INDUCTIVE BIASES FOR DEEP LINGUISTIC STRUCTURED PREDICTION WITH 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

August 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)	**07/22/2022** Date Approved
Ellen M Riloff ,	Member	**07/22/2022** Date Approved
Qingyao Ai ,	Member	**07/22/2022** Date Approved
Zachary E Imel ,	Member	Date Approved
Yi Zhang ,	Member	**07/22/2022** Date Approved

by **Mary Hall** , Chair/Dean of
the Department/College/School of **Computing**
and by **David B. Kieda** , Dean of The Graduate School.

ABSTRACT

Discovering the underlying structure of unstructured text can help make sense of the rapidly growing textual data. This dissertation 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 dissertation, 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

ACKNOWLEDGEMENTS

The past 7 years of Ph.D. study have been an unforgettable and invaluable experience for me. I would not have been able to complete this dissertation without the guidance, support, and efforts of so many people with whom I have interacted.

First and foremost, I am profoundly grateful to my advisor, Vivek Srikumar, who has demonstrated to me how to be an excellent researcher, advisor, teacher, and friend. He encouraged me to freely explore and choose my favorite topics to start rather than immediately following those rising hot topics on deep learning. I always remember his encouragement during my first seminar talk on Combinatorial Categorical Grammar (CCG), for that, I was attracted by a bunch of books by Mark Steedman. Thanks to this early exploration, I build my early research interests in compositional semantics. It also inspired my final dissertation on independent factorization. He always believed in me and encouraged me in my early dream of building the world's best AMR Parser. However, I was easily attracted by engineering details and got stuck in many 'local minimum'. I always felt shame for my unproductive in my early years, but he taught me a lot during my difficult time. One important lesson is to write more. He always reminds me to write more papers than codes and pulls me back to think more about important problems rather than the details. More importantly, another lesson is to upgrade your weapon when struggling too long. He urged me archieve the old AMR parser based on feature engineering and try deep learning on a new task. Such weapon upgrading also made my early dream come true at last: our AMR parser ranked 1st in MRP'2019 shared task. I am forever grateful for his extremely kind, caring, and supportive.

I want to thank Ellen Riloff. It is her NLP course leading me into the NLP field. She named our team "Talking Geckos", which won 1st in the QA challenge (Fall 2015, NLP Course project). This early course award also encouraged me for my Ph.D. study. I appreciate her suggestions and comments on my research work. I am also impressed by her meticulous scholarship; I believe it will also impact me in my future career. More

importantly, I want to thank Ellen and Vivek for setting up the great Utah NLP group, my home at the U, and I will miss this family and the MEB 4130.

It is also my great honor to have Zac Imel, Qingyao Ai, and Yi Zhang on my thesis committee. Thank you all for supervising my dissertation research and providing valuable input. I thank Zac for his great support in my first interdisciplinary collaboration on psychotherapy dialogue. I thank Qingyao for his detailed suggestions on my thesis proposal, presentation skills, and thesis writing. I thank Yi for his mentoring on two wonderful internships in Amazon Lex, and the two projects eventually led to the important parts of this dissertation.

I have been so lucky to have a good number of friends in Utah NLP Group, with a special mention to Youngjun Kim, Ashequl Qadir, Haibo Ding, Xingyuan Pan, Tao Li, Yichu Zhou, Annie Cherkaev, Tianyu Jiang, Yuan Zhuang, Zhenduo Wang, Vivek Gupta, Maitrey Mehta, Mattia Medina-Grespan, Ashim Gupta, Nate Stringham and Tarun Sunkaraneni. I I will never forget the amazing time I had with them.

During my Ph.D., I have done three unforgettable internships at WeChat AI and AWS AI (two times in Amazon Lex Team). I thank my mentors Yik-Cheung Tam, Cheng Niu, Yi Zhang, and Adel Youssef when I worked at these places. I thank my other collaborators out of my lab: Michael Tanana, Eric Poitras, Zhiqiang Liu, Zuohui Fu, Shuo Sun, Debjyoti Paul, Zhimin Li, and others. I immensely enjoyed the collaborations.

I want to thank my parents, Xianhua Cao and Yuanxiang Wang, and my elder sisters, Wenjuan and Li. When I decided to leave my previous industrial job and study abroad, they may feel confused, but they gave me unbending support and constant encouragement. Without them, I cannot be the current me today. I never know how to pay them back. I also want to thank my friends, Min Xian and Fei Xu, who make me like home. I want to thank Xin Yu for her love and support. Without her, I would not have come to the U and would not continue my research as a Postdoctoral at CU Boulder.

Last but not least, I would like to acknowledge an NSF Cyberlearning grant (#182287) and a GPU gift from NVIDIA Corporation. Their valuable support helps me to complete this dissertation.

