54.0	10.8
$\text{BiGRU}_{\text{ELMo}}$	62.6	94.5	51.6	49.4	70.7	72.1	80.8	57.2	24.2
Can et al. (2015)	-	94.0	49.0	45.0	74.0	72.0	81.0	-	-
Tanana et al. (2016)	-	94.0	48.0	39.0	69.0	68.0	77.0	-	-
CONCAT^{C}	61.0	94.5	54.6	34.3	73.3	73.6	81.4	54.6	22.0
GMGRU^{H}	64.9	94.9	**56.0**	54.4	**75.5**	**75.7**	**83.0**	**58.2**	21.8
BiDAF^{H}	63.8	94.7	55.9	49.7	75.4	73.8	80.7	56.2	24.0
$\mathcal{C}_T$	**65.4**	**95.0**	55.7	**54.9**	74.2	74.8	82.6	56.6	**29.7**
$\Delta = \mathcal{C}_T - \underline{\text{score}}$	+5.2	+0.3	+3.9	+3.8	+0.2	+2.8	+1.6	+2.6	+18.9

Table 4.8: Main results on categorizing therapist codes, in terms of macro F_1, and F_1 for each therapist code. Models are the same as Table 4.7, but tuned for therapist codes. For the two grouped MISC set MIA and MIN, their results are not reported in the original work due to different setting.

The first set of baselines (above the line) do not encode dialogue history and use only the current utterance encoded with a BiGRU. The work of Xiao et al. (2016) falls in this

category, and uses a 100-dimensional domain-specific embedding with weighted cross-entropy loss. Previously, it was the best model in this class. We also re-implemented this model to use either ELMo or Glove vectors with focal loss.[4]

The second set of baselines (below the line) are models that use dialogue context. Both Can et al. (2015) and Tanana et al. (2016) use well-studied linguistic features and then tagging the current utterance with both past and future utterance with CRF and MEMM, respectively. To study the usefulness of the hierarchical encoder, we implemented a model that uses a bidirectional GRU over a long sequence of flattened utterance. We refer to this as CONCAT^{C}. This model is representative of the work of Huang et al. (2018), but was reimplemented to take advantage of ELMo.

For categorizing client codes, BiGRU$_{\text{ELMo}}$ is a simple but robust baseline model. It outperforms the previous best no-context model by more than 2 points on macro F_1. Using the dialogue history, the more sophisticated model $\mathcal{C}_C$ further gets 1 point improvement. Especially important is its improvement on the infrequent, yet crucial labels CT and ST. It shows a drop in the F_1 on the FN label, which is essentially considered to be an unimportant, background class from the point of view of assessing patient progress. For therapist codes, as the highlighted numbers in Table 4.8 show, only incorporating GMGRU-based word-level attention, GMGRU^{H} has already outperformed many baselines, our proposed model $\mathcal{F}_T$ which uses both GMGRU-based word-level attention and anchor-based multi-head multihop sentence-level attention can further achieve the best overall performance. Also, note that our models outperform approaches that take advantage of future utterances.

For both client and therapist codes, concatenating dialogue history with CONCAT^{C} always performs worse than the hierarchical method and even the simpler BiGRU$_{\text{ELMo}}$.

Results on Forecasting. Since the forecasting task is new, there are no published baselines to compare against. Our baseline systems essentially differ in their representation of dialogue history. The model CONCAT^{F} uses the same architecture as the model CONCAT^{C} from the categorizing task. We also show comparisons to the simple HGRU model and

[4]Other related work in no context exists (e.g., Pérez-Rosas et al., 2017; Gibson et al., 2017), but they either do not outperform Xiao et al. (2016) or use different data.

Method	Dev		Test			
	CT	ST	macro	FN	CT	ST
CONCAT^F	20.4	30.2	43.6	84.4	23.0	**23.5**
HGRU	19.9	31.2	**44.4**	85.7	**24.9**	22.5
GMGRU^H	19.4	30.5	44.3	87.1	23.3	22.4
$\mathcal{F}_C$	**21.1**	**31.3**	44.3	85.2	24.7	22.7

Table 4.9: Main results on forecasting client codes, in terms of F_1 for ST, CT on dev set, and macro F_1, and F_1 for each client code on the test set.

Method	Recall	F_1								
	R@3	macro	FA	RES	REC	GI	QUC	QUO	MIA	MIN
CONCAT^F	72.5	23.5	63.5	0.6	0.0	53.7	27.0	15.0	18.2	9.0
HGRU	76.0	28.6	71.4	12.7	**24.9**	58.3	28.8	5.9	**17.4**	9.7
GMGRU^H	76.6	26.6	**72.6**	10.2	20.6	58.8	27.4	6.0	8.9	7.9
$\mathcal{F}_T$	**77.0**	**31.1**	71.9	**19.5**	24.7	**59.2**	**29.1**	**16.4**	15.2	**12.8**

Table 4.10: Main results on forecasting therapist codes, in terms of Recall@3, macro F_1, and F_1 for each label on test set

the GMGRU^H model that uses a gated matchGRU for word attention.[5]

Tables **??** (a,b) show our forecasting results for client and therapist respectively. For client codes, we also report the CT and ST performance on the development set because of their importance. For the therapist codes, we also report the recall@3 to show the performance of a suggestion system that displayed three labels instead of one. The results show that even without an utterance, the dialogue history conveys signal about the next MISC label. Indeed, the performance for some labels is even better than some categorization baseline systems. Surprisingly, word attention (GMGRU^H) in Table **??** did not help in forecasting setting, and a model with the SELF$_{42}$ utterance attention is sufficient. For the therapist labels, if we always predicted the three most frequent labels (FA, GI, and RES), the recall@3 is only 67.7, suggesting that our models are informative if used in this suggestion-mode.

[5]The forecasting task bears similarity to the next utterance selection task in dialogue state tracking work Yoshino et al. (2018). In preliminary experiments, we found that the Dual-Encoder approach used for that task consistently underperformed the other baselines described here.

4.5 Analysis and Ablations

This section reports error analysis and an ablation study of our models on the development set. The appendix shows a comparison of pretrained domain-specific ELMo/glove with generic ones and the impact of the focal loss compared to simple or weighted cross-entropy.

4.5.1 Label Confusion and Error Breakdown

Figure 4.1 shows the confusion matrix for the client categorization task. The confusion between FN and CT/ST is largely caused by label imbalance. There are 414 CT examples that are predicted as ST and 391 examples vice versa. To further understand their confusion, we selected 100 of each for manual analysis. We found four broad categories of confusion, shown in Table 4.11.

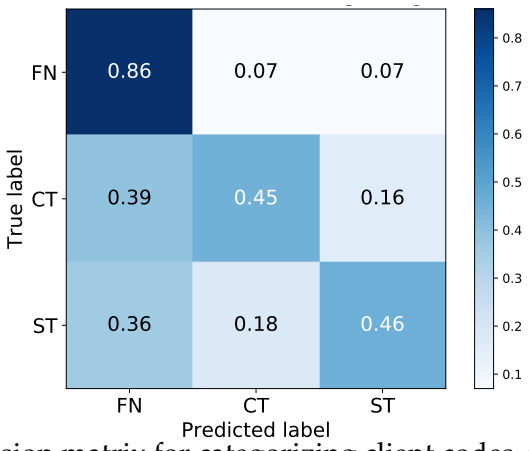

Figure 4.1: Confusion matrix for categorizing client codes, normalized by row.

The first category requires more complex reasoning than just surface form matching. For example, the phrase *seven out of ten* indicates that the client is very confident about changing behavior; the phrase *wind down after work* indicates, in this context, that the client drinks or smokes after work. We also found that the another frequent source of error is incomplete information. In a face-to-face therapy session, people may use concise and effient verbal communication, with guestures and other body language conveying information without explaining details about, for example, coreference. With only textual context, it is difficult to infer the missing information. The third category of errors is introduced when speech is transcribed into text. The last category is about ambivalent

Category and Explaination	Client Examples (Gold MISC)
Reasoning is required to understand whether a client wants to change behavior, even with full context (50,42)	T: On a scale of zero to ten how confident are you that you can implement this change? C: I don't know, seven maybe (CT); I have to wind down after work (ST)
Concise utterances which are easy for humans to understand, but missing information such as coreference, zero pronouns (22,31)	I mean I could try it (CT) Not a negative consequence for me (ST) I want to get every single second and minute out of it(CT)
Extremely short ($\leq$ 5) or long sentence ($\geq$ 40), caused by incorrect turn segementation. (21,23)	It is a good thing (ST) Frankly, I hate it (CT) Painful (CT)
Ambivalent speech, very hard to understand even for human. (7,4)	What if it does n't work I mean what if I can't do it (ST) But I can stop whenever I want(ST)

Table 4.11: Categorization of CT/ST confusions.The two numbers in the brackets are the count of errors for predicting CT as ST and vice versa. We exampled 100 examples for each case.

speech. Discovering the real attitude towards behavior change behind such utterances could be difficult, even for an expert therapist.

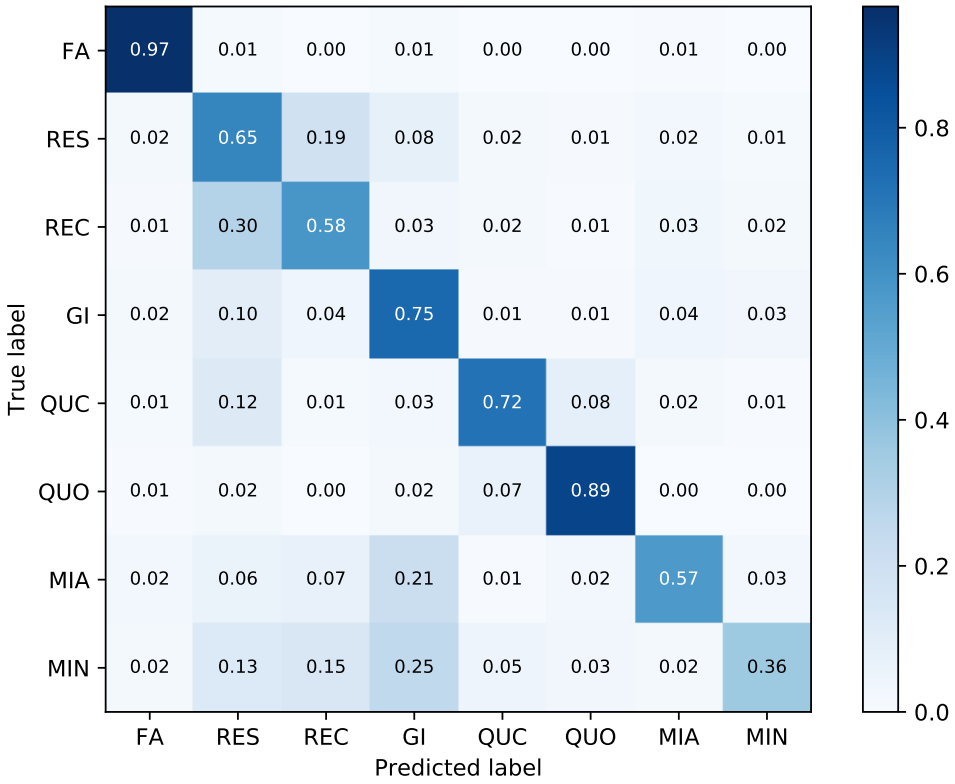

Figure 4.2: Confusion matrix for categorizing therapist codes, normalized by row.

Figures 4.1 and 4.2 show the label confusion matrices for the best categorization models. We will examine confusions that are not caused purely by a label being frequent. We observe a common confusion between the two reflection labels, REC and RES. Compared to the confusion matrix from Xiao et al. (2016), we see that our models show much-decreased confusion here. There are two reason for this confusion persisting. First, the reflections may require a much longer information horizon. We found that by increasing the window size to 16, the overall reflection results improved. Second, we need to capture richer meaning beyond surface word overlap for RES. We found that complex reflections usually add meaning or emphasis to previous client statements using devices such as analogies, metaphors, or similes rather than simply restating them.

Closed questions (QUC) and simple reflections (RES) are known to be a confusing set of labels. For example, an utterance like *Sounds like you're suffering?* may be both. Giving information (GI) is easily confused with many labels because they relate to providing information to clients, but with different attitudes. The MI adherent (MIA) and non-adherent (MIN) labels may also provide information, but with supportive or critical attitude that may be difficult to disentangle, given the limited number of examples.

4.5.2 How Context and Attention Help?

We evaluated various ablations of our best models to see how changing various design choices changes performance. We focused on the context window size and impact of different word level and sentence level attention mechanisms. Tables 4.13 and 4.14 summarize our results.

History Size. Increasing the history window size generally helps. The biggest improvements are for categorizing therapist codes (Table 4.14), especially for the RES and REC. However, increasing the window size beyond 8 does not help to categorize client codes (Table 4.13) or forecasting (Table 4.12).

Word-level Attention. Only the model $\mathcal{C}_T$ uses word-level attention. As shown in Table 4.14, when we remove the word-level attention from it, the overall performance drops by 3.4 points, while performances of RES and REC drop by 3.3 and 5 points respectively. Changing the attention to BiDAF decreases performance by about 2 points (still higher than the model without attention).

Ablation	Options	CT	ST	R@3	FA	RES	REC	GI	QUC	QUO	MIA	MIN
history size	1	17.2	15.1	66.4	59.4	12.6	9.0	44.6	16.3	14.8	11.9	4.1
	4	16.8	22.6	75.3	71.4	15.6	21.1	57.1	**29.3**	11.0	11.2	14.4
	8*	24.7	22.7	**77.0**	**72.8**	**20.8**	23.1	58.1	28.3	**17.7**	15.9	9.0
	16	23.9	20.7	76.5	71.2	13.7	24.1	**58.5**	25.9	9.7	16.2	12.7
word attention	GMGRU	14.0	**23.2**	75.7	71.7	14.2	23.0	57.5	26.5	8.0	15.4	11.6
	GMGRU$_{4h}$	19.1	22.9	76.3	71.3	12.1	23.3	58.1	24.5	12.6	11.7	14.0
sentence attention	− SELF$_{42}$	**24.9**	22.5	76.0	71.4	12.7	24.9	58.3	28.8	5.9	**17.4**	9.7
	\ ANCHOR$_{42}$	22.9	22.9	76.2	72.2	15.5	**24.6**	59.5	27.1	7.7	16.3	8.3
	+ GMGRU \ ANCHOR$_{42}$	6.8	23.4	76.9	70.8	8.0	24.5	58.3	24.6	10.6	14.9	**12.1**

Table 4.12: Ablation on forecasting task on both client and therapist code. $*$ row are results of our best forecasting model $\mathcal{F}_C$, and $\mathcal{F}_T$. \ means substitute anchor attention with self attention. +GMGRU ANCHOR$_{42}$ means using word-level attention and achor-based sentence-level attention together. Word-level attention shows no help for both client and therapist codes. While sentence-level attention helps more on therapist codes than on client codes. Multi-head self attention also achieves better performance than anchor-based attention in forecasting tasks.

Ablation	Options	macro	FN	CT	ST
history window size	0	51.6	87.6	39.2	32.0
	4	52.6	88.5	37.8	31.5
	8*	53.9	89.6	39.1	33.1
	16	52.0	89.6	39.1	33.1
word attention	+ GMGRU	52.6	89.5	37.1	31.1
	+ BiDAF	50.4	87.6	36.5	27.1
sentence attention	+ SELF$_{42}$	53.9	89.2	39.1	33.2
	+ ANCHOR$_{42}$	53.0	88.2	38.9	32.0

Table 4.13: Ablation study on categorizing client code. $*$ is our best model $\mathcal{C}_C$. All ablation is based on it. The symbol + means adding a component to it. The default window size is 8 for our ablation models in the word attention and sentence attention parts.

Sentence-level Attention. Removing sentence attention from the best models that have it decreases performance for the models $\mathcal{C}_T$ and $\mathcal{F}_T$ (in appendix). It makes little impact on the $\mathcal{F}_C$, however. Table 4.13 shows that neither attention helps categorizing clients codes.