CHAPTER 1

INTRODUCTION

Human language is essential for human intelligence and arguably our most powerful tool for learning and transmitting 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 information processing ability, we cannot easily discover the knowledge hidden in the vast amount of unstructured text.

One classical way to study unstructured natural language is to represent the language via various structured symbolic representations at different levels [1]. Before the revolution of representation learning with deep learning, the NLP community had put decades of effort into solving other linguistic structured prediction tasks to get various aspects of text understanding. Let us look at some examples.

Firstly, we illustrate the *classic linguistic structures*, such as part-of-speech (POS), constituency and dependency trees. 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 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 modeling 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 dependency 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 headwords 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., 2, 3, 4], they are still helpful in deep learning based systems [5, 6, 7].

Secondly, we exmaine *broad-coverage meaning representations*. Besides the above structures capturing specific lexical, syntactic or semantic information, a broad-coverage semantic representation is a general-purpose meaning representation language aiming to represent the multiple phenomena in a single structure for broad-coverage text. Figure 1.4 shows the Abstract Meaning Representation [*AMR*, 8] of the example sentence. The node in the graph represents 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*" represents the word sense predefined in the Propbank [9]; 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 coreference information. Figure 1.5 shows the foundational layer of Universal Conceptual Cognitive Annotation [*UCCA*, 10], 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 the 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*, 11] and Prague Semantic Dependencies [*PSD*, 12, 13]. More details about their captured semantic content and their structure properties will be introduced comparatively in Section 2.2.1.

Finally, besides the above broad coverage syntactic and semantic structures in natural language, researchers have designed various *symbolic representations for specific applications*. Dialogue acts are firstly designed to represent the speech act or intention of each utterance, to represent the functions of each utterance in the dialogue [14, 15]. Then inspired by the

---

[1] More details about various semantic phenomena will be introduced in Section 2.2.1

[2] AMR concepts include PropBank framesets, and other special dates, spatial entities, etc. More details about AMR will be introduced in Section 2.2.1

case theory [16], frame-based representation in GUS [17] are introduced to represent the state of dialogue, which consists of a collection of slots and each with a set of possible values. Figure 1.6 shows an example of dialogue state tracking, where each table is filled with intent, slot, and slot values, representing a dialogue 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, and programmable reasoning. Furthermore, they can be potentially helpful to offer actionable suggestions to guide human behavior, such as improving mental health counseling [18], dialogue state tracking [19], scientific document analysis [20], 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 feed a large amount of labeled raw data into an end-to-end deep learning model and outperform many previous pipeline models built from hand-crafted features. Recently, pretrained large language models even became the unified base model for many of the NLP tasks, which further boosts the performance.

However, recent research has 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 interoperability and robustness and they are fragile to maintain when deployed to real users. Using those large language models without any careful intervention can lead to fairness issues [21]. 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 [22], summarization [23], question answering [24] and so on.

Predicting structured representations of text is essential for natural language processing, even in the deep learning era. In this dissertation, 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 study 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 examine 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 dissertation addresses 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 based on particular and limited data must constrain its hypotheses somehow. 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 [25], which can potentially be helpful for machine intelligence. According to extensive cognitive science studies [25, 26, 27, 28, 29, 30], there are many inductive biases for human intelligence, such as compositionality, causality, learning to learn, etc. We do not imply machine intelligence should mimic human intelligence. Instead, we argue that those key human inductive biases help overcome limited observations and resources that may inspire us to design machine intelligence.

On the machine intelligence side, the no-free-lunch theorem for machine learning [31, 32] tells us that inductive biases that influence hypodissertation selection is necessary to obtain generalization. Mitchell [33] argues that inductive biases constitute the heart of generalization and, indeed a key basis for learning itself.

1.1.1.1 The Definition of Inductive Biases

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 with some training examples that demonstrate the intended relationship between the input and output values. Then the learner is supposed to learn a target function that captures the correlations 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 to generalize unseen data a generalization. This generalization problem cannot be solved without additional assumptions since unseen situations might have an arbitrary output value. The kind of necessary assumptions is subsumed in the phrase inductive bias.

In this dissertation, following the definition of bias in [33], we define *indutive bias* as: "Any bias for choosing one generalization over another, other than strict consistency with the observed training instances".

1.1.1.2 The Use of Inductive Biases

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

For the popular supervised learning setting, let $\mathcal{H}$ refer to machine learning model families, including deep learning models. Finding a target hypodissertation h is reduced to estimating the model parameters by fitting the training data. Hence, preferences beyond training data can naturally be organized into two goals: choosing the hypodissertation class $\mathcal{H}$ and finding the h is necessary to generalize to new data. For example, different *model families* can represent different hypodissertation 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 separable, the choices of *kernels design* also introduce inductive bias in kernel-based SVM models. There are also many assumptions about optimization for finding the specific hypodissertation h. For example, smoothness assumptions in the *optimization* method, such as Stochastic gradient descent, were 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 hypodissertation 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 hypodissertation 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 against a particular demographic group, which often leads to model predictions that unfairly disadvantage members of that group. Hence, *Data manipulation* can also help find the desired hypodissertation by augmenting the original training data with inductive biases.

In this dissertation, we mainly use *neural architectures* and *data manipulation* to support our inductive biases for deep linguistic structure prediction. To help understand the inductive biases, in the following, we will first introduce two examples of inductive biases used in the deep learning era, then we present the main study the goal of using inductive biases for deep linguistic structured prediction with independent factorization.

1.1.2 Inductive Biases in Deep Learning

According to the universal approximation theorem [34], a properly parameterized neural network can represent any function. Furthermore, training data seems rich enough for many tasks nowadays. It seems purely data-driven deep learning can learn any target function. Then, what kind of inductive biases do we need in the deep learning era? In the following, we show two examples of inductive biases used in computer vision and natural language processing in the deep learning era.

1.1.2.1 Computer Vision Example: Shift-Invariant

As the image classification task is shown in Figure 1.7, if a model is trained on the first image with a cat in the bottom-left corner, we hope it can still predict "*cat*" when shift the cat to the upper-right corner. Considering a feed-forward neural network, which can capture any function, it may fail in this shifted case because not all the shifts will exist in the training data. Augmenting the training data with the shifted images may mitigate this problem. However, a more elegant way is to use convolutional neural networks with a pooling layer. The pooling operation over convolutional filters is largely shift-invariant [35]. Beyond the training data, the inductive bias here assumes the model should be shift-invariant. Similarly, on the third image in Figure 1.7, we also can assume the model should be rotation-invariant

to predict correctly on unseen rotated cat images [36].

1.1.2.2 Natural Language Example: Compositionality

For natural language processing, Figure 1.8 shows another example of inductive biases in the named entity recognization task. Natural language is naturally compositional. Beyond the training data, we mainly make assumptions about the compositional properties for unseen data. Imagining the training data contains the first annotated sentence: "The Boxer can find the bone the other dogs hid from it.", the word "Boxer" is annotated as "*ANIMAL*". Now we add unseen new words, such as "*cannot*" and "*Husky*", and also add unseen new phrases by swapping the phrases "*the bone*" and "*the other dogs*". Finally, it forms an unseen new sentence in the bottom of the figure. Can our model still predict the unseen word "*Husky*" in the unseen sentence as "*ANIMAL*" ? We can augment the training data by enumerating unseen combinations for the original training data. However, it is intractable. To achieve this inductive bias about compositionality, instead of augmenting all combinations, various techinique are proposed to capture the underlying composing patterns inspired by linguistic studies. For example, inspired by morphology study, word piece [37] and byte pair encoding [BPE, 38] may learn the representation of unseen words, and they are widely used in the recent advances in large language models, such as BERT [39], GPT3 [40]. For unseen phrases and sentences, inspired by the assumption about recurrent syntax and grammar, recurrent neural networks (such as LSTM [41], RNN [42], GRU [43], and etc.) are proposed to capture the compositional patterns of natural language sequences. Instead of the the strong recurrent assumption, a weaker word-order assumption also works quite well by using a positional encoding with self-attention mechanism in the popular transformer-based models [44]. Finally, inspired by the distributional hypodissertation, where words that are used and occur in the same contexts tend to purport similar meanings [45], bi-directional architectures are the standard for modeling word embedding and language models. (e.g., looking at both the past and future via Bi-directional LSTM, self-attention, and so on)

Besides the shift-invariant, rotation-invariant and compositional assumptions for the *input* image and natural language data, there are other assumptions beyond the training data that can be used in deep learning era. In this dissertation, we extend the compositional

inductive biases for linguistic structured prediction tasks.

1.1.3 Independent Factorization for Deep Linguistic Structured Prediction

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

First, beyond the compositionality of input natural language, we believe that both the input text and output structured symbolic representations will follow the principle of compositionality. The composition of input natural language and the composition of output representations are correlated with each other. Hence, we propose to use the *independent factorization* framework to study the compositional inductive biases in natural language and its output structures.

1.1.3.1 Independent Factorization

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 example, 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 representing 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 dialogue 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 depend on multiple subcomponents of x and y.

The interdependence assumptions between those sub-components in x and y are key in a 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 interdependence. In this dissertation, we always assume the *independent factorization*, where each factor c only depends on a well-segmented subset of subcomponents y_c and the aligned x components (anchors) $x_c = a(y_c)$. In other words, once the output is decomposed into mutually exclusive output segments, we consider each segment as an atomic output part, and each atomic part is independent from the others.