4.5.3 How Focal Loss Helps on Label Imbalance?

We always use the same α for all weighted focal loss. Besides considering the label frequency, we also consider the performance gap between previous reported F_1. We choose

Ablation	Options	macro	RES	REC	MIN
history window size	0	62.6	51.6	49.4	24.2
	4	64.4	54.3	53.2	23.7
	8*	65.4	55.7	54.9	29.7
	16	**65.6**	55.4	**56.7**	26.7
word attention	- GMGRU	62.0	51.9	51.7	16.0
	\ BiDAF	63.5	54.2	51.3	22.6
sentence attention	- ANCHOR$_{42}$	64.9	56.0	54.4	21.8
	\ SELF$_{42}$	63.4	55.5	48.2	21.1

Table 4.14: Ablation study on categorizing therapist codes, $*$ is our proposed model $\mathcal{C}_T$. \ means substituting and $-$ means removing that component. Here, we only report the important REC, RES labels for guiding, and the MIN label for warning a therapist.

to balance weights α as {1.0,1.0,0.25} for CT,ST and FN respectively, and {0.5, 1.0, 1.0, 1.0, 0.75, 0.75,1.0,1.0} for FA, RES, REC, GI, QUC, QUO, MIA, MIN. As shown in Table 4.15, we report our ablation studies on cross-entropy loss, weighted cross-entropy loss, and focal loss. Besides the fixed weights, focal loss offers flexible hyperparameters to weight examples in different tasks. Experiments shows that except for the model $\mathcal{C}^T$, focal loss outperforms cross-entropy loss and weighted cross entropy.

Loss	**Client**			**Therapist**				
	F_1	CT	ST	F_1	RES	REC	MIA	MIN
$\mathcal{C}^{ce}$	47.0	28.4	22.0	60.9	54.3	53.8	53.7	4.8
$\mathcal{C}^{wce}$	53.5	39.2	32.0	65.4	55.7	54.9	56.6	29.7
$\mathcal{C}^{fl}$	53.9	39.1	33.1	65.4	55.7	54.9	56.6	29.7
$\mathcal{F}^{ce}$	42.1	17.7	18.5	26.8	3.3	20.8	16.3	8.3
$\mathcal{F}^{wce}$	43.1	20.6	23.3	30.7	17.9	25.0	17.7	10.9
$\mathcal{F}^{fl}$	44.2	24.7	22.7	31.1	19.5	24.7	15.2	12.8

Table 4.15: Abalation study of different loss function on categorizing and forecasting task. Based on our proposed model for our four settings, we compared our best model with crossentropy loss(ce), α balanced cross-entropy(wce) and focal loss. Here we only report the macro F_1 for rare labels and the overall macro F_1. $\gamma = 1$ is the best for both the model $\mathcal{C}_C$ and $\mathcal{F}_C$, while $\gamma = 0$ is the best for $\mathcal{C}_T$ and $\gamma = 3$ for $\mathcal{F}_T$. Worth to mention, when $\gamma = 0$, the focal loss degraded into α-balanced crossentropy, that first two rows are the same for therapist model.

Model	Embedding	macro	FN	CT	ST	macro	FA	RES	REC	GI	QUC	QUO	MIA	MIN
$\mathcal{C}$	ELMo	53.9	89.6	**39.1**	**33.1**	**65.4**	**95.0**	**55.7**	**54.9**	**74.2**	**74.8**	**82.6**	**56.6**	**29.7**
	$\text{ELMo}_{\text{psyc}}$	46.9	88.9	27.5	24.3	64.2	94.9	53.3	53.3	75.8	74.8	82.2	56.1	23.5
	Glove	50.6	**89.9**	33.4	28.6	62.2	94.6	53.7	54.2	70.3	70.0	79.1	54.7	20.9
	$\text{Glove}^{\text{pysc}}$	47.4	88.4	23.9	30.0	63.4	94.9	54.7	52.8	75.2	71.4	80.8	53.6	23.5
$\mathcal{F}$	ELMo	**44.3**	**85.2**	**24.7**	22.7	**31.1**	71.9	19.5	**24.7**	**59.2**	28.3	**17.7**	15.9	9.0
	$\text{ELMo}_{\text{psyc}}$	43.8	84.0	22.4	25.0	29.1	**73.5**	15.5	24.3	59.1	**29.1**	9.5	12.1	10.1
	Glove	42.7	83.9	21.0	23.1	30.0	72.8	**20.8**	23.7	58.2	26.2	14.5	14.5	9.6
	$\text{Glove}^{\text{pysc}}$	43.6	81.9	23.3	**25.7**	30.8	72.1	19.7	24.4	57.3	28.9	13.7	**17.8**	**23.5**

Table 4.16: Ablation study for our proposed model with embeddings trained on the psychotherapy corpus.

4.5.4 Can Domain Specific Glove and ELMo Help More?

We use the general psychotherapy corpus with 6.5M words (Alexander Street Press) to train the domain specific word embeddings $\textbf{Glove}_{psyc}$ with 50, 100, 300 dimension. Also, we trained ELMo with 1 highway connection and 256-dimensional output size to get $\textbf{ELMo}_{psyc}$. We found that ELMo 5.5B performs better than ELMo psyc in our experiments, and general Glove-300 is better than the $\textbf{Glove}_{psyc}$. Hence for main results of our models, we use $\textbf{ELMo}_{generic}$ by default. Please see more details in Table 4.16

4.5.5 Can We Suggest Empathetic Responses?

Our forecasting models are trained on regular MI sessions, according to the label distribution on Table 4.1, there are both MI adherent or non-adherent data. Hence, our models are trained to show how the therapist usually respond to a given statement.

To show whether our model can mimic *good* MI policies, we selected 35 MI sessions from our test set which were rated 5 or higher on a 7-point scale empathy or spirit. On these sessions, we still achieve a recall@3 of 76.9, suggesting that we can learn good MI policies by training on all therapy sessions. These results suggest that our models can help train new therapists who may be uncertain about how to respond to a client.

4.6 Conclusion

We addressed the question of providing real-time assistance to therapists and proposed the tasks of categorizing and forecasting MISC labels for an ongoing therapy session. By developing a modular family of neural networks for these tasks, we show that our models outperform several baselines by a large margin. Extensive analysis shows that our model

can decrease the label confusion compared to previous work, especially for reflections and rare labels, but also highlights directions for future work.

CHAPTER 5

REPRESENTING INTENT/SLOT CONCEPT WITH NATURAL LANGUAGE DESCRIPTION

From early frame-driven dialog system GUS (Bobrow et al., 1977) to virtual assistants (Alexa, Siri, and Google Assistant *et al.*), frame-based dialog state tracking has long been studied to meet various challenges. In particular, how to support an ever-increasing number of services and APIs spanning multiple domains has been a focal point in recent years, evidenced by multi-domain dialog modeling (Budzianowski et al., 2018; Byrne et al., 2019; Shah et al., 2018a) and transferable dialog state tracking to unseen intent/slots Mrkšić et al. (2017); Wu et al. (2019); Hosseini-Asl et al. (2020).

Recently, Rastogi et al. (2019) proposed a new paradigm called schema-guided dialog for transferable dialog state tracking by using natural language description to define a dynamic set of service schemata. As shown in Figure 5.1, the primary motivation is that these descriptions can offer effective knowledge sharing across different services, e.g., connecting semantically similar concepts across heterogeneous APIs, thus allowing a unified model to handle unseen services and APIs. With the publicly available schema-guided dialog dataset (SG-DST henceforward) as a testbed, they organized a state tracking shared task composed of four subtasks: intent classfication (INTENT), requested slot identification (REQ), categorical slot labeling (CAT), and noncategorical slot labeling (NONCAT) Rastogi et al. (2020). Many participants achieved promising performance by exploiting the schema description for dialog modeling, especially on unseen services.

Despite the novel approach and promising results, current schema-guided dialog state tracking task only evaluates on a single dataset with limited variation in schema definition. It is unknown how this paradigm generalizes to other datasets and other different styles of descriptions. In this paper, we focus our investigation on the study of three aspects in schema-guided dialog state tracking: (1) schema encoding model architectures (2) sup-

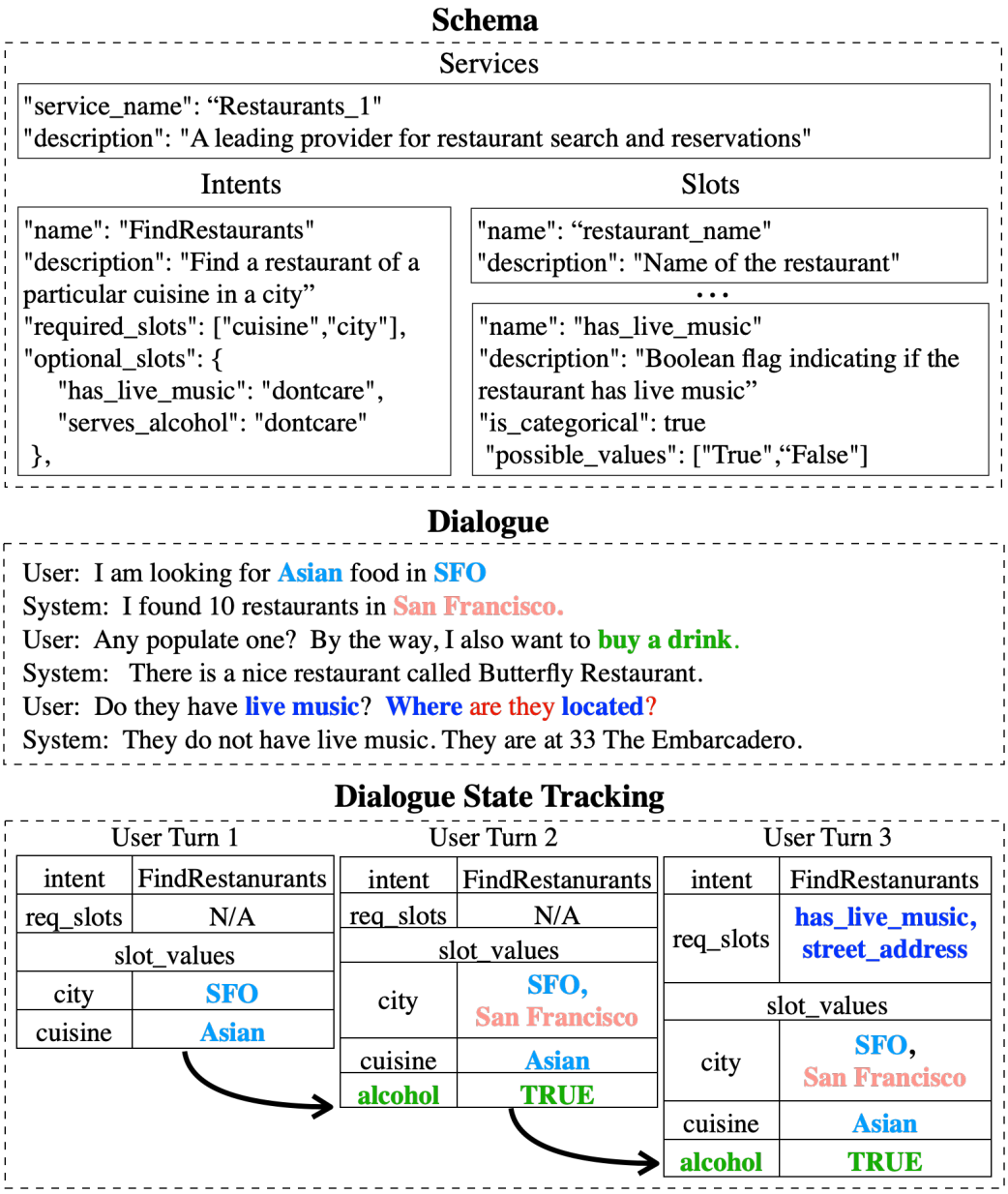

Figure 5.1: An example dialog from Restaurant_1 service, along with its service/intent/slot descriptions and dialog state representation.

plementary training on intermediate tasks (3) various styles for schema description. To make a more general discussion on the schema-guided dialog state tracking, we perform extensive empirical studies on both SG-DST and MULTIWOZ 2.2 datasets. In summary, our contributions include:

- A comparative study on schema encoding architectures, suggesting a partial-attention encoder for good balance between inference speed and accuracy.

- An experimental study of supplementary training on schema-guided dialog state tracking, via intermediate tasks including natural language inference and question answering.

- An in-depth analysis of different schema description styles on a new suite of benchmarking datasets with variations in schema description for both SG-DST and MULTIWOZ 2.2.

5.1 Schema-Guided Dialog State Tracking

A classic dialog state tracker predicts a dialog state frame at each user turn given the dialog history and predefined domain ontology. As shown in Figure 5.1, the key difference between schema-guided dialog state tracking and the classic paradigm is the newly added natural language descriptions. In this section, we first introduce the four subtasks and schema components in schema-guided dialog state tracking, then we outline the research questions in our paper.

Subtasks. As shown in Figure 5.1, the dialog state for each service consists of 3 parts: *active intent*, *requested slots*, *user goals (slot values)*. Without loss of generality, for both SG-DST and MULTIWOZ 2.2 datasets, we divide their slots into categorical and non-categorical slots by following previous study on dual-strategies Zhang et al. (2019a). Thus to fill the dialog state frame for each user turn, we solve four ***independent*** subtasks: intent classification (INTENT), requested slot identification (REQ), categorical slot labeling (CAT), and non-categorical slot labeling (NONCAT). All subtasks require matching the current dialog history with candidate schema descriptions for multiple times.

Schema Components. Figure 5.1 shows three main schema components: service, intent, slot. For each intent, the schema also describes *optional* or *required* slots for it. For each slot, there are flags indicating whether it is categorical or not. *Categorical* means there is a set of predefined candidate values (Boolean, numeric or text). For instance, *has_live_music* in Figure 5.1 is a categorical slot with Boolean values. *Non-categorical*, on the other hand, means the slot values are filled from the string spans in the dialog history.

New Questions. These added schema descriptions pose the following three new questions. We discuss each of them in the following sections.

Q1. How should dialogue and schema be encoded? §5.4
Q2. How do different supplementary trainings impact each subtask? §5.5
Q3. How do different description styles impact the state tracking performance? §5.6

5.2 Related Work

Our work is related to three lines of research: multi-sentence encoding, multi-domain and transferable dialog state tracking. However, our focus is on the comparative study of different encoder architectures, supplementary training, and schema description style variation. Thus we adopt existing strategies from multi-domain dialog state tracking.

Multi-Sentence Encoder Strategies. Similar to the recent study on encoders for response selection and article search tasks Humeau et al. (2019), we also conduct our comparative study on the two typical architectures *Cross-Encoder* Bordes et al. (2014); Lowe et al. (2015a) and *Dual-Encoder* Wu et al. (2017); Yang et al. (2018). However, they only focus on sentence-level matching tasks. All subtasks in our case require sentence-level matching between dialog context and each schema, while the non-categorical slot filling task also needs to produce a sequence of token-level representation for span detection. Hence, we study multi-sentence encoding for both sentence-level and token-level tasks. Moreover, to share the schema encoding across subtasks and turns, we also introduce a simple *Fusion-Encoder* by caching schema token embeddings in §5.4.1, which improves efficiency without sacrificing much accuracy.

Multi-domain Dialog State Tracking. Recent research on multi-domain dialog system have been largely driven by the release of large-scale multi-domain dialog datasets, such as MultiWOZ Budzianowski et al. (2018), M2M Shah et al. (2018a), accompanied by studies on key issues such as in/cross-domain carry-over Kim et al. (2019). In this paper, our goal is to understanding the design choice for schema descriptions in dialog state tracking. Thus we simply follow the in-domain cross-over strategies used in **TRADE** Wu et al. (2019). Additionally, explicit cross-domain carryover Naik et al. (2018) is difficult to generalize to new services and unknown carryover links. We use longer dialog history to inform the model on the dialog in the previous service. This simplified strategy does impact our model performance negatively in comparison to a well-designed dialog state

Datasets	Splits	Dialog	Domains	Services	Zero-shot Domains	Zero-shot Services	Function Overlapp	Collecting Method
SG-DST	Train	16142	16	26	-	-	Across/Within-domain	M2M
	Dev	2482	16	17	1	8		
	Test	4201	18	21	3	11		
MULTIWOZ 2.2	Train	9617	3	3	-	-	Across-domain	H2H
	Dev	2455	5	5	2	2		
	Test	2969	8	8	5	5		

Table 5.1: Summary of characteristics of SG-DST MULTIWOZ 2.2 datasets, in domain diversity, function overlap, data collecting methods

tracking model on seen domains. However, it helps reduce the complexity of matching extra slot descriptions for cross-service carryover. We leave the further discussion for future work.