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

where $a(y_c)$ 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 (still constrained by some global constraints). In this way, independent factorization largely simplifies the learning of linguistic structured prediction. More importantly, it also makes the inference tractable, and 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. Thus, by leveraging the compositionality of the input and output, and the alignments

between their decompositions, we can focus on the prior knowledge about the correlations between the input and output. Futhermore, we can extend that compositionality to help the model to generalize to unseen inputs and output structures.

During inference, when we encounter a new sentence as shown in Figure 1.9, we hope the model can leaverage the seen decomposed mappings in the training data to generalize to the unseen sentences. In this way, we first prepare a list of candidate anchors $seg_{in}(x) = \{$"*The*", "*dog*", "*found*", "*the*", "*bone*", "*it*", "*hide*"$\}$, then the model will easily produce each independent prediction y_c of each anchor as $\{\phi$, "*dog*", "*find-01*", ϕ, "*bone*", "*it*", "*hide*"$\}$ because most of the decomposed inputs are seen in previous Figure 1.4. Then we assemble the non-empty y_c by predicting the relations between each other and finally forms y via postprocessing[3].

1.1.3.2 Three Steps in Independent Factorizations

In summary, in the independence factorization setting, we factorize the input and the output via the compositionality of both the input and output, and then we hope the model can learn correlations between the decomposed input and output parts. Hence, the whole structured prediction problem is reduced into three challenges:

- **Output Decomposition:** How to decompose the output y into a set of independent parts y_c.
- **Input Decomposition and Alignment Discovery:** How to decompose x and offer a set of candidates that may generate each independent part y_c.
- **Factor Modeling:** How to find the relevant parts $x_{a_{y_c}}$ in x aligned to y_c and model the factorized energy score $E(x, a(y_c), y_c)$

The first question on independently decomposing y is either straightforward or has been resolved by previously existing methods in our studied tasks. In this dissertation, we mainly focus on the remaining challenges on modeling alignment and representation learning, which requires different inductive biases to help with the modeling. We consider the inductive biases to help modeling the above structured correlations between the input

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

and output structures as *structual inductive biases*, then we also use *natural language as inductive biases* to extend the independent factorization for cross-domain and cross-task generalization.

1.1.3.3 Structural Inductive Biases

To design the independent factorization for a new task, we require the prior knowledge about structures of input and output, e.g., proper input decomposition, or the linguistic analysis on the semantic-syntactic interface for the alignment information, or more recent structural bias for deep learning based language modeling.

In this part, using the AMR parsing in Figure 1.10) as a running example, we consider the structural inductive biases for its independent factorization. The first sentence and its corresponding AMR graph (left) are in training data, and we need to build a model to predict the corresponding AMR graph for the unseen sentence. Hence, we consider the compositionality for both the input and output to offer more inductive biases beyond the training data. We leave more details of AMR in Section 2.2.1.3 and more inductive biases about decomposing AMR in Section 3.2.1.1. In this part, we mainly show the intuitive understanding of the structural inductive bias for the independent factorization.

- **Output Decomposition:** To decompose an AMR graph, we need to know the meaning of each part of the AMR graph. The nodes in AMR can be categorized into five main categories (as shown in Figure 1.10 and a figure in later section Figure 2.4): frame (e.g., "*find-01*", "*hide-01*"), basic concept (e.g., "*dog*", "*jury*"), string ("*Pierre Vinken*"), number (e.g., "*61*") and other constant (e.g., "-"), while there many templated subgraphs used to represent special entities in the AMR, such as quantities, named entities, special roles, and other entities in dates, times, percentages, phone, email, URLs. In this part, we mainly show an example of subgraph segmentation for AMR, which will simplify the independent factorization. As shown in the left AMR graph, we draw a rectangle for the subgraph "*(possible-01 :polarity -)*", which means it forms a subgraph when decomposing the output AMR graph. The whole subgraph will be aligned to the word "*cannot*" the first sentence. Furthermore, when there comes the second sentence, for the same word "*cannot*" in the new sentence, we hope the model can produce the same subgraph "*(possible-01 :polarity -)*" from it. Besides this

subgraph, other segmented constituents in the left AMR graph will be each single node. While in other AMR graphs (Figure 2.4), we must consider the subgraphs used to represent the special entities. According to the above knowledge about the AMR graph, we use rule-based recategorization preprocessing to do the segmentation, which is inspired by the previous work on addressing the data sparsity issue in AMR parsing [46, 47, 48, 49]