Transferable Dialog State Tracking. Another line of research focuses on how to build a transferable dialog system that is easily scalable to newly added intents and slots. This covers diverse topics including e.g., resolving lexical/morphological variabilities by symbolic de-lexicalization-based methods Henderson et al. (2014); Williams et al. (2016), neural belief tracking Mrkšić et al. (2017), generative dialog state tracking Peng et al. (2020); Hosseini-Asl et al. (2020), modeling DST as a question answering task Zhang et al. (2019a); Lee et al. (2019); Gao et al. (2020, 2019). Our work is similar with the last class. However, we further investigate whether the DST can benefit from NLP tasks other than question answering. Furthermore, without rich description for the service/intent/slot in the schema, previous works mainly focus on simple format on question answering scenarios, such as domain-slot-type compounded names (e.g., "*restaurant-food*"), or simple question template "*What is the value for* **slot i***?*". We incorporate different description styles into a comparative discussion on §5.6.1.

5.3 Datasets and Model Setup

To the best of our knowledge, at the time of our study, SG-DST and MULTIWOZ 2.2 are the only two publicly available corpus for schema-guided dialog study. We choose both of them for our study. In this section, we first introduce these two representative datasets, then we discuss the generalizibility in domain diversity, function overlapping, data collecting methods.

5.3.1 Schema-Guided Dialog Dataset

SG-DST dataset [1] is especially designed as a test-bed for schema-guided dialog, which contains well-designed heterogeneous APIs with overlapping functionalities between services Rastogi et al. (2019). In DSTC8 Rastogi et al. (2020), SG-DST was introduced as the standard benchmark dataset for schema-guided dialog research. SG-DST covers 20 domains, 88 intents, 365 slots.[2] However, previous research are mainly conducted based on this single dataset and the provided single description style. In this paper, we further extended this dataset with other benchmarking description styles as shown in §**??**, and then we perform both homogenous and hetergenous evalution on it.

5.3.2 Remixed MultiWOZ 2.2 Dataset

To eliminate potential bias from the above single SG-DST dataset, we further add MULTIWOZ 2.2 Zang et al. (2020) to our study.

Statistic on MultiWOZ 2.2 Remix. To evaluate performance on seen/unseen services with MultiWOZ, we remix the MULTIWOZ 2.2 dataset to include as seen services dialogs related to *restaurant*, *attraction* and *train* during training, and eliminate slots from other domains/services from training split. For dev, we add two new domains *hotel* and *taxi* as unseen services. For test, we add all remaining domains as unseen, including those that have minimum overlap with seen services, such as *hospital*, *police*, *bus*. The statistics are as shown in Table 5.2

Domain	#dialogs/#turns					
	train		dev		test	
restaurant	3900	37953	458	6979	451	7104
attraction	2716	28632	405	6198	400	6290
train	3001	29646	481	5897	491	6150
hotel	0	0	737	8509	718	7911
taxi	0	0	374	2692	364	2659
hospital	0	0	0	0	287	766
police	0	0	0	0	252	475
bus	0	0	0	0	6	132

Table 5.2: The total number of dialogs and turns related to each domain in train, dev and test split of MultiWOZ

[1] `https://github.com/google-research-datasets/dstc8-schema-guided-dialogue`

[2] Please refer to the original paper for more details.

Among various extended versions for MultiWOZ dataset (2.0-2.3, Budzianowski et al., 2018; Eric et al., 2020; Zang et al., 2020; Han et al., 2020) , besides rectifying the annotation errors, MULTIWOZ 2.2 also introduced the schema-guided annotations, which covers 8 domains, 19 intents, 36 slots. To evaluate performance on seen/unseen services with MultiWOZ, we remix the MULTIWOZ 2.2 dataset to include as seen services dialogs related to *restaurant*, *attraction* and *train* during training, and eliminate slots from other domains/services from training split. For dev, we add two new domains *hotel* and *taxi* as unseen services. For test, we add all remaining domains as unseen, including those that have minimum overlap with seen services, such as *hospital*, *police*, *bus*. The statistics of data splits are shown in **??**. Note that this data split is different from the previous work on zero-shot MultiWOZ DST which takes a leave-one-out approach in Wu et al. (2019). By remixing the data in the way described above, we can evaluate the zero-shot performance on MultiWOZ in a way largely compatible with SG-DST.

5.3.3 Discussion on Datasets

First, the two datasets cover diverse domains. MULTIWOZ 2.2 covers various possible dialogue scenarios ranging from requesting basic information about attractions through booking a hotel room or travelling between cities. While SG-DST covers more domains, such as 'Payments', 'Calender', 'DoctorServices' and so on.

Second, they include different levels of overlapping functionalities. SG-DST allows frequent function overlapping between multiple services, within the same domain (e.g. BookOneWayTicket v.s. BookRoundTripTicket), or across different domains (BusTicket v.s. TrainTicket). However, the overlapping in MULTIWOZ 2.2 only exists across different domains, e.g., 'destination', 'leaveat' slots for Taxi and Bus services, 'pricerange', 'bookday' for Restaurant and Hotel services.

Third, they are collected by two different approaches which are commonly used in dialog collecting. SG-DST is firstly collected by machine-to-machine self-play (M2M, Shah et al., 2018b) with dialog flows as seeds, then paraphrased by crowd-workers. While MULTIWOZ 2.2 are human-to-human dialogs (H2H, Kelley, 1984), which are collected with the Wizard-of-Oz approach.

We summarize the above discussion in Table 5.1. We believe that results derived from

these two representative datasets can guide future research in schema guided dialog.

5.3.4 Experiment Setup

All models are based on BERT-base-cased model with 2 V100 GPUs (with 16GB GPU RAM each). We train each models for maximum 10 epoch, by using AdamW to schedule the learning rate with a warm-up portion of 0.1. During training, we evaluate checkpoints per 3000 steps on dev splits, and select the model with best performance on dev split on all seen and unseen services. In our experiments, our model achieves the best performance on around 2-4 epochs on INTENT, REQ. and CAT, while NONCAT needs 5-8 epochs to get the best performance. For all subtasks, as we model all of them as sentence pair encoding during training, we use batch size as 16 for each GPU, and gradient accumulate for 8 steps, in total 256 batch size on 2 GPUs.

5.4 Dialog & Schema Representation and Inference (Q1)

In this section, we focus on the model architecture for matching dialog history with schema descriptions using pretrained BERT Devlin et al. (2019) [3]. To support four subtasks, we first extend *Dual-Encoder* and *Cross-Encoder* to support both sentence-level matching and token-level prediction. Then we propose an additional *Fusion-Encoder* strategy to get faster inference without sacrificing much accuracy. We summarize different architectures in Figure 5.2. Then we show the classification head and results for each subtask.

5.4.1 Encoder Architectures

Dual-Encoder. It consists of two separate BERTs to encode dialog history and schema description respectively, as Figure 5.2 (a). We follow the setting in the official baseline provided by DSTC8 Track4 Rastogi et al. (2020). We first use a fixed BERT to encode the schema description once and cached the encoded schema $\mathbf{CLS}_S$. Then for sentence-level representation, we concatenate dialog history representation $\mathbf{CLS}_D$ and candidate schema representation $\mathbf{CLS}_S$ as the whole sentence-level representation for the pair, denoted as $\mathbf{CLS}^{DE}$. For token-level representation, we concatenate the candidate schema

[3] We use BERT-base-cased for all main experiments. Other pretrained language models can be easily adapted to our study

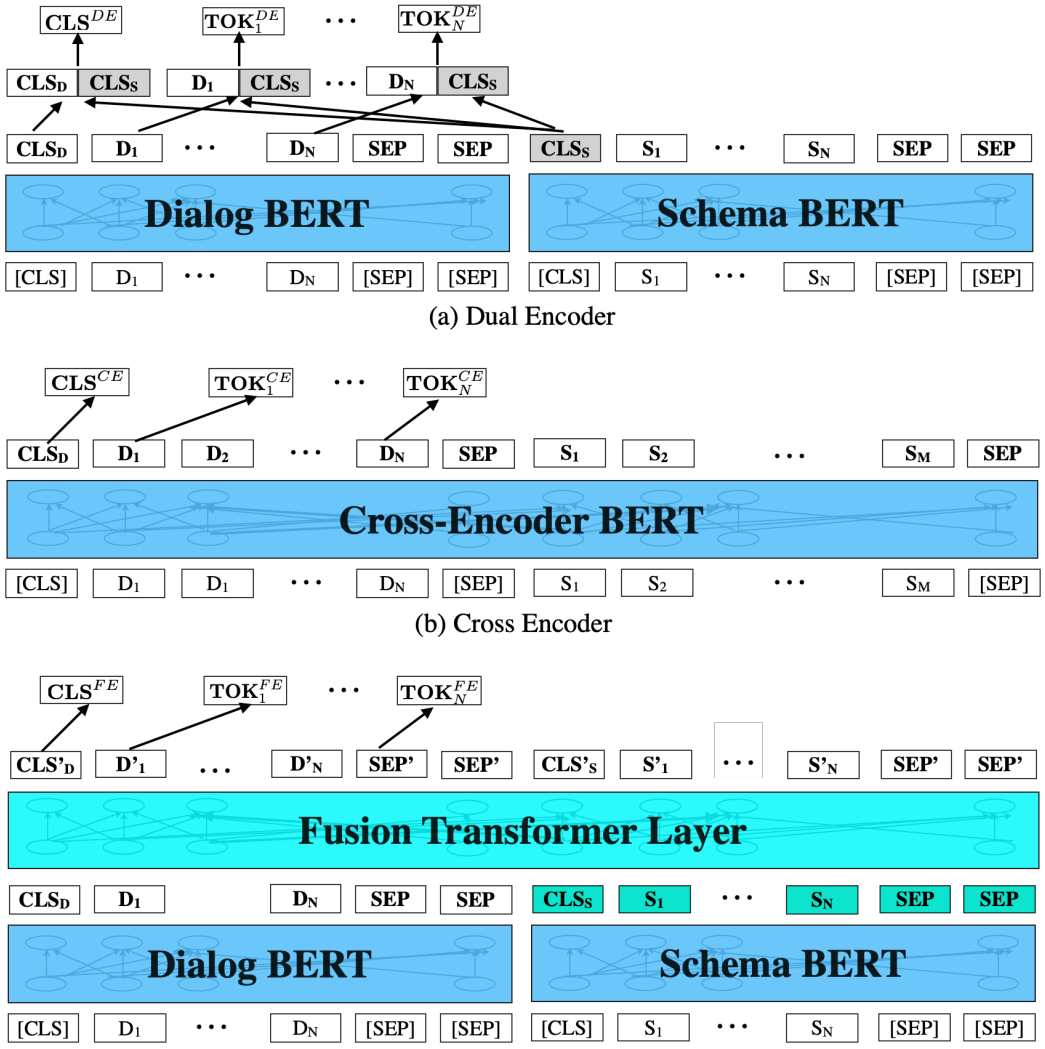

Figure 5.2: Dual-Encoder, Cross-Encoder and Fusion Encoder, shaded block will be cached during training

$\textbf{CLS}_S$ with each token embedding in the dialog history, denoted as $\textbf{TOK}^{DE}$.[4] Because the candidate schema embeddings are encoded independently from the dialog context, they can be pre-computed once and cached for fast inference.

Cross-Encoder. Another popular architecture as Figure 5.2 (b) is *Cross-Encoder*, which concatenates the dialog and schema as a single input, and encodes jointly with a single self-attentive encoder spanning over the two segments. When using BERT to encode

[4] A schema-aware dialog token embedding can also be computed by attention or other method for span-based detection tasks Humeau et al. (2019); Noroozi et al. (2020)

the concatenated sentence pair, it performs full (cross) self-attention in every transformer layers, thus offer rich interaction between the dialog and schema. BERT naturally produces a summarized representation with [CLS] embedding $\mathbf{CLS}^{CE}$ and each schema-attended dialog token embeddings $\mathbf{TOK}^{CE}$. Since the dialog and schema encoding always depend on each other, it requires recomputing dialog and schema encoding for multiple times, thus much slower in inference.

Fusion-Encoder. In Figure 5.2 (c), similar to *Dual-Encoder*, *Fusion-Encoder* also encodes the schema independently with a fixed BERT and finetuning another BERT for dialog encoding. However, instead of caching a single [CLS] vector for schema representation, it caches **all token representation** for the schema including the [CLS] token. What's more, to integrate the sequences dialog token representation with schema token representation, an extra stack of transformer layers are added on top to allow token-level fusion via self-attention, similar to *Cross-Encoder*. The top transformer layers will produce embeddings for each token $\mathbf{TOK}^{FE}$ including a schema-attended $\mathbf{CLS}^{FE}$ of the input [CLS] from the dialog history. With cached schema token-level representations, it can efficiently produce schema-aware sentence- and token-level representation for each dialog-schema pairs.

5.4.2 Models for Factorized Subtasks

All the above 3 encoders will produce both sentence- and token-level representations for a given sentence pair. In this section, we abstract them as two representations **CLS** and **TOK**, and present the universal classification heads to make decisions for each subtask.

Active Intent. To decide the intent for current dialog turn, we match current dialog history D with each intent descriptions $I_0 ... I_k$. For each dialog-intent pair (D, I_k), we project the final sentence-level **CLS** representation to a single number $P_{I_k}^{active}$ with a linear layer follows a sigmoid function. We predict *"NONE"* if the $P_{I_k}^{active}$ of all intents are less than a threshold 0.5, which means no intent is active. Otherwise, we predict the intent with largest $P_{I_k}^{active}$. We predict the intent for each turn independently without considering the prediction on previous turns.

Requested Slot. As in Figure 5.1, mulitple requested slots can exist in a single turn. We use the same strategy as in active intent prediction to predict a number P_{req}^{active}. However,

to support the multiple requested slots prediction. We predict all the requested slots with $P_{req}^{active} > 0.5$.

Categorical Slot. Categorical slots have a set of candidate values. We cannot predict unseen values via n-way classification. Instead, we do binary classification on each candidate value. Besides, rather than directly matching with values, we also need to check that whether the corresponding slot has been activated. For *Cross-Encoder* and *Fusion-Encoder*, we use typical two-stage state tracking to incrementally build the state: **Step** 1. Using **CLS** to predict the slot status as *none*, *dontcare* or *active*. When the status is *active*, we use the predicted slot value; Otherwise, it will be assigned to *dontcare* meaning no user preference for this slot, or *none* meaning no value update for the slot in current turn; **Step** 2. If Step 1 is *active*, we match the dialog history with each value and select the most related value by ranking. We train on cross entropy loss. Two-stage strategy is efficient for *Dual-Encoder* and *Fusion-Encoder*, where cached schema can be reused, and get efficiently ranked globally in a single batch. However, it is not scalable for *Cross-Encoder*, especially for large number of candidate values in MultiWOZ dataset. Hence, during training, we only use a binary cross-entropy for each single value and postpone the ranking only to the inference time.

Noncategorical Slot. The slot status prediction for noncategorical slot use the same two-stage strategy. Besides that, we use the token representation of dialog history **TOK** to compute two softmax scores f_{start}^{i} and f_{end}^{i} for each token i, to represent the score of predicting the token as start and end position respectively. Finally, we find the valid span with maximum sum of the start and end scores.

5.4.3 Experiments on Encoder Comparison

To fairly compare all three models, we follow the same schema input setting as in Table 5.3. We trained separate models for SG-DST and the remixed MultiWOZ datasets for all the experiments in our papers[5]. Because there are very few intent and requested slots in MULTIWOZ 2.2 dataset, we ignore the intent and requested slots tasks for MULTIWOZ 2.2 in our paper.