- **Input Decomposition and Alignment Discovery:** As shown in the first row of Lego blocks in Figure 1.10, it naturally forms the decomposition of the first sentence for the AMR parsing case. However, for a more complicated case in Figure 2.4, rather than each standalone token, we may still need to consider other multiword expressions and the special entities in the sentence. Furthermore, when considering the previous UCCA parsing example shown in Figure 1.5, we also need to decompose the input sentence into phrases so that it can be easily aligned to the non-terminal nodes in the decomposed UCCA graph. Another problem after the input decomposition is the alignment discovery problem. We can notice two "*dog*" lego blocks in the first sentence, and we also have two "*dog*" AMR nodes. We have the words "*from*" in the sentence. However, it cannot align to any node, the AMR nodes. We need to build an alignment model to distinguish the alignments between them. In sum, we must consider the input decomposition with the inductive biases on the semantic contents in the symbolic representations and how they are derived from the surface tokens.

- **Factor Modeling:** The above output and input decomposition are usually done with prior knowledge about the language and the corresponding symbolic representations. Once we get the segmented nodes or subgraphs for output decomposition and the Lego blocks as the input decomposition, the whole structured prediction problem is simplified to learning a set of models. For example, we hope the model can map the words "*cannot*" into the subgraph "*(possible-01 :polarity -)*", and "*hid*" into "*hide-01*". When there is explicit alignment information, it will be easy to model them with aligned input and output pairs. However, when there are uncertainties about the alignments, we may need to model the alignment discovery model with the factor modeling jointly. We propose a latent alignment model for AMR parsing

in Section 3.2. Besides that, considering the same word "*find*" in both sentences, we need a discriminative contextualized representation to predict the correct meaning representation as the "*find-01*" for the first sentence and the "*find-02*" for the second sentence. The efficient contextualized representation is also the key to making such independent factorization possible. We will introduce more details of the two-stage AMR parsing in Section 3.2, which introduces how to assemble those local decisions about nodes and edges into a graph.

In sum, although the independent factorization simplifies structured prediction into simpler classification problems, however, we need many inductive biases to make the right design choices. Besides the above AMR parsing running example, we also design different models with independent factorization for a set of tasks, e.g., parsing graph-based representations with lexical-anchoring (Section 3.2.1) and phrasal-anchoring (Section 3.3.2), observing dialog in therapy (Section 4.2).

1.1.3.4 Natural Language as Inductive Biases

We also study natural language description as inductive biases to describe the function of each decomposed output part. We still take the AMR Parsing in Figure 1.10 as a running example. When we don't know the meaning of "*find-01*" and "*find-02*", to make the word "*find*" in the second sentence can generate the "*find-02*" instead of "*find-01*", it still needs a lot of aligned pairs about the "*find*" from other training data to learn. This is true for both human and machines. However, if we look at the explanation of "*find-01*" and "*find-02*" in PropBank [9], we may easily find that "*find-01*" means discovery, while "*find-02*" means verdict. Hence, human now can easily tell that the second "*find*" should predict "*find-02*" because it means "*verdict*" by the jury, without any more examples about "*find-02*". Once we know the explanation of "*find-01*" and "*find-02*", we can disambiguate them because we understand the meaning of the natural language that is used for the explanation. We believe that the knowledge in the natural language can also be used as inductive biases for machine learning. In this dissertation, we mainly extend the independent factorization of task-oriented dialogue state tracking with natural language descriptions (Chapter 5). We show that natural language as inductive provides great zero-shot performance on unseen dialogue services.

In sum, motivated by the compositional property of natural language and related symbolic representation, we propose to use independent factorization to model the correlations between the input and output structures. Furthermore, to make the independent factorization work for our models, we present a detailed analysis of how to do it for each task, and how to use the above inductive biases to help model the independent factorization.

1.2 Dissertation Statement and Research Contributions

Our claim is that by designing *structural inductive biases* 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.

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

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

1. Based on the assumption of independent factorization, we proposed a unified parsing framework to support both *explicit lexical-anchoring* (including DELPH-IN MRS Bi-lexical Dependencies [DM, 11] and Prague Semantic Dependencies [PSD, 12, 13]), and *implicit lexical anchoring* (AMR). Over 16 teams in the shared tasks, my parser [50] *ranked 1st on AMR, 6th in DM, and 7th in PSD*. By combining Perturb-and-MAP sampling [51] with differentiable Gumbel-Softmax Sinkhorn Networks [52], 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, 53] and Task-oriented Dialog Parsing [TOP, 54] are similar to a 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 inference to the new UCCA and TOP parsing tasks. Powered by a strong span-representation learning method, my

system [50] *ranked 5/16 on UCCA parsing*, and it can be reused for TOP parsing after a few preprocessing steps, and outperform several baseline models.

2. To provide real-time guidance to therapists with a dialogue observer, we decompose the dialog structure analysis with two independent prediction tasks: (1) categorizing therapist and client MI behavioral codes and (2) forecasting 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 [55]. 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. By decomposing the dialogue state tracking into four independent sub-tasks, we use natural language description as inductive biases to describe the functions of each independent intent/slot label, thus capturing the functional overlapping between different services. We show that such natural language descriptions can support the zero-shot learning for each independent subtask for unseen service. 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, homogeneous, and heterogeneous evaluations.

1.3 Dissertation Outline

In the dissertation, we discuss prior work related to an application in its own chapter, instead of putting them all in a single chapter. Furthermore, at the beginning of each application chapter, we will first show the motivation and inductive biases of applying independent factorization to each task. Then we offer the details about how we model the independent factorization. This dissertation is divided into five parts, which we describe below.

Chapter 2 reviews the background of this dissertation study with two sections:

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

- *Structures in NLP.* We first provide the necessary background about natural language structures to highlight our contributions in the remaining chapters better.

Chapter 3 describes the details of *structural inductive biases for lexical and phrasal anchoring*. We first introduce lexical and phrasal anchoring analysis to decompose the output structures for independent factorization, 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 energy 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 presents structural inductive biases for sentenctial anchoring. Besides the above lexical and phrasal anchoring in a single sentence, 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, 56, 57], 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 the flow of the whole therapy session. We decompose the dialog structure analysis with two independent prediction tasks: categorizing and forecasting the dialog flow in the form of MISC codes. By developing a modular family of neural networks for two independent tasks, we show that the above mechanisms on dialogue representation can efficiently model the sequential structure of dialogue flow, and offer realtime guidance to a therapist.

Chapter 5 introduces natural language as inductive biases. We studies how to use

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 domains and services. We study three main challenges of using natural language for label representation: schema encoding, supplementary training, and description styles.

Chapter 6 concludes by providing a summary of contributions and a discussion of possible directions of future work.

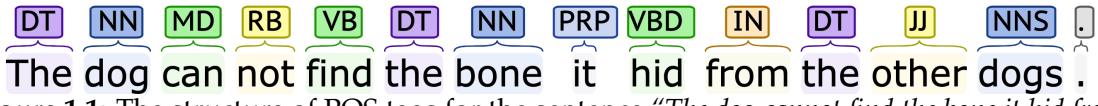

Figure 1.1: The structure of POS tags for the sentence *"The dog cannot find the bone it hid from the other dogs."* This image shows the tag set used in Penn Treebank [58].

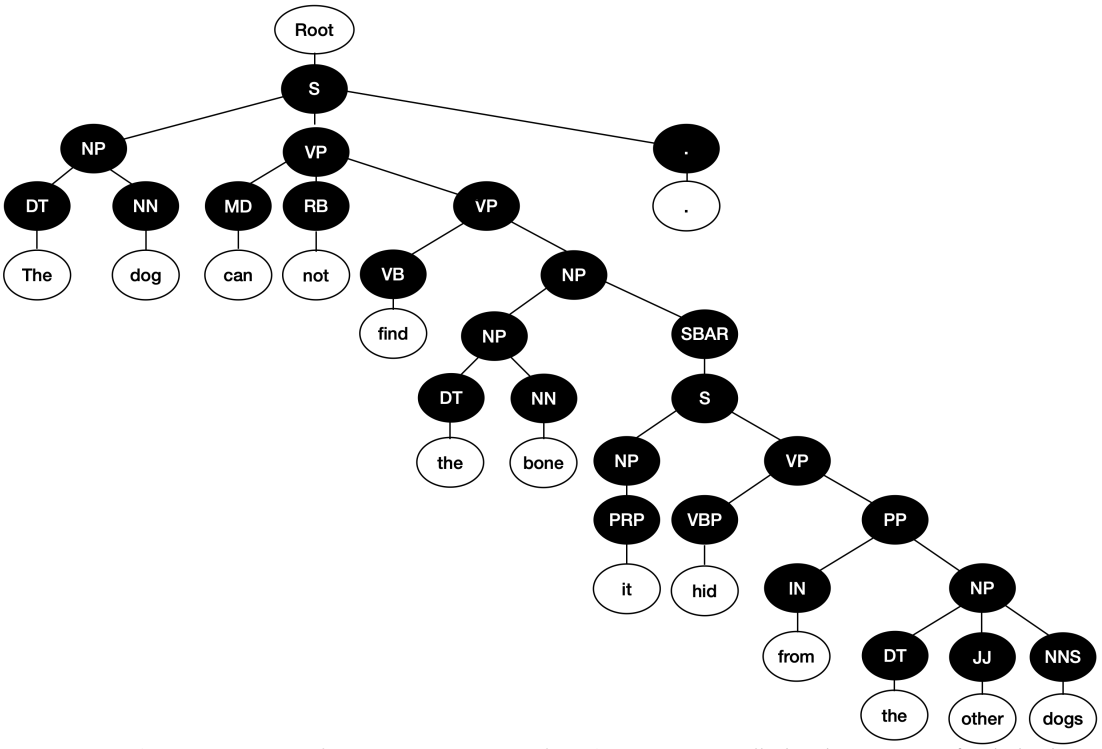

Figure 1.2: The structure of constituent tree for the sentence *"The dog cannot find the bone it hid from the other dogs."*