Results. As shown in Table 5.4, *Cross-Encoder* performs the best over all subtasks. Our

[5]Appendix **??** shows the detailed experiment setup

Intent	service description, intent description
Req	service description, slot description
Cat	slot description, cat value
NonCat	service description, slot description

Table 5.3: Schema description input used for different tasks to compare *Dual-Encoder*, *Cross-Encoder*, and *Fusion-Encoder*. In the appendix B.1, we also studies other compositions of description input. We found that service description will not help for INTENT, REQ and CAT tasks, while the impact on NONCAT task also varies from SG-DST and MULTIWOZ 2.2 dataset.

	SG-DST					MULTIWOZ 2.2		
Method/Task	Acc	F1	Joint Acc			Joint Acc		
	Intent	Req	Cat	NonCat	All	Cat	NonCat	All
Seen Services								
Dual-Encoder	94.51	99.62	87.92	47.77	43.20	79.20	79.34	65.64
Fusion-Encoder	94.90	**99.69**	88.94	48.78	58.52	81.37	80.58	67.43
Cross-Encoder	**95.55**	99.59	**93.68**	**91.85**	**87.58**	**85.99**	**81.02**	**71.93**
Unseen Services								
Dual-Encoder	89.73	95.20	42.44	31.62	19.51	56.92	50.82	31.83
Fusion-Encoder	90.47	95.95	48.79	35.91	22.85	57.01	52.23	33.64
Cross-Encoder	**93.84**	**98.26**	**71.55**	**74.13**	**54.54**	**59.85**	**59.62**	**38.46**

Table 5.4: Test set results on SG-DST and MULTIWOZ 2.2. The *Dual-Encoder* model is a re-implementation of official DSTC8 baseline from Rastogi et al. (2019). Other models are trained with the architecture described in our paper.

Fusion-Encoder with partial attention outperforms the *Dual-Encoder* by a large margin, epsecially on categorical and noncategorical slots predictions. Additionally, on seen services, we found that *Dual-Encoder* and *Fusion-Encoder* can perform as good as *Cross-Encoder* on INTENT and REQ tasks. However, they cannot generalize well on unseen services as *Cross-Encoder*.

Inference Speed. To test the inference speed, we conduct all the experiments with a maximum affordable batch size to fully exploit 2 V100 GPUs (with 16GB GPU RAM each). During training, we log the inference time of each evaluation on dev set. Both *Dual-Encoder* and *Fusion-Encoder* can do joint inference across 4 subtasks to obtain an integral dialog state for a dialog turn example. *Dual-Encoder* achieves the highest inference speed of **603.35** examples per GPU second, because the encoding for dialog and schema are fully separated. A dialog only needed to be encoded for once during the inference of a dialog

	SG-DST											
	intent			req			cat			noncat		
	all	seen	unseen	all	seen	unseen	all	seen	unseen	all	seen	unseen
$\Delta_{\textbf{SNLI}}$	**+0.51**	+0.02	**+0.68**	-0.19	+0.38	-0.38	-1.63	-2.87	-1.23	-4.7	-0.1	-6.25
$\Delta_{\textbf{SQuAD}}$	-1.81	-0.17	-1.32	-0.25	-0.01	-0.33	-2.87	-3.02	-5.17	**+1.99**	-1.79	**+3.25**

Table 5.5: Relative performance improvement of different supplementary training on SG-DST dataset

	MULTIWOZ 2.2					
	cat			noncat		
	all	seen	unseen	all	seen	unseen
$\Delta_{\textbf{SNLI}}$	+2.05	+0.6	–0.7	**+3.64**	+1.05	**+4.84**
$\Delta_{\textbf{SQuAD}}$	+0.04	-0.71	+0.41	**+1.93**	-2.21	**+4.27**

Table 5.6: Relative performance improvement of different supplementary training on MULTIWOZ 2.2 dataset

state example while the schema are precomputed once. However, for *Cross-Encoder*, to predict a dialog state for a single turn, it need to encode more than 300 sentence pairs in a batch, thus only processes **4.75** examples per GPU second. *Fusion-Encoder* performs one time encoding on dialog history, but it needs to jointly encode the same amount of dialog-schema pair ws *Cross-Encoder*, instead, however, with a two-layer transformer encoder. Overall it achieves **10.54** examples per GPU second, which is **2.2x** faster than *Cross-Encoder*. With regarding to the accuracy in Table 5.4, *Fusion-Encoder* performs much better than *Dual-Encoder*, especially on unseen services.

5.5 Supplementary Training (Q2)

Besides the pretrain-fintune framework used in §**??**, Phang et al. (2018) propose to add a supplementary training phase on an intermediate task after the pretraining, but before finetuning on target task. It shows significant improvement on the target tasks. Moreover, large amount pretrained and finetuned transformer-based models are publicly accessible, and well-organized in model hubs for sharing, training and testing[6]. Given the new task of schema-guided dialog state tracking, in this section, we study our four subtasks with

[6]e.g., Huggingface(`https://huggingface.co/models`) and ParlAL(`https://parl.ai/docs/zoo.html`), etc.

different intermediate tasks for supplementary training.

5.5.1 Intermediate Tasks

As described in § 5.4.2, all our 4 subtasks take a pair of dialog and schema description as input, and predict with the summerized sentence-pair **CLS** representation. While NON-CAT also requires span-based detection such as question answering. Hence, they share the similar problem structure with the following sentence-pair encoding tasks.

Natural Language Inference. Given a hypothesis/premise sentence pair, natural language inference is a task to determine whether a hypothesis is entailed, contradicted or neutral given that premise. **Question Answering.** Given a passage/question pairs, the task is to extract the span-based answer in the passage.

Hence, when finetuning BERT on our subtaks, instead of directly using the originally pretrained BERT, we use the BERT finetuned on the above two tasks for further finetuning. Due to better peformance of *Cross-Encoder* in §**??**, we directly use the finetuned *Cross-Encoder* version of BERT models on SNLI and SQuAD2.0 dataset from Huggingface model hub. We add extra speaker tokens [user:] and [system:] into the vocabulary for encoding the multi-turn dialog histories.

5.5.2 Results on Supplementary Training

Table **??** shows the performances gain when finetuning 4 subtasks based on models with the above SNLI and SQuAD2.0 supplementary training.

We mainly find that SNLI helps on INTENT task, SQuAD2 mainly helps on NON-CAT task, while neither of them helps much on CAT task. Recently, Namazifar et al. (2020) also found that when modeling dialog understanding as question answering task, it can benefit from a supplementary training on SQuAD2 dataset, especially on few-shot scenarios, which is a similar findings as our NONCAT task. Result difference on REQ task is minor, because it is a relatively easy task, adding any supplementary training did n't help much. Moreover, for CAT task, the sequence 2 of the input pair is the slot description with a categorical slot value, thus the meaning overlapping between the full dialog history and the slot/value is much smaller than SNLI tasks. On the other side, **CLS** token in SQuAD BERT is finetuned for null predictions via start and end token classifers, which is different

from the the single CLS classifer in CAT task.

5.6 Impact of Description Styles (Q3)

Previous work on schema-guided dialog Rastogi et al. (2020) are only based on the provided descriptions in SG-DST dataset. Recent work on modeling dialog state tracking as reading comprehension Gao et al. (2019) only formulate the descriptions as simple question format with existing intent/slot names, it is unknown how it performs when compared to other description styles. Moreover, they only conduct homogeneous evaluation where training and test data share the same description style. In this section, We also investigate how a model trained on one description style will perform on other different styles, especially in a scenario where chat-bot developers may design their own descriptions. We first introduce different styles of descriptions in our study, and then we train models on each description style and evaluate on tests with corresponding homogeneous and heterogeneous styles of descriptions. Given the best performance of *Cross-Encoder* shown in the previous section and its popularity in DSTC8 challenges, we adopt it as our model architecture in this section.

5.6.1 Benchmarking Styles

For each intent/slot, we describe their functionalities by the following different descriptions styles:

IDENTIFER. This is the least informative case of name-based description: we only use meaningless intent/slot identifiers, e.g. Intent_1, Slot_2. It means we don't use description from any schema component. We want to investigate how a simple identifier-based description performs in schema-guided dialog modeling, and the performance lower-bound on transferring to unseen services.

NAMEONLY. Using the original intent/slot names in SG-DST and MULTIWOZ 2.2 dataset as descriptions, to show whether name is enough for schema-guided dialog modeling.

Q-NAME. This is corresponding to previous work by Gao et al. (2019). For each intent/slot, it generate a question to inquiry about the intent and slot value of the dialog. For each slot, it simply follows the template '*What is the value for* **slot i***?*'. Besides that, our work also extend the intent description by following the template "*Is the user intending to* **intent j** ".

ORIG. The original descriptions in SG-DST and MULTIWOZ 2.2 dataset.

Q-ORIG. Different from the Q-NAME, firstly it is based on the original descriptions; secondly, rather than always use the "what is" template to inquiry the intent/slot value, We add "what", "which", "how many" or "when" depending on the entity type required for the slot. Same as Q-NAME, we just add prefixes as "Is the user intending to…" in front of the original description. In a sum, this description is just adding question format to original description. The motivation of this description is to see whether the question format is helpful or not for schema-guided dialog modeling.

To test the model robustness, we also create two paraphrased versions NAME-PARA and ORIG-PARA for NAMEONLY and ORIG respectively. We first use nematus Sennrich et al. (2017) to automatically paraphrase the description with back translation, from English to Chinese and then translate back, then we manually check the paraphrase to retain the main meaning. Appendix **??** shows examples for different styles of schema descriptions.

5.6.2 Results on Description Styles

Unlike the composition used in Table 5.3, we don't use the service description to avoid its impact. For each style, we train separate models on 4 subtasks, then we evaluate them on different target styles. First, Table 5.7 summarizes the performance for homogeneous evaluation, while Table 5.8 shows how the question style description can benefit from SQuAD2 finetuning. Then we also conduct heterogeneous evaluation on the other styles[7] as shown in Table 5.9.

5.6.2.1 Homogeneous Evaluation

Is name-based description enough? As shown in Table 5.7, IDENTIFER is the worst case of using name description, its extremely bad performance indicates name-based description can be very unstable. However, we found that simple meaningful name-based description actually can perform the best in INTENT and REQ task, and they perform worse on CAT and NONCAT tasks comparing to the bottom two *rich* descriptions. [8] After careful

[7] We don't consider the meaningless IDENTIFER style due to its bad performance

[8] Only exception happens in CAT on MULTIWOZ 2.2. When creating MULTIWOZ 2.2 Zang et al. (2020), the slots with less than 50 different slot values are classified as categorical slots, which leads to inconsistencies.

Style\Task	SG-DST				MULTIWOZ 2.2	
	Intent	Req	Cat	NonCat	Cat	NonCat
IDENTIFER	61.16	91.48	62.47	30.19	34.25	52.28
NAMEONLY	**94.24**	98.84	74.01	75.63	53.72	56.18
Q-NAME	93.31	**98.86**	74.36	74.86	**54.19**	56.17
ORIG	93.01	98.55	74.51	75.76	52.19	57.20
Q-ORIG	93.42	98.51	**76.64**	**76.60**	53.61	**57.80**

Table 5.7: Homogeneous evaluation results of different description style on SG-DST dataset and MULTIWOZ 2.2 datasets. The middle horizontal line separate the two name-based descriptions and two rich descriptions in our settings. All numbers in the table are mixed performance including both seen and unseen services.

analysis on the intents in SG-DST datasets, we found that most services only contains two kinds of intents, an information retrieval intent with a name prefix "Find-", "Get-", "Search-"; another transaction intent like "Add-", "Reserve-" or "Buy-". Interestingly, we found that all the intent names in the original schema-guided dataset strictly follows an action-object template with a composition of words without abbreviation, such as "FindEvents", "BuyEventTickets". This simple name template is good enough to describe the core functionality of an intent in SG-DST dataset. [9] Additionally, REQ is a relaitively simper task, requesting information are related to specifial attributes, such as "has_live_music", "has_wifi", where keywords co-occured in the slot name and in the user utterance, hence rich explanation cannot help further. On the other side, *rich* descriptions are more necessary for CAT and NONCAT task. Because in many cases, slot names are too simple to represent the functionalities behind it, for example, slot name "passengers" cannot fully represent the meaning "number of passengers in the ticket booking".

Does question format help? As shown in Table 5.7, when comparing row Q-ORIG v.s. ORIG, we found extra question format can improve the performance on CAT and NONCAT task on both SG-DST and MULTIWOZ 2.2 datasets, but not for INTENT and REQ tasks. We believe that question format helps the model to focus more on specific entities in the dialog history. However, when adding a simple question pattern to NAMEONLY, comparing row

We put detailed discuss about MULTIWOZ 2.2 in the supplementary material

[9] This action-object template has also been found efficient for open domain intent induction task(e.g., Vedula et al., 2020, OPINE).

Style/Dataset	SG-DST			MULTIWOZ 2.2		
	all	seen	unseen	all	seen	unseen
ORIG	+1.99	-1.79	**+3.25**	+1.93	-2.21	**+4.27**
Q-ORIG	+6.13	-2.01	**+8.84**	+1.06	-1.28	**+3.06**
NAMEONLY	-0.45	-1.49	-0.11	+1.75	+0.58	**+1.77**
Q-NAME	+0.05	-2.98	**+1.04**	-0.04	-0.32	**+1.25**

Table 5.8: Performance changes when using BERT finetuned on SQuAD2 dataset to further finetuning on our NONCAT task.

Q-NAME and NAMEONLY, there is no consistent improvement on both of the two datasets. Further more, we are curious about whether BERT finetuned on SQuAD2 (SQuAD2-BERT) can further help on the question format. Because NONCAT are similar with span-based question answering, we focus on NONCAT here. Table 5.8 shows that, after applying the supplementary training on SQuAD2 (§**??**), almost all models get improved on unseen splits however slightly dropped on seen services. Moreover, comparing to Q-NAME, Q-ORIG is more similar to the natural questions in the SQuAD2, we obverse that Q-ORIG gains more than Q-NAME from pretrained model on SQuAD2.

5.6.2.2 Heterogeneous

In this subsection, we first simulate a scenario when there is no recommended description style for the future unseen services. Hence, unseen services can follow any description style in our case. We average the evaluation performance on three other descriptions and summarized in Table 5.9. The Δ column shows the performance change compared to the homogeneous performance. It is not surprising that almost all models perform worse on heterogeneous styles than on homogeneous styles due to different distribution between training and evaluation. The bold number shows the best average performance on heterogeneous evaluation for each subtask. The trends are similar with the analysis in homogeneous evaluation 5.6.2.1, the name-based descriptions perform better than other rich descriptions on intent classification tasks. While on other tasks, the ORIG description performs more robust, especially on NONCAT task.

Furthermore, we consider another scenario where fixed description convention such as NAMEONLY and ORIG are suggested to developers, they must obey the basic style convention but still can freely use their own words, such as abbreviation, synonyms,

adding extra modifiers. We train each model on NAMEONLY and ORIG, then evaluate on the corresponding paraphrased version respectively. In the last two rows of Table 5.9, the column 'para' shows performance on paraphrased schema, while Δ shows the performance change compared to the homogeneous evaluation. ORIG still performs more robust than NAMEONLY when schema descriptions get paraphrased on unseen services.

Style\Task	SG-DST							
	Intent(Acc)		Req(F1)		Cat(Joint Acc)		NonCat(Joint Acc)	
	mean	Δ	mean	Δ	mean	Δ	mean	Δ
NAMEONLY	82.47	-11.47	96.92	-1.64	61.37	-5.54	56.53	-14.68
Q-NAME	**93.27**	+0.58	**97.88**	-0.76	68.55	+2.63	62.92	-6.30
ORIG	79.47	-12.70	97.42	-0.74	**68.58**	-0.3	**66.72**	-3.11
Q-ORIG	84.57	-8.24	96.70	-1.45	68.40	-2.89	56.17	-15.00
	para	Δ	para	Δ	para	Δ	para	Δ
NAMEONLY	**92.22**	-1.74	97.69	-0.87	67.39	-0.7	67.17	-4.04
ORIG	91.54	**-0.63**	**98.42**	**+0.26**	**71.74**	**+2.86**	**67.68**	**-2.16**

Table 5.9: Results on unseen service with heterogeneous description styles on SG-DST dataset. More results and qualitative analysis are in the appendix B.3

5.7 Conclusion

In this paper, we studied three questions on schema-guided dialog state tracking: encoder architectures, impact of supplementary training, and effective schema description styles. The main findings are as follows:

By caching the token embedding instead of the single CLS embedding, a simple partial-attention *Fusion-Encoder* can achieve much better performance than *Dual-Encoder*, while still infers two times faster than *Cross-Encoder*. We quantified the gain via supplementary training on two intermediate tasks. By carefully choosing representative description styles according to recent works, we are the first of doing both homogeneous/heterogeneous evaluations for different description style in schema-guided dialog. The results show that simple name-based description performs well on INTENT and REQ tasks, while NON-CAT tasks benefits from richer styles of descriptions. All tasks suffer from inconsistencies in description style between training and test, though to varying degrees.