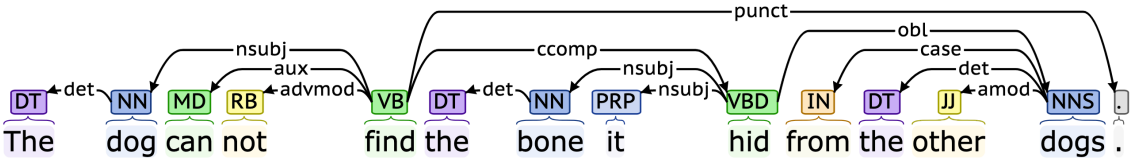

Figure 1.3: The structure of dependency tree for the sentence *"The dog cannot find the bone it hid from the other dogs."*

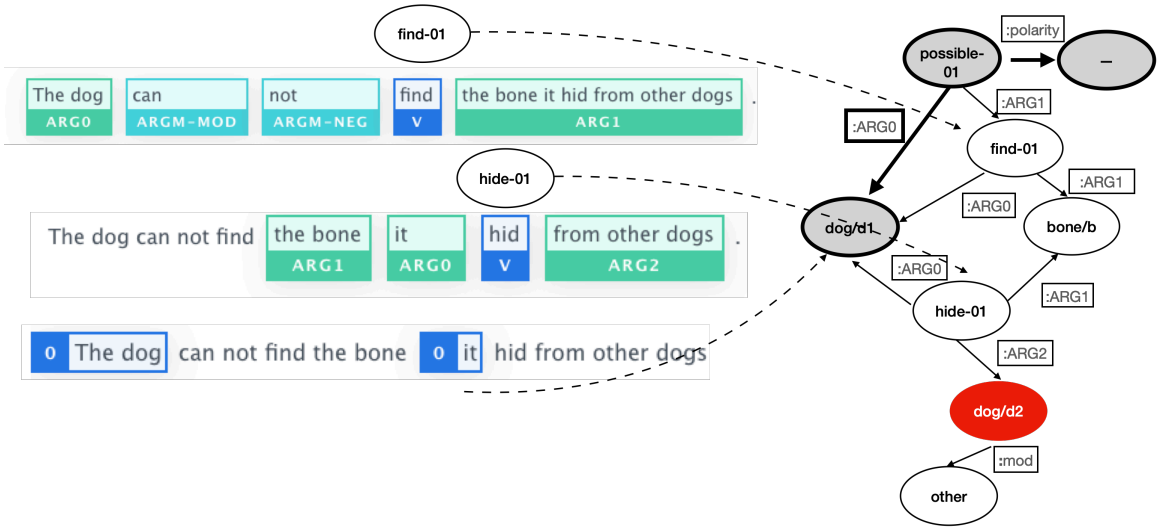

Figure 1.4: The broad coverage meaning representation AMR for the sentence *"The dog cannot find the bone it hid from the other dogs."*. It represents multiple phenomena in a single structure, inclduing the predicate-argument structure and word sense disambiguiation in semantic role labelling, coreference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