Our study are mainly conducted on two datasets: SG-DST and MULTIWOZ 2.2, while the speed-accuracy balance of encoder architectures and the findings in supplementary training are expected to be dataset-agnostic, because they depend more on the nature of the subtasks than the datasets. Based on our proposed benchmarking descriptions suite, the

homogeneous and heterogeneous evaluation has shed the light on the robustness of cross-style schema-guided dialog modeling, we believe our study will provide useful insights for future research.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

In this chapter, we summarize our contributions and highlight some open problems, suggesting possible directions for future research. Fine-grained summerization and discussion about remaining issues about each topic can be found at the end of the corresponding chapters.

6.1 Claims and Research Contribution Revisited

Thesis Statement. Our claim is that by designing ***Structural Inductive Bias*** and ***Natural Language as Inductive Biases***, models with naive independent factorization can achieve strong performance on predicting the structures of natural language across multiple tasks and domains.

Contribution. The main contributions of this thesis are as follows:

1. We proposed a unified parsing framework to support both **explicit lexical-anchoring** (including DELPH-IN MRS Bi-lexical Dependencies (DM, Ivanova et al., 2012a) and Prague Semantic Dependencies (PSD, Hajic et al., 2012; Miyao et al., 2014)), and **implicit lexical anchoring** (AMR). For the **phrasal-anchoring** Universal Conceptual Cognitive Annotation (UCCA, Abend and Rappoport, 2013b) and Task-oriented Dialog Parsing (TOP, Gupta et al., 2018), according to their similarity to constituency tree structure, we extrapolate the existed algorithmic inductive bias on tree structure prediction and Cost-augmented CKY inferece to the new UCCA and TOP parsing tasks. Powered by the above structural inductive biases, over 16 teams, our parsing system (Cao et al., 2019b) ***ranked 1st on AMR, 6th in DM, 7th in PSD, 5th on UCCA parsing, and outperform serveral baseline models on TOP parsing.***

2. We address the problem of providing real-time guidance to therapists with a dialogue observer. It decompose the dialog structure analysis with two independent

factorization tasks: (1) categorizes therapist and client MI behavioral codes and, (2) forecasts codes for upcoming utterances to help guide the conversation and potentially alert the therapist. For both tasks, I studied a hierarchical gated recurrent unit (HGRU) with the ***word-level attention*** and ***sentence-level attention*** to distinguish different importance of words and sentences to our prediction (Cao et al., 2019a). Our experiments demonstrate that our models can outperform several baselines for both tasks. We also report the results of a careful analysis that reveals the impact of the various network design tradeoffs for modeling therapy dialogue.

3. Natural language can be leveraged as inductive biases to describe the functions of the intent/slot labels in task-oriented dialogue. We are among the first to use large pretrained language models on description-based dialog state tracking, we offer detailed comparative studies how to transfer inductive biases to new domains and APIs with overlapping functions and task structures, including encoding strategies, supplementary pretraining, homogenuous and heterogeneous evalutions.

6.2 Future Work

6.2.1 For Other Factorization

In this thesis, we mainly focus on the independent factorization, which ignores the interdependence between output parts. Our experiments shows contexturalized representation capturing the interdependence within the input parts can still offer discriminative features to independently predict each local output parts. However, there are other factorizations we can extend our work on, such as auto-regressive factorization, or arbitrary high order factorizations. In those cases, the output parts will either depend on the previously predicted output, or have other high order dependecies. Take the auto-regressive factorization as an example, we consider a more broad-coverage construction of output y as sequential decisions as shown in Equation 6.1.

$$E(x, y) = \sum_{j=0}^{M} E(y_j | x, y_{<j}) \tag{6.1}$$

In autoregressive factorization, as shown in Figure 6.1, for every step, a new decision y_j will depend on the input x and previous decisions $y_{<j}$. In this case, the main challenge of the model is to learn the representation of x and previous sequential decisions $y_{<j}$, then

make a decision y_j based on them. Especially inspired by distributed representation of the natural language, we also study the distributed representation of the output structures $y_{<j}$, and leverage them to guide constrained structured prediction.

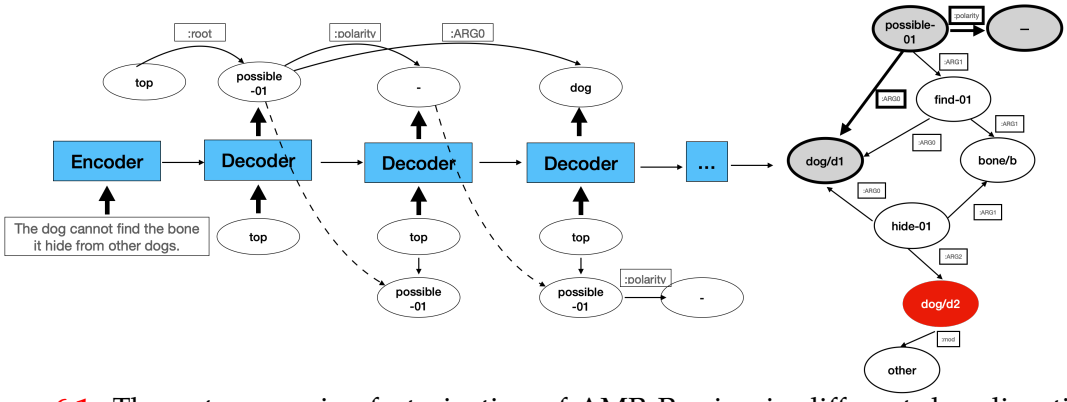

Figure 6.1: The autoregressive factorization of AMR Parsing in different decoding time step

6.2.2 Apply to Other Symbolic Representations

As shown in **??**, above structural inductive bias in lexical, phrasal, sentential anchoring can be easily extended to other linguistic structured prediction tasks, such as coreference resolution, semantic role labeling, where the main task strutures has been studied in our broad-coverage meaning representation parsing.

For application-specific symbolic representation, besides the single sentence representation in (TOP, Gupta et al., 2018), we also can extend our structured prediction models into session-based conversational representation such as session-based TOP Aghajanyan et al. (2020), (TreeDST, Cheng et al., 2020), and (Dataflow, Andreas et al., 2020). Beyond conversational analysis, in the future, I plan to exploit this structured analysis on symbolic representation to offer rigorous document analysis, easier knowledge organization, programmable reasoning, which are potentially helpful for structured social analysis such as mental health, cyberbullying, thus offering structural suggestions to guide human behavior.

6.2.3 Future Work on Contexualized Representation

The strong power of contexturalized representation learning make out independent factorization works still gold under our inductive biases. However, there are still many

challenges on contexturalized representation learning.

Extreme Long Context. First, we need to resolve extrem long text encoding problem. Our current models on psychotherapy dialogue and schema-guided dialog only consider 8 to 16 utterance as the dialogue history window. However, we have more than 500 utterance in a single therapy session. Further more, a psychotherapy treatment may last for months and years, which involves multiple dialogue sessions. The long context problem also exists in other domain, such as scientific document analysis, and threaded conversations in social medias.

Contextualized Representation Beyond Text. multimodality

6.2.4 Other Biases in Other Formalism

Besides the above structural inductive bias on compositionality and hierarchial structure, in the future, I will continue the study on how to represent other inductive biases in other ways.

Declaritive Constraints and other latent Models. In the future, we also inject other structural interdependence/constraints with declaritive tools, such as integer linear programming, probabilistic neural logic rules. Further more, following the line of latent variable models, we plan to study more ways to relax structural inductive biases as continuous and differentiable latent variables in the end2end deep learning.

Approximate Bias. With limited observations and resources (time, memory, energy), our human intelligence of generalizing to new environments makes us efficiently learn when interacting with the world and other human beings. This efficiency largely depends on many ***inductive biases*** and ***approximate biases*** from human intelligence Gershman (2021), which are potentially helpful for machine intelligence. We also plan to working approximate biases for reasoning. Especially, for deep learning, we plan to design approximate inference methods for deep structured prediction.

Causality. Current factorization and markov random field based formalism only can capture the correlation between different variables. In the future, we will also extend the formalism to bayesian networks and intervention based models.

6.2.5 Learning and Transfering the Inductive Biases

Learn the Inductive Biases via Self-supervision. Inspired by self-supervised pretraining in ELMo and BERT Devlin et al. (2019), our VLDB'2022 paper Paul* et al. (2021) extend a contrastive-learning method to learn the representation for tree-structured database query plans. With a large amount of raw database query plans, we calculate the graph similarity metric *Smatch* Cai and Knight (2013) to represent the degree of overlap between a pair of plans.[1] After we get the *Smatch* scores s_{ij} of each plan-pair $< p_i, p_j >$, this can easily form a large dataset with *Smatch* score as the contrastive self-supervision. In our experiments on the downstream applications, we show that the structure encoder pre-trained from this task can be easily finetuned for a new task or domain.

Transfering Inductive Bias via Supplementary Training. Learning to learn is an essential inductive bias in human intelligence Harlow (1949), human can generalize experience learned from similar tasks to learn new tasks. Nowadays many datasets and pre-trained models are publicly accessible, besides transferring the inductive bias from initial language models, I also studied how to transfer the inductive biases learned from the well-studied tasks to new tasks. My NAACL'2021 work on schema-guided dialogue state tracking Cao and Zhang (2021) proposed to add a supplementary pretraining phase on an intermediate task between the pretraining-finetuning framework. Given a brand new task like schema-guided dialogue state tracking, we show that supplementary pretraining on intermediate tasks with similar problem structures will offer efficient distributional inductive biases. More specifically, we found that inductive bias learned in sentence-pair matching (via Natural Language Inference on SNLI) helps with intent classification tasks, and span-based retrieval task structure (via Question Answering on SQuAD2) helps on the non-categorical slot labeling task.

Besides passively receiving fixed training data to learn, intelligent systems can improve themselves by actively discovering more supervisions. My future work will ground the explorations on two sub areas: (a) learning on how to retrieve and integrate the experience with new observations. This is similar to how humans learn from the experience and other

[1] The Smatch score ([0,1]) between two tree-structure plans can be computed by graph matching optimization algorithm, such as Integer Linear Programming (ILP) or Hill-climbing methods.

existing tools, including searching over similar datasets or pre-trained models. (b) learning on make the model self-evolve, such as via the feedback after deployment.

APPENDIX A

MORE ABOUT MISC CODES

A.1 Different Clustering Strategies for MISC

Code	Count	Description	Examples
MIA	3869	Group of MI Adherent codes : Affirm(AF); Reframe(RF); Emphasize Control(EC); Support(SU); Filler(FI); Advise with permission(ADP); Structure(ST); Raise concern with permission(RCP)	"You've accomplished a difficult task." (AF) "ItâĂŹs your decision whether you quit or not" (EC) "That must have been difficult." (SU) "Nice weather today!" (FI) "Is it OK if I suggested something?" (ADP) "Let's go to the next topic" (ST) "Frankly, it worries me." (RCP)
MIN	1019	Group of MI Non-adherent codes: Confront(CO); Direct(DI); Advise without permission(ADW); Warn(WA); Raise concern without permission(RCW)	"You hurt the baby's health for cigarettes?" (CO) "You need to xxx." (DI) "You ask them not to drink at your house." (ADW) "You will die if you don't stop smoking." (WA) "You may use it again with your friends." (RCW)

Table A.1: Label distribution, description and exmaples for MIA and MIN

MISC-28. The original MISC description of Miller et al. (2003) included 28 labels (9 client, 19 therapist).

MISC-8. Due to data scarcity and label confusion, some labels were merged into a coarser set. Can et al. (2015) retain 6 original labels FA, GI, QUC, QUO, REC, RES, and merge remaining 13 rare labels into a single COU label, they merge all 9 client codes into a single CLI label.

MISC-15. Instead, Tanana et al. (2016) merge only 8 of rare labels in therapist codes into a OTHER label and they cluster client codes into 3 labels according to the valence of changing, sustaining or being neutral on the addictive behavior Atkins et al. (2014).

MISC-11. Then Xiao et al. (2016) combine and improve above two clustering strategies by splitting the all 13 rare labels according to whether the code represents MI-adherent(MIA)

and MI-nonadherent (MIN) We show more details about the original labels in MIA and MIN in Table A.1

APPENDIX B

MORE ANALYSIS ON SCHEMA-GUIDED DIALOGUE STATE TRACKING

B.1 Composition of Descriptions

For each subtask, the key description element must be included, e.g., intent description for intent task, and value for categorical slot tasks. However, besides the description about intents and slots, we also have service description and the names for intents and slots. What other information will help? and what kind of composition will offer better performance. In this section, we will mainly answer the questions with the following experiments on compostion of descriptions.

B.1.1 Composition Settings

To show how each component helps schema-guided dialog state tracking, we incrementally add richer schema component one by one.

ID. This is the least informative case: we only use meaningless intent/slot identifiers, e.g. Intent_4, Slot_2. It means we don't use description from any schema component. We want to investigate how a simple identifier-based description performs in schema-guided dialog modeling, and the performance lower-bound on transferring to unseen services.

I/S Desc. Only using the original intent/slot description of intent/slot in SG-DST and MULTIWOZ 2.2 dataset for corresponding tasks.

Service + I/S Desc. Adding a service description to the above original description. Service description summarize the functionalities of the whole service, hence may offer extra background information for intent and slots. For categorical slot value detection, we simply add the value after each of the above composition.

Model\Task	SG-DST				MultiWOZ	
	Intent	Req	Cat	NonCat	Cat	NonCat
Seen Service						
Identifer	92.76	99.70	87.86	88.38	58.46	77.29
I/S Desc	**95.35**	**99.74**	92.10	**93.52**	**85.84**	**83.67**
Service + I/S Desc	95.28	**99.74**	**93.19**	92.34	85.07	80.56
Unseen Service						
Identifier	50.63	88.74	54.34	10.77	53.05	56.18
I/S Desc	**92.17**	**98.16**	**68.88**	69.84	56.49	**61.39**
Service + I/S Desc	86.95	97.99	67.08	**71.30**	**60.58**	59.63

Table B.1: Models using different composition of schema, results on test set of SG-DST and our remixed MULTIWOZ 2.2

		sgd												multiwoz					
		intent			req			cat			noncat			cat			noncat		
snli	uncased	93.31	95.4	92.62	98.62	99.34	98.37	75.66	93.39	69.98	80.38	90.93	76.87	51.77	84.93	59.40	56.47	82.39	61.62
	snli	93.82	95.42	93.3	98.43	99.72	97.99	74.03	90.52	68.75	75.68	90.83	70.62	53.82	85.53	58.70	60.11	83.44	66.46
	Δ_{SNLI}	**+0.51**	+0.02	**+0.68**	-0.19	+0.38	-0.38	-1.63	-2.87	-1.23	-4.7	-0.1	-6.25	+2.05	+0.6	–0.7	**3.64**	+1.05	**+4.84**
squad	cased	93.01	95.51	92.2	98.59	99.59	98.26	74.51	92.1	71.23	75.76	93.52	69.84	52.19	85.74	56.49	57.2	83.67	61.39
	squad	91.2	95.34	90.88	98.34	99.58	97.93	71.64	89.08	66.06	77.75	91.73	73.09	52.23	85.03	56.90	59.13	81.46	65.66
	Δ_{SQuAD}	-1.81	-0.17	-1.32	-0.25	-0.01	-0.33	-2.87	-3.02	-5.17	**1.99**	-1.79	**3.25**	+0.04	-0.71	+0.41	**1.93**	-2.21	**+4.27**

Table B.2: Results of different supplementary training on SG-DST and MULTIWOZ 2.2 dataset

B.1.2 Results on Description Compositions

Table B.1 shows the results of using different description compositions. First, there are consistent findings across datasets and subtasks: (1) using meaningless identifier as intent/slot description shows the worse performance on all tasks of both datasets, and can not generalize well to unseen services. (2) using intent/slot descriptions can largely boost the performance, especially on unseen services.

However, the impact of service description varies by tasks. For example, it largely hurts performance on intent classification task, but does not impact requested slot and categorical slot tasks. According to manual analysis of SG-DST and MULTIWOZ 2.2 dataset, we found that service description consists of the main functions of the service, especially the meaning of the supported intents. Hence, using service description for intent causes confusion between the intent description information and other supported intents. Moreover, in categorical slot value prediction task, the most important information is the slot description and value. When adding extra information from service description, it improves marginally on seen service while not generalizing well on unseen services, which indicates the model learns artifacts that are not general useful for unseen services.

Finally, on non-categorical slot tasks, the impact of service description may also varies on datasets. On SG-DST, there are 16 domains and more than 30 services, the rich background context from service description contains both domain and service-specific information, which seems to help both seen and unseen services. However, on MULTIWOZ 2.2, it hurts the performance on seen service *restaurant* the most, while improving the performance on the unseen service *hotel* by 4 points. In this case, it works like a regularizer rather than a definitive clues. Because in MULTIWOZ 2.2, there are only 8 domains, and one service per domain, thus service descriptions just contain domain related information without much extra information, it will not help the model to detect the span for the slot.

B.2 More Results of Supplementary Training

Table B.2 shows the detailed performance when using different intermediate tasks as supplementary training. For SNLI tasks, as the pretrained model is uncased model (textattack/ bert-base-uncased-snli), hence, we first train different models with BERT-base-uncased, then compare the performance with SNLI pretrained model. For SQuAD2, we use deepset/bert-base-cased-SQuAD2 model, hence, we compare it all cased model. To fairly compare with our original *Cross-Encoder*, we add extra speaker tokens [user:] and [system:] for encoding the multi-turn dialog histories.

B.3 Homogeneous and Hetergenuous Evaluation on Different Styles

B.3.1 Examples for Different Description Styles

Table B.3 shows examples for different styles of schema descriptions.

B.3.2 More details on SQuAD2 Results on Different Styles

For homogeneous evaluation, Table B.4 shows the detailed performance when we apply SQuAD2-finetuned BERT on our models.

B.3.3 More Results On Homogeneous and Heterogeneous Evalution

We list the detailed results for our evaluation across different styles. We use *italic* to show the homogeneous evaluation, where the results are shown in the diagonal of each table, and we underline the best homogeneous results in the diagonal. We use **bold** to

show the best heterogeneous performance and the best performance gap in the last two columns

Intent. The results on SG-DST dataset are shown in Table B.5. Because there are very few intents in MULTIWOZ 2.2 dataset, we don't conduct intent classification on MULTIWOZ 2.2. All performance get dropped when evaluating on heterogeneous descriptions styles. For both heterogeneous and homogeneous evaluation, adding rich description on intent classification tasks seems not bring much benefits than simply using the named-based description. As the discussion in §5.6.2.1, we believe the name template is good enough to describe the core functionality of an intent in SG-DST dataset.

Requested Slot. Table B.6 shows the results on SG-DST dataset for the requested slots subtask. We ignore the requested slots in MULTIWOZ 2.2 dataset due to its sparsity. Overall, the requested slot subtask are relatively easy, performances on heterogeneous styles still drops but not much. For both heterogeneous and homogeneous evaluation, the performance are not sensible to rich description.

Categorical Slot. The results on SG-DST and MULTIWOZ 2.2 dataset are shown in Table B.7. When creating MULTIWOZ 2.2 Zang et al. (2020), the slots with less than 50 different slot values are classified as categorical slots. We noticed that this leads inconsistent results with SG-DST dataset. It is hard to draw a consistent conclusion on the two datasets. According to the definition, we believe SG-DST are more suitable for categorical slot sub-tasks, we can further verify our guess when more datasets are created for the research of schema-guided dialog in the future.

Non-categorical Slot. We conduct non-categorical slot identification sub-tasks on both SG-DST and MULTIWOZ 2.2 dataset. The results are shown in Table B.8. Overall, the rich description performs better on both homogeneous and heterogeneous evaluations.

B.3.4 Qualitative Analysis On Heterogeneous Evaluation

We conduct qualitative analysis on heterogeneous evaluation on named-based description. Table B.9 shows how paraphrasing the named-based description impact on the categorical and non-categorical slot prediction tasks.

The first 3 rows at the top are showing the cases of adding modifiers to the name.

When the added extra modifiers are keywords in other slots, e.g. "attraction" are the keywords also used in "attraction_name". The first shows "attraction_location" may wrongly predicted as "attraction_name". It seems the model does not understand the compound nouns well, and they seems just pay attention to each key words "attraction" and "movie" here.

The 3 rows in the middle are showing the cases of using synonyms. Changing "to" to "target", and changing "movie" to "film" will cause extra confusion, which shows the model may fail to the synonyms.

The last 4 rows at the bottom is showing using abbreviations. Changing "number" to "num" will not impact the model, while changing "subtitle" to "sub" may let the model miss the key meaning of subtitle. The performance drop in the later case may be due to the misuse of the "sub" prefix, in English, it usually means "secondary, less important, parts". We also found the "orig" and "dest" abbreviations may also understand well by the model. The above abbreviations seems reasonable paraphrases people will use for naming, while the are not understood well in the given context. Hence, in the design of schema-guided dialog, if using named-based description, we should be careful for about abbreviations used in the naming.

style	Intent Description	Slot Description
IDENTIFER	intent_1	slot_4
NAMEONLY	CheckBalance	account_type
Q-NAME	Is the user intending to CheckBalance?	What is the value of acctount_type ?
ORIG	Check the amount of money in a user's bank account	The account type of the user
Q-ORIG	Does the user want to check the amount of money in the bank account ?	What is the account type of the user ?
NAME-PARA	CheckAccountBalance	user_account_type
ORIG-PARA	Check the balance of the user's bank account	Type of the user account

Table B.3: Different extensions of schema descriptions

Style/Dataset	SG-DST			MULTIWOZ 2.2		
	all	seen	unseen	all	seen	unseen
ORIG	75.76	93.52	69.84	57.2	83.67	61.39
	77.75	91.73	73.09	59.13	81.46	65.66
	+1.99	-1.79	**+3.25**	+1.93	-2.21	**+4.27**
Q-ORIG	76.60	92.86	71.18	57.80	82.45	62.45
	82.73	90.85	80.02	58.86	81.17	65.51
	+6.13	-2.01	**+8.84**	+1.06	-1.28	**+3.06**
NAMEONLY	75.63	88.90	71.21	56.18	81.68	61.30
	75.18	87.41	71.10	57.93	82.26	63.07
	-0.45	-1.49	**–0.11**	+1.75	+0.58	**+1.77**
Q-NAME	74.86	91.78	69.22	56.17	81.19	60.47
	74.91	88.8	70.26	56.13	80.87	61.72
	+0.05	-2.98	**+1.04**	-0.04	-0.32	**+1.25**

Table B.4: Results on different description style on SG-DST and MULTIWOZ 2.2 dataset, when performing SQuAD2 supplementary training

Style	NAMEONLY	Q-NAME	ORIG	Q-ORIG	mean	Δ
NAMEONLY	*93.94*	78.27	93.18	75.95	82.47	-11.47
Q-NAME	93.18	*92.69*	93.26	93.36	**93.27**	**+0.58**
ORIG	81.57	66.42	*92.17*	90.43	79.47	-12.70
Q-ORIG	81.48	79.04	93.19	*92.81*	84.57	-8.24

Table B.5: Accuracy of intent classification subtask with different description styles on unseen services. Train the model on SG-DST dataset for each description in each row, then evaluating on 4 different descriptions styles. The mean are average performance of the remaining 3 descriptions styles. The Δ means the performance gap between the mean and the homogeneous performance

Style	NAMEONLY	Q-NAME	ORIG	Q-ORIG	mean	Δ
NAMEONLY	*98.56*	96.01	97.2	97.54	96.92	-1.64
Q-NAME	98.37	*98.64*	97.8	97.48	**97.88**	-0.76
ORIG	97.95	95.78	*98.16*	98.52	97.42	**-0.74**
Q-ORIG	97.24	95.85	97.00	*98.15*	96.70	-1.45

Table B.6: F1 Score of requested slot classification subtask with different description styles on unseen services. We train the model on SG-DST dataset for the description style in each row, then evaluate on 4 different descriptions styles. The mean are average performance of the remaining 3 descriptions styles. The Δ means the performance gap between the mean and the homogeneous performance

Style	NAMEONLY	Q-NAME	ORIG	Q-ORIG	mean	Δ
SG-DST						
NAMEONLY	*68.09*	58.41	63.49	62.21	61.37	-6.72
Q-NAME	69.01	*68.29*	68.53	68.12	68.55	**+0.26**
ORIG	70.19	65.91	*68.88*	69.64	**68.58**	-0.30
Q-ORIG	69.98	65.97	69.26	*71.29*	68.40	-2.89
MULTIWOZ 2.2						
NAMEONLY	*59.24*	59.32	59.12	59.29	59.24	0.00
Q-NAME	58.64	*59.74*	58.49	59.43	58.85	-0.89
ORIG	59.26	59.91	*56.49*	58.97	**59.38**	**+2.89**
Q-ORIG	60.00	60.70	51.18	*58.95*	57.29	-1.66

Table B.7: Joint accuracy of categorical slot Subtask with different description styles on unseen services. Train the model on SG-DST and MULTIWOZ 2.2 datasets respectively for each description style in each row, then evaluate on all 4 descriptions styles. The mean are the average performance of the remaining 3 descriptions styles. The Δ means the performance gap between the mean and the homogeneous performance

Style	NAMEONLY	Q-NAME	ORIG	Q-ORIG	mean	Δ
SG-DST						
NAMEONLY	*71.21*	49.85	59.8	59.95	56.53	-14.68
Q-NAME	66.32	*69.22*	61.67	60.77	62.92	-6.30
ORIG	78.73	51.57	*69.84*	69.87	**66.72**	**-3.12**
Q-ORIG	62.6	36.44	69.49	*71.18*	56.18	-15.00
MULTIWOZ 2.2						
NAMEONLY	*61.30*	57.88	61.51	64.05	61.15	-0.15
Q-NAME	60.62	*60.47*	60.6	62.58	61.27	+0.80
ORIG	61.77	65.4	*61.39*	62.4	**63.19**	**+1.80**
Q-ORIG	61.29	60.6	62.46	*62.45*	61.45	-1.00

Table B.8: Joint accuracy of non-categorical slot Subtask with different description styles on unseen services. We train the model on SG-DST and MULTIWOZ 2.2 datasets respectively for the description style in each row, then evaluate on all 4 different descriptions styles. The mean are the average performance of the remaining 3 descriptions styles. The Δ means the performance gap between the mean and the homogeneous performance

Service Name	Original Name	Paraphrased Name	Extra impact by the paraphrased name
Travel_1	location	attraction_location	Confused with other "attraction" prefixed slots, e.g. attraction _name
Movies_1	genre	movie_genre	Confused with movie_name
Movies_1	price	ticket_price, total_price	No impact
Bueses_3	to_city	target_city	The synonyms "target" is not understood well by model, confused with from_city
Movies_1	movie_name	film_name	The synonyms "film" is not understood well, getting wrong with theather_name
Hotels_2	where_to	house_loc	Improved by specific "house" keywords
Flights_4	origin_airport	orig_city_airport	More frequently predicted to slot "destination_airport"
Flights_4	destination_airport	dest_city_airport	More frequently predicted to slot "origin_airport"
Media_3	subtitle_language	sub_lang	Missing keyword "subtitle" make the slot inactive
Flights_4	number_of_tickets	num_of_tickets	No impact

Table B.9: We analyze the confusion matrix of above slots before and after using the paraphrased name. We summarize the extra impact for using each paraphrased name.

	MULTIWOZ 2.2			
	cat		noncat	
	seen	unseen	seen	unseen
Δ_{SNLI}	+0.6	–0.7	+1.05	**+4.84**
Δ_{SQuAD}	-0.71	+0.41	-2.21	**+4.27**

Table B.10: Relative performance improvement of different supplementary training on SG-DST and MULTIWOZ 2.2 dataset

Style/Dataset	SG-DST		MULTIWOZ 2.2	
	seen	unseen	seen	unseen
ORIG	-1.79	**+3.25**	-2.21	**+4.27**
Q-ORIG	-2.01	**+8.84**	-1.28	**+3.06**
NAMEONLY	-1.49	-0.11	+0.58	**+1.77**
Q-NAME	-2.98	**+1.04**	-0.32	**+1.25**

Table B.11: Performance changes when using BERT finetuned on SQuAD2 dataset to further finetuning on our NONCAT task.

Style\Task	SG-DST			
	Intent(Acc)	Req(F1)	Cat(Joint Acc)	NonCat(Joint Acc)
	Δ	Δ	Δ	Δ
NAMEONLY	-11.47	-1.64	-5.54	-14.68
Q-NAME	+0.58	-0.76	+2.63	-6.30
ORIG	-12.70	-0.74	-0.3	-3.11
Q-ORIG	-8.24	-1.45	-2.89	-15.00
	Δ	Δ	Δ	Δ
NAMEONLY	-1.74	-0.87	-0.7	-4.04
ORIG	**-0.63**	**+0.26**	**+2.86**	**-2.16**

Table B.12: Results on unseen service with heterogeneous description styles on SG-DST dataset. More results and qualitative analysis are in the appendix B.3

REFERENCES

Omri Abend and Ari Rappoport. 2013a. Universal Conceptual Cognitive Annotation (UCCA). In *Proceedings of the 51th Meeting of the Association for Computational Linguistics*, pages 228 – 238, Sofia, Bulgaria.

Omri Abend and Ari Rappoport. 2013b. Universal conceptual cognitive annotation (ucca). In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 228–238.

Omri Abend and Ari Rappoport. 2017. The state of the art in semantic representation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 77–89.

Armen Aghajanyan, Jean Maillard, Akshat Shrivastava, Keith Diedrick, Michael Haeger, Haoran Li, Yashar Mehdad, Veselin Stoyanov, Anuj Kumar, Mike Lewis, et al. 2020. Conversational semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5026–5035.

Jacob Andreas, John Bufe, David Burkett, Charles Chen, Josh Clausman, Jean Crawford, Kate Crim, Jordan DeLoach, Leah Dorner, Jason Eisner, et al. 2020. Task-oriented dialogue as dataflow synthesis. *Transactions of the Association for Computational Linguistics*, 8:556–571.

Yoav Artzi, Kenton Lee, and Luke Zettlemoyer. 2015. Broad-coverage CCG Semantic Parsing with AMR. *EMNLP*.

David C Atkins, Mark Steyvers, Zac E Imel, and Padhraic Smyth. 2014. Scaling up the evaluation of psychotherapy: evaluating motivational interviewing fidelity via statistical text classification. *Implementation Science*, 9(1):49.

Stephen H Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. 2017. Hinge-loss markov random fields and probabilistic soft logic. *Journal of Machine Learning Research*, 18:1–67.

John S Baer, Elizabeth A Wells, David B Rosengren, Bryan Hartzler, Blair Beadnell, and Chris Dunn. 2009. Agency context and tailored training in technology transfer: A pilot evaluation of motivational interviewing training for community counselors. *Journal of substance abuse treatment*, 37(2):191–202.

Collin F Baker, Charles J Fillmore, and John B Lowe. 1998. The berkeley framenet project. In *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 86–90.

James K Baker. 1979. Trainable grammars for speech recognition. *The Journal of the Acoustical Society of America*, 65(S1):S132–S132.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013a. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178 – 186, Sofia, Bulgaria.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013b. Abstract Meaning Representation for Sembanking. *LAW@ACL*.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. 2014. DonâĂŹt count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 1.

Jonathan Baxter. 2000. A model of inductive bias learning. *Journal of artificial intelligence research*, 12:149–198.

Kenneth R Beesley and Lauri Karttunen. 2003. Finite-state morphology: Xerox tools and techniques. *CSLI, Stanford*.

David Belanger and Andrew McCallum. 2016. Structured prediction energy networks. In *International Conference on Machine Learning*, pages 983–992. PMLR.

Emily M. Bender, Dan Flickinger, Stephan Oepen, Woodley Packard, and Ann Copestake. 2015. Layers of interpretation. On grammar and compositionality. In *Proceedings of the 11th International Conference on Computational Semantics*, pages 239 – 249, London, UK.

Jonathan Berant, Vivek Srikumar, Pei-Chun Chen, Abby Vander Linden, Brittany Harding, Brad Huang, Peter Clark, and Christopher D Manning. 2014. Modeling biological processes for reading comprehension. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1499–1510.

Elie Bienenstock, Stuart Geman, and Daniel Potter. 1996. Compositionality, mdl priors, and object recognition. In *NIPS*.

John Binder, Kevin Murphy, and Stuart Russell. 1997. Space-efficient inference in dynamic probabilistic networks. *Bclr*, 1:t1.

Daniel G Bobrow, Ronald M Kaplan, Martin Kay, Donald A Norman, Henry Thompson, and Terry Winograd. 1977. Gus, a frame-driven dialog system. *Artificial intelligence*, 8(2):155–173.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. 2021. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*.

Antoine Bordes, Jason Weston, and Nicolas Usunier. 2014. Open question answering with weakly supervised embedding models. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 165–180. Springer.

Samuel R. Bowman, Jon Gauthier, Abhinav Rastogi, Raghav Gupta, Christopher D. Manning, and Christopher Potts. 2016. A fast unified model for parsing and sentence understanding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1466–1477, Berlin, Germany. Association for Computational Linguistics.

Peter F Brown, Vincent J Della Pietra, Stephen A Della Pietra, and Robert L Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational linguistics*, 19(2):263–311.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 2018. Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026.

Harry Bunt, Jan Alexandersson, Jean Carletta, Jae-Woong Choe, Alex Chengyu Fang, Koiti Hasida, Kiyong Lee, Volha Petukhova, Andrei Popescu-Belis, Laurent Romary, et al. 2010. Towards an iso standard for dialogue act annotation. In *Seventh conference on International Language Resources and Evaluation (LREC'10)*.

Brian L Burke, Christopher W Dunn, David C Atkins, and Jerry S Phelps. 2004. The emerging evidence base for motivational interviewing: A meta-analytic and qualitative inquiry. *Journal of Cognitive Psychotherapy*, 18(4):309–322.

Bill Byrne, Karthik Krishnamoorthi, Chinnadhurai Sankar, Arvind Neelakantan, Daniel Duckworth, Semih Yavuz, Ben Goodrich, Amit Dubey, Andy Cedilnik, and Kyu-Young Kim. 2019. Taskmaster-1: Toward a realistic and diverse dialog dataset. *arXiv preprint arXiv:1909.05358*.

S Cai and K Knight. 2013. Smatch: an Evaluation Metric for Semantic Feature Structures. *ACL (2)*.

Doğan Can, David C Atkins, and Shrikanth S Narayanan. 2015. A dialog act tagging approach to behavioral coding: A case study of addiction counseling conversations. In *Sixteenth Annual Conference of the International Speech Communication Association*.

Jie Cao, Michael Tanana, Zac Imel, Eric Poitras, David Atkins, and Vivek Srikumar. 2019a. Observing dialogue in therapy: Categorizing and forecasting behavioral codes. In *Proceedings of ACL 2019*.

Jie Cao and Yi Zhang. 2021. A comparative study on schema-guided dialogue state tracking. In *Proceedings of NAACL 2021*.

Jie Cao, Yi Zhang, Adel Youssef, and Vivek Srikumar. 2019b. Amazon at mrp 2019: Parsing meaning representations with lexical and phrasal anchoring. In *Proceedings of the Shared Task on MRP at the CoNLL 2019*.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. 2003. Building a discourse-tagged corpus in the framework of rhetorical structure theory. In *Current and new directions in discourse and dialogue*, pages 85–112. Springer.

Kai-Wei Chang, Akshay Krishnamurthy, Alekh Agarwal, Hal Daumé III, and John Langford. 2015. Learning to search better than your teacher. In *International Conference on Machine Learning*, pages 2058–2066. PMLR.

Ming-Wei Chang, Lev Ratinov, and Dan Roth. 2012. Structured learning with constrained conditional models. *Machine learning*, 88(3):399–431.

Wanxiang Che, Longxu Dou, Yang Xu, Yuxuan Wang, Yijia Liu, and Ting Liu. 2019. HIT-SCIR at MRP 2019: A unified pipeline for meaning representation parsing via efficient training and effective encoding. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 76 – 85, Hong Kong, China.

Wei-Te Chen and Martha Palmer. 2017. Unsupervised amr-dependency parse alignment. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 558–567.

Jianpeng Cheng, Devang Agrawal, Héctor Martínez Alonso, Shruti Bhargava, Joris Driesen, Federico Flego, Dain Kaplan, Dimitri Kartsaklis, Lin Li, Dhivya Piraviperumal, et al. 2020. Conversational semantic parsing for dialog state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8107–8117.

Yoeng-Jin Chu. 1965. On the shortest arborescence of a directed graph. *Scientia Sinica*, 14:1396–1400.

John Cocke. 1969. *Programming languages and their compilers: Preliminary notes*. New York University.

Kenneth Mark Colby. 1975. *Artificial Paranoia: A Computer Simulation of Paranoid Process*. Pergamon Press.

Michael Collins. 2003. Head-driven statistical models for natural language parsing. *Computational linguistics*, 29(4):589–637.

Caio Corro and Ivan Titov. 2019. Learning latent trees with stochastic perturbations and differentiable dynamic programming. *ACL*.

Hal Daumé, John Langford, and Daniel Marcu. 2009. Search-based structured prediction. *Machine learning*, 75(3):297–325.

Franck Dernoncourt and Ji Young Lee. 2017. Pubmed 200k rct: a dataset for sequential sentence classification in medical abstracts. *IJCNLP 2017*, page 308.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Lucia Donatelli, Meaghan Fowlie, Jonas Groschwitz, Alexander Koller, Matthias Lindemann, Mario Mina, and Pia WeiÃ§enhorn. 2019. Saarland at MRP 2019: Compositional parsing across all graphbanks. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 66 – 75, Hong Kong, China.

Timothy Dozat and Christopher D Manning. 2016. Deep biaffine attention for neural dependency parsing. *ICLR*.

Jack Edmonds et al. 1967. Optimum branchings. *Journal of Research of the national Bureau of Standards B*, 71(4):233–240.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyang Gao, Adarsh Kumar, Anuj Goyal, Peter Ku, and Dilek Hakkani-Tur. 2020. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 422–428.

Charles J. Fillmore. 1968. The case for case. In Emmon Bach and Robert T. Harms, editors, *Universals in Linguistic Theory*, pages 1 – 88. Holt, Rinehart and Winston, London.

Jenny Rose Finkel, Trond Grenager, and Christopher D Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd annual meeting of the association for computational linguistics (ACLâĂŹ05)*, pages 363–370.

Jeffrey Flanigan, Sam Thomson, Jaime G Carbonell, Chris Dyer, and Noah A Smith. 2014. A Discriminative Graph-Based Parser for the Abstract Meaning Representation. *ACL*.

Daniel Flickinger, Valia Kordoni, Yi Zhang, AntÃşnio Branco, Kiril Simov, Petya Osenova, Catarina Carvalheiro, Francisco Costa, and SÃl'rgio Castro. 2012. ParDeepBank. Multiple parallel deep treebanking. In *Proceedings of the 11th International Workshop on Treebanks and Linguistic Theories*, pages 97 – 108, Lisbon, Portugal. EdiÃğÃţes Colibri.

Andrea Galassi, Marco Lippi, and Paolo Torroni. 2019. Attention, please! a critical review of neural attention models in natural language processing. *arXiv preprint arXiv:1902.02181*.

Shuyang Gao, Sanchit Agarwal, Tagyoung Chung, Di Jin, and Dilek Hakkani-Tur. 2020. From machine reading comprehension to dialogue state tracking: Bridging the gap. *arXiv preprint arXiv:2004.05827*.

Shuyang Gao, Abhishek Sethi, Sanchit Agarwal, Tagyoung Chung, Dilek Hakkani-Tur, and Amazon Alexa AI. 2019. Dialog state tracking: A neural reading comprehension approach. In *20th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, page 264.

Samuel J. Gershman. 2021. *What Makes Us Smart*. Princeton University Press.

James Gibson, Dogan Can, Panayiotis Georgiou, David C Atkins, and Shrikanth S Narayanan. 2017. Attention networks for modeling behaviors in addiction counseling. In *Proceedings of the 2016 Conference of the International Speech Communication Association INTERSPEECH*.

Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic parsing for task oriented dialog using hierarchical representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2787–2792, Brussels, Belgium. Association for Computational Linguistics.

Jan Hajic, Eva Hajicová, Jarmila Panevová, Petr Sgall, Ondrej Bojar, Silvie Cinková, Eva Fucíková, Marie Mikulová, Petr Pajas, Jan Popelka, et al. 2012. Announcing prague czech-english dependency treebank 2.0. In *LREC*, pages 3153–3160.

Ting Han, Ximing Liu, Ryuichi Takanobu, Yixin Lian, Chongxuan Huang, Wei Peng, and Minlie Huang. 2020. Multiwoz 2.3: A multi-domain task-oriented dataset enhanced with annotation corrections and co-reference annotation. *arXiv e-prints*, pages arXiv–2010.

Harry F Harlow. 1949. The formation of learning sets. *Psychological review*, 56(1):51.

Matthew Henderson, Blaise Thomson, and Steve Young. 2014. Word-based dialog state tracking with recurrent neural networks. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 292–299.

Daniel Hershcovich, Zohar Aizenbud, Leshem Choshen, Elior Sulem, Ari Rappoport, and Omri Abend. 2019. SemEval-2019 task 1: Cross-lingual semantic parsing with UCCA. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1–10, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Daniel Hershcovich and Ofir Arviv. 2019. TUPA at MRP 2019: A multi-task baseline system. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 28 – 39, Hong Kong, China.

Sepp Hochreiter and Jurgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Matthew Honnibal and Ines Montani. 2017. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. *To appear*.

Kurt Hornik, Maxwell Stinchcombe, and Halbert White. 1989. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *arXiv preprint arXiv:2005.00796*.

Dirk Hovy, Stephen Tratz, and Eduard Hovy. 2010. WhatâĂŹs in a preposition? dimensions of sense disambiguation for an interesting word class. In *Coling 2010: Posters*, pages 454–462.

Xiaolei Huang, Lixing Liu, Kate Carey, Joshua Woolley, Stefan Scherer, and Brian Borsari. 2018. Modeling temporality of human intentions by domain adaptation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 696–701.

Samuel Humeau, Kurt Shuster, Marie-Anne Lachaux, and Jason Weston. 2019. Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring. In *ICLR*.

Zac E Imel, Derek D Caperton, Michael Tanana, and David C Atkins. 2017. Technology-enhanced human interaction in psychotherapy. *Journal of counseling psychology*, 64(4):385.

Angelina Ivanova, Stephan Oepen, Lilja Øvrelid, and Dan Flickinger. 2012a. Who did what to whom?: A contrastive study of syntacto-semantic dependencies. In *Proceedings of the sixth linguistic annotation workshop*, pages 2–11. Association for Computational Linguistics.

Angelina Ivanova, Stephan Oepen, Lilja Øvrelid, and Dan Flickinger. 2012b. Who did what to whom? A contrastive study of syntacto-semantic dependencies. In *Proceedings of the 6th Linguistic Annotation Workshop*, pages 2 – 11, Jeju, Republic of Korea.

Wei Jiang, Yu Zhang, Zhenghua Li, and Min Zhang. 2019. Hlt@ suda at semeval 2019 task 1: Ucca graph parsing as constituent tree parsing. *arXiv preprint arXiv:1903.04153*.

Richard Johansson and Pierre Nugues. 2008. The effect of syntactic representation on semantic role labeling. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 393 – 400, Manchester, UK.

Dan Jurafsky and James H Martin. 2019. *Speech and language processing*. Pearson.

Pavan Kapanipathi, Ibrahim Abdelaziz, Srinivas Ravishankar, Salim Roukos, Alexander Gray, Ramón Fernandez Astudillo, Maria Chang, Cristina Cornelio, Saswati Dana, Achille Fokoue-Nkoutche, et al. 2021. Leveraging abstract meaning representation for knowledge base question answering. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3884–3894.

Tadao Kasami. 1966. An efficient recognition and syntax-analysis algorithm for context-free languages. *Coordinated Science Laboratory Report no. R-257*.

John F Kelley. 1984. An iterative design methodology for user-friendly natural language office information applications. *ACM Transactions on Information Systems (TOIS)*, 2(1):26–41.

Sungdong Kim, Sohee Yang, Gyuwan Kim, and Sang-Woo Lee. 2019. Efficient dialogue state tracking by selectively overwriting memory. *arXiv preprint arXiv:1911.03906*.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *ICLR*.

Paul Kingsbury and Martha Palmer. 2002. From TreeBank to PropBank. In *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, pages 1989 – 1993, Las Palmas, Spain.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. *arXiv preprint arXiv:1805.01052*.

Ben Taskar Carlos Guestrin Daphne Koller. 2004. Max-margin markov networks. *Advances in neural information processing systems*, 16:25.

Ioannis Konstas, Srinivasan Iyer, Mark Yatskar, Yejin Choi, and Luke Zettlemoyer. 2017. Neural amr: Sequence-to-sequence models for parsing and generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 146–157.

Terry Koo, Amir Globerson, Xavier Carreras, and Michael Collins. 2007. Structured prediction models via the matrix-tree theorem. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 141–150, Prague, Czech Republic. Association for Computational Linguistics.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289.

Brenden M. Lake, Tomer David Ullman, Joshua B. Tenenbaum, and Samuel J. Gershman. 2016. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40.

Christine M Lee, Jason R Kilmer, Clayton Neighbors, David C Atkins, Cheng Zheng, Denise D Walker, and Mary E Larimer. 2013. Indicated prevention for college student marijuana use: A randomized controlled trial. *Journal of consulting and clinical psychology*, 81(4):702.

Christine M Lee, Clayton Neighbors, Melissa A Lewis, Debra Kaysen, Angela Mittmann, Irene M Geisner, David C Atkins, Cheng Zheng, Lisa A Garberson, Jason R Kilmer, et al. 2014. Randomized controlled trial of a spring break intervention to reduce high-risk drinking. *Journal of consulting and clinical psychology*, 82(2):189.

Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. 2019. Sumbt: Slot-utterance matching for universal and scalable belief tracking. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5478–5483.

Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015a. A hierarchical neural autoencoder for paragraphs and documents. *arXiv preprint arXiv:1506.01057*.

Shaohua Li, Jun Zhu, and Chunyan Miao. 2015b. A generative word embedding model and its low rank positive semidefinite solution. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1599–1609.

Tao Li and Vivek Srikumar. 2019. Augmenting Neural Networks with First-order Logic. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Zuchao Li, Hai Zhao, Zhuosheng Zhang, Rui Wang, Masao Utiyama, and Eiichiro Sumita. 2019. SJTU–NICT at MRP 2019: Multi-task learning for end-to-end uniform semantic graph parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 45 – 54, Hong Kong, China.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A Smith. 2015. Toward abstractive summarization using semantic representations. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1077–1086.

Yang Liu and Mirella Lapata. 2018. Learning structured text representations. *Transactions of the Association for Computational Linguistics*, 6:63–75.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015a. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *The 16th Annual SIGdial Meeting on Discourse and Dialogue*.

Ryan Lowe, Nissan Pow, Iulian V. Serban, and Joelle Pineau. 2015b. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *Proceedings of SIGDIAL*.

Brad W Lundahl, Chelsea Kunz, Cynthia Brownell, Derrik Tollefson, and Brian L Burke. 2010. A meta-analysis of motivational interviewing: Twenty-five years of empirical studies. *Research on social work practice*, 20(2):137–160.

Minh-Thang Luong, Ilya Sutskever, Quoc V Le, Oriol Vinyals, and Wojciech Zaremba. 2014. Addressing the rare word problem in neural machine translation. *IJCNLP*.

Chunchuan Lyu and Ivan Titov. 2018. Amr parsing as graph prediction with latent alignment. *ACL*.

Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The Penn Treebank: Annotating predicate argument structure. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

Mitchell Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpora of English. The Penn Treebank. *Computational Linguistics*, 19:313 – 330.

Renata K Martins and Daniel W McNeil. 2009. Review of motivational interviewing in promoting health behaviors. *Clinical psychology review*, 29(4):283–293.

J May. 2016. SemEval-2016 Task 8: Meaning Representation Parsing. In *Proceedings of SemEval*, pages 1063–1073, Stroudsburg, PA, USA. Association for Computational Linguistics.

Gonzalo Mena, David Belanger, Scott Linderman, and Jasper Snoek. 2018. Learning latent permutations with gumbel-sinkhorn networks. *arXiv preprint arXiv:1802.08665*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3111–3119.

George A Miller. 1998. *WordNet: An electronic lexical database*. MIT press.

William Miller and Stephen Rollnick. 2003. Motivational interviewing: Preparing people for change. *Journal for Healthcare Quality*, 25(3):46.

William R Miller, Theresa B Moyers, Denise Ernst, and Paul Amrhein. 2003. Manual for the motivational interviewing skill code (misc). *Unpublished manuscript. Albuquerque: Center on Alcoholism, Substance Abuse and Addictions, University of New Mexico*.

William R Miller and Stephen Rollnick. 2012. *Motivational interviewing: Helping people change*. Guilford press.

Tom M Mitchell. 1980. *The need for biases in learning generalizations*. Department of Computer Science, Laboratory for Computer Science Research âĂę.

Yusuke Miyao, Stephan Oepen, and Daniel Zeman. 2014. In-house: An ensemble of pre-existing off-the-shelf parsers. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 335–340.

Nafise Sadat Moosavi and Michael Strube. 2018. Using linguistic features to improve the generalization capability of neural coreference resolvers. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 193–203, Brussels, Belgium. Association for Computational Linguistics.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2017. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1777–1788.

K Murphy. 2022. Probabilistic machine learning: Advanced topics.

Chetan Naik, Arpit Gupta, Hancheng Ge, Mathias Lambert, and Ruhi Sarikaya. 2018. Contextual slot carryover for disparate schemas. *Proc. Interspeech 2018*, pages 596–600.

Mahdi Namazifar, Alexandros Papangelis, Gokhan Tur, and Dilek Hakkani-Tür. 2020. Language model is all you need: Natural language understanding as question answering. *arXiv preprint arXiv:2011.03023*.

Clayton Neighbors, Christine M Lee, David C Atkins, Melissa A Lewis, Debra Kaysen, Angela Mittmann, Nicole Fossos, Irene M Geisner, Cheng Zheng, and Mary E Larimer. 2012. A randomized controlled trial of event-specific prevention strategies for reducing problematic drinking associated with 21st birthday celebrations. *Journal of consulting and clinical psychology*, 80(5):850.

Vahid Noroozi, Yang Zhang, Evelina Bakhturina, and Tomasz Kornuta. 2020. A fast and robust bert-based dialogue state tracker for schema-guided dialogue dataset. *Workshop on Conversational Systems Towards Mainstream Adoption at KDD 2020*.

Stephan Oepen, Omri Abend, Jan Hajič, Daniel Hershcovich, Marco Kuhlmann, Tim O'Gorman, Nianwen Xue, Jayeol Chun, Milan Straka, and Zdeňka Urešová. 2019. MRP 2019: Cross-framework Meaning Representation Parsing. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 1 – 27, Hong Kong, China.

Stephan Oepen and Dan Flickinger. 2019. The ERG at MRP 2019: Radically compositional semantic dependencies. In *Proceedings of the Shared Task on Cross-Framework Meaning Representation Parsing at the 2019 Conference on Natural Language Learning*, pages 40 – 44, Hong Kong, China.

Stephan Oepen, Daniel Flickinger, Kristina Toutanova, and Christopher D. Manning. 2004. LinGO Redwoods. A rich and dynamic treebank for HPSG. *Research on Language and Computation*, 2(4):575 – 596.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Silvie Cinková, Dan Flickinger, Jan Hajič, and Zdeňka Urešová. 2015. SemEval 2015 Task 18. Broad-coverage semantic dependency parsing. In *Proceedings of the 9th International Workshop on Semantic Evaluation*, page 915 – 926, Bolder, CO, USA.

Stephan Oepen, Marco Kuhlmann, Yusuke Miyao, Daniel Zeman, Dan Flickinger, Jan Hajič, Angelina Ivanova, and Yi Zhang. 2014. SemEval 2014 Task 8. Broad-coverage semantic dependency parsing. In *Proceedings of the 8th International Workshop on Semantic Evaluation*, page 63 – 72, Dublin, Ireland.

Stephan Oepen and Jan Tore Lønning. 2006. Discriminant-based MRS banking. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, pages 1250 – 1255, Genoa, Italy.

Stephan Oepen and Jan Tore Lønning. 2006. Discriminant-based mrs banking. In *LREC*, pages 1250–1255.

Maria Leonor Pacheco and Dan Goldwasser. 2021. Modeling content and context with deep relational learning. *Transactions of the Association for Computational Linguistics*, 9:100–119.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106.

George Papandreou and Alan Yuille. 2011. Perturb-and-map random fields: Using discrete optimization to learn and sample from energy models–iccv 2011 paper supplementary material–. *ICCV*.

Debjyoti Paul*, Jie Cao*, Feifei Li, and Vivek Srikumar. 2021. Database workload characterization with query plan encoders. *VLDB'22*.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2020. Soloist: Few-shot task-oriented dialog with a single pre-trained auto-regressive model. *arXiv*, pages arXiv–2005.

Xiaochang Peng, Linfeng Song, and Daniel Gildea. 2015. A Synchronous Hyperedge Replacement Grammar based approach for AMR parsing. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 32–41, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

Verónica Pérez-Rosas, Rada Mihalcea, Kenneth Resnicow, Satinder Singh, Lawrence Ann, Kathy J Goggin, and Delwyn Catley. 2017. Predicting counselor behaviors in motivational interviewing encounters. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 1128–1137.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018a. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018b. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018c. Deep contextualized word representations. In *Proc. of NAACL*.

Matthew E Peters, Sebastian Ruder, and Noah A Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019)*, pages 7–14.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Carl Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. Studies in Contemporary Linguistics. The University of Chicago Press, Chicago, USA.

Nima Pourdamghani, Yang Gao, Ulf Hermjakob, and Kevin Knight. 2014. Aligning English Strings with Abstract Meaning Representation Graphs. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 425–429, Stroudsburg, PA, USA. Association for Computational Linguistics.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind K Joshi, and Bonnie L Webber. 2008. The penn discourse treebank 2.0. In *LREC*. Citeseer.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2):257–287.

Michael Pust, Ulf Hermjakob, Kevin Knight, Daniel Marcu, and Jonathan May. 2015. Using Syntax-Based Machine Translation to Parse English into Abstract Meaning Representation. *arXiv.org*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/research-covers/language-unsupervised/language_ understanding_paper. pdf*.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2019. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. *AAAI 2019*.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Schema-guided dialogue state tracking task at dstc8. *Workshop on DSTC8, AAAI 2020*.

Bob Rehder. 2003. A causal-model theory of conceptual representation and categorization. *Journal of experimental psychology. Learning, memory, and cognition*, 29 6:1141–59.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings.

Dan Roth and Wen-tau Yih. 2005. Integer linear programming inference for conditional random fields. In *Proceedings of the 22nd international conference on Machine learning*, pages 736–743.

Dan Roth and Wen-tau Yih. 2007. Global inference for entity and relation identification via a linear programming formulation. *Introduction to statistical relational learning*, pages 553–580.

Peter Roy-Byrne, Kristin Bumgardner, Antoinette Krupski, Chris Dunn, Richard Ries, Dennis Donovan, Imara I West, Charles Maynard, David C Atkins, Meredith C Graves, et al. 2014. Brief intervention for problem drug use in safety-net primary care settings: a randomized clinical trial. *Jama*, 312(5):492–501.

Alexander M Rush and MJ Collins. 2012. A tutorial on dual decomposition and lagrangian relaxation for inference in natural language processing. *Journal of Artificial Intelligence Research*, 45:305–362.

Jost Schatzmann, Kallirroi Georgila, and Steve Young. 2005. Quantitative evaluation of user simulation techniques for spoken dialogue systems. In *6th SIGdial Workshop on DISCOURSE and DIALOGUE*.

Craig S Schwalbe, Hans Y Oh, and Allen Zweben. 2014. Sustaining motivational interviewing: a meta-analysis of training studies. *Addiction (Abingdon, England)*, 109(8):1287–94.

Rico Sennrich, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, and Maria Nădejde. 2017. Nematus: a toolkit for neural machine translation. In *Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 65–68, Valencia, Spain. Association for Computational Linguistics.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. In *ICLR*.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, volume 16, pages 3776–3784.

Petr Sgall, Eva Hajičová, and Jarmila Panevová. 1986. *The Meaning of the Sentence and Its Semantic and Pragmatic Aspects*. D. Reidel Publishing Company, Dordrecht, The Netherlands.

Pararth Shah, Dilek Hakkani-Tür, Bing Liu, and Gokhan Tür. 2018a. Bootstrapping a neural conversational agent with dialogue self-play, crowdsourcing and on-line reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, pages 41–51, New Orleans - Louisiana. Association for Computational Linguistics.

Pararth Shah, Dilek Hakkani-Tür, Gokhan Tür, Abhinav Rastogi, Ankur Bapna, Neha Nayak, and Larry Heck. 2018b. Building a conversational agent overnight with dialogue self-play. *arXiv preprint arXiv:1801.04871*.

Sameer Singh, Michael Wick, and Andrew McCallum. 2012. Monte carlo mcmc: efficient inference by approximate sampling. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1104–1113.

Noah A Smith. 2011. Linguistic structure prediction. *Synthesis lectures on human language technologies*, 4(2):1–274.

Noah A Smith and Jason Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACLâĂŹ05)*, pages 354–362.

Linfeng Song, Daniel Gildea, Yue Zhang, Zhiguo Wang, and Jinsong Su. 2019. Semantic neural machine translation using amr. *Transactions of the Association for Computational Linguistics*, 7:19–31.

Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. 2015. A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 553–562. ACM.

Elizabeth S. Spelke. 1990. Principles of object perception. *Cognitive Science*, 14:29–56.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics*, 26(3):339–373.

Emma Strubell, Patrick Verga, Daniel Andor, David Weiss, and Andrew McCallum. 2018. Linguistically-informed self-attention for semantic role labeling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5027–5038, Brussels, Belgium. Association for Computational Linguistics.

Ida Szubert, Adam Lopez, and Nathan Schneider. 2018. A structured syntax-semantics interface for english-amr alignment. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1169–1180.

Michael Tanana, Kevin A Hallgren, Zac E Imel, David C Atkins, and Vivek Srikumar. 2016. A comparison of natural language processing methods for automated coding of motivational interviewing. *Journal of substance abuse treatment*, 65:43–50.

Sean J Tollison, Christine M Lee, Clayton Neighbors, Teryl A Neil, Nichole D Olson, and Mary E Larimer. 2008. Questions and reflections: the use of motivational interviewing microskills in a peer-led brief alcohol intervention for college students. *Behavior Therapy*, 39(2):183–194.

Lifu Tu and Kevin Gimpel. 2018. Learning approximate inference networks for structured prediction. *ICLR*.

Zdeňka Urešová, Eva Fučíková, and Jana Šindlerová. 2016. Czengvallex: a bilingual czech-english valency lexicon. *The Prague Bulletin of Mathematical Linguistics*, 105(1):17.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Nikhita Vedula, Nedim Lipka, Pranav Maneriker, and Srinivasan Parthasarathy. 2020. Open intent extraction from natural language interactions. In *Proceedings of The Web Conference 2020*, pages 2009–2020.

Andrew Viterbi. 1967. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE transactions on Information Theory*, 13(2):260–269.

Chuan Wang and Nianwen Xue. 2017. Getting the most out of amr parsing. In *Proceedings of the 2017 conference on empirical methods in natural language processing*, pages 1257–1268.

Chuan Wang, Nianwen Xue, and Sameer Pradhan. 2015. Boosting Transition-based AMR Parsing with Refined Actions and Auxiliary Analyzers. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1–6.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 189–198.

Joseph Weizenbaum. 1966. ELIZA – a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45.

TomáËĞs Werner and Daniel PruËĞsa. 2014. The power of lp relaxation for map inference. *Advanced Structured Prediction*, page 19.

Jason Williams, Antoine Raux, and Matthew Henderson. 2016. The dialog state tracking challenge series: A review. *Dialogue & Discourse*, 7(3):4–33.

Andrew G Wilson and Pavel Izmailov. 2020. Bayesian deep learning and a probabilistic perspective of generalization. *Advances in neural information processing systems*, 33:4697–4708.

Ludwig Wittgenstein. 2010. *Philosophical investigations*. John Wiley & Sons.

Florian Wolf and Edward Gibson. 2005. Representing discourse coherence: A corpus-based study. *Computational linguistics*, 31(2):249–287.

David H Wolpert, William G Macready, et al. 1995. No free lunch theorems for search. Technical report, Technical Report SFI-TR-95-02-010, Santa Fe Institute.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 808–819.

Yu Wu, Wei Wu, Chen Xing, Ming Zhou, and Zhoujun Li. 2017. Sequential matching network: A new architecture for multi-turn response selection in retrieval-based chatbots. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 496–505.

Bo Xiao, Dogan Can, James Gibson, Zac E Imel, David C Atkins, Panayiotis G Georgiou, and Shrikanth S Narayanan. 2016. Behavioral coding of therapist language in addiction counseling using recurrent neural networks. In *Proceedings of the 2016 Conference of the International Speech Communication Association INTERSPEECH*, pages 908–912.

Liu Yang, Minghui Qiu, Chen Qu, Jiafeng Guo, Yongfeng Zhang, W Bruce Croft, Jun Huang, and Haiqing Chen. 2018. Response ranking with deep matching networks and external knowledge in information-seeking conversation systems. *arXiv preprint arXiv:1805.00188*.

Pengcheng Yin, Chunting Zhou, Junxian He, and Graham Neubig. 2018. Structvae: Tree-structured latent variable models for semi-supervised semantic parsing. *ACL*.

Koichiro Yoshino, Chiori Hori, Julien Perez, Luis Fernando D'Haro, Lazaros Polymenakos, Chulaka Gunasekara, Walter S. Lasecki, Jonathan Kummerfeld, Michael Galley, Chris Brockett, Jianfeng Gao, Bill Dolan, Sean Gao, Tim K. Marks, Devi Parikh, and Dhruv Batra. 2018. The 7th dialog system technology challenge. *arXiv preprint*.

Daniel H Younger. 1967. Recognition and parsing of context-free languages in time n3. *Information and control*, 10(2):189–208.

Xiaoxue Zang, Abhinav Rastogi, Srinivas Sunkara, Raghav Gupta, Jianguo Zhang, and Jindong Chen. 2020. MultiWOZ 2.2 : A dialogue dataset with additional annotation corrections and state tracking baselines. In *Proceedings of the 2nd Workshop on Natural Language Processing for Conversational AI*, pages 109–117, Online. Association for Computational Linguistics.

Jian-Guo Zhang, Kazuma Hashimoto, Chien-Sheng Wu, Yao Wan, Philip S Yu, Richard Socher, and Caiming Xiong. 2019a. Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. *arXiv*, pages arXiv–1910.

Justine Zhang, Jonathan P Chang, Cristian Danescu-Niculescu-Mizil, Lucas Dixon, Yiqing Hua, Nithum Thain, and Dario Taraborelli. 2018. Conversations Gone Awry: Detecting Early Signs of Conversational Failure. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Sheng Zhang, Xutai Ma, Kevin Duh, and Benjamin Van Durme. 2019b. AMR Parsing as Sequence-to-Graph Transduction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Florence, Italy. Association for Computational Linguistics.