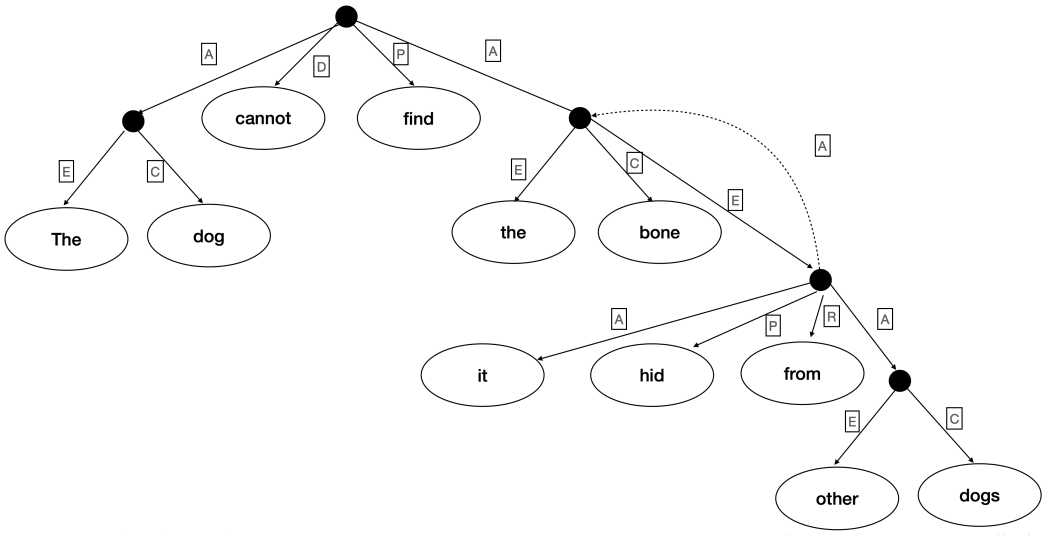

Figure 1.5: The broad coverage meaning representation UCCA for the sentence *"The dog cannot find the bone it hid from the other dogs."*

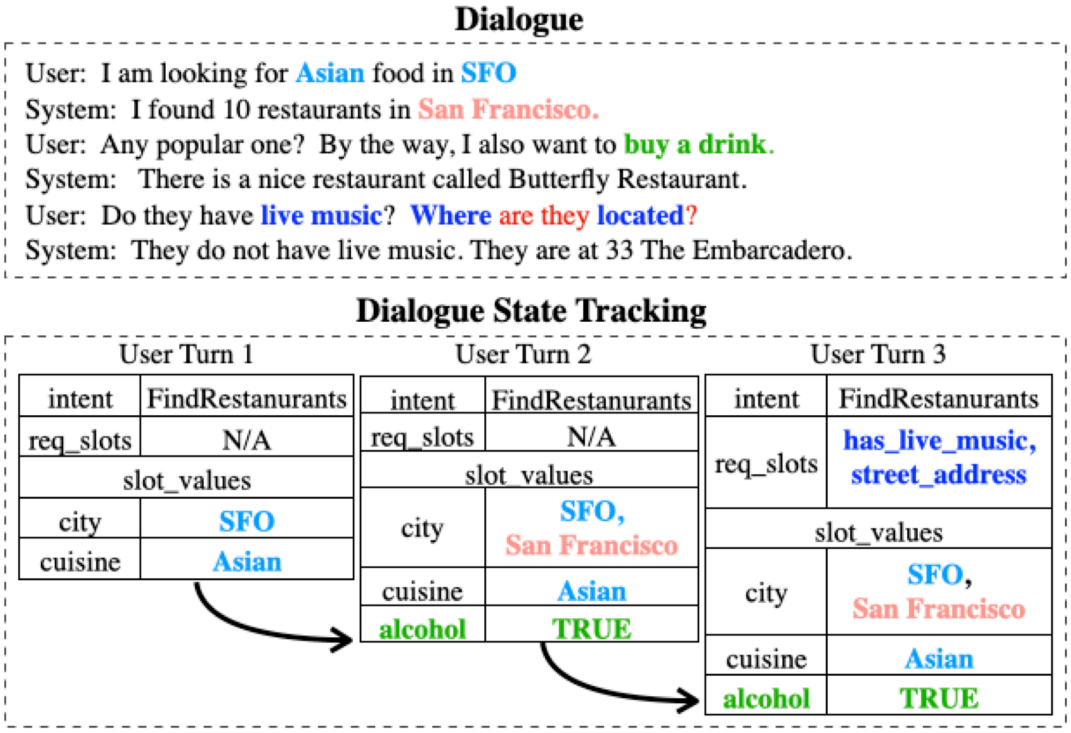

Figure 1.6: An example for dialogue state tracking.

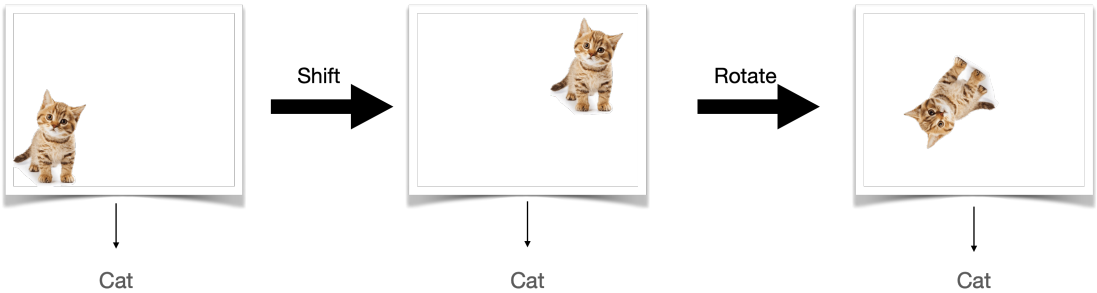

Figure 1.7: In the image classication task, we hope the learned model can still recognize "*cat*" for the unseen image with shifted or rotated cat.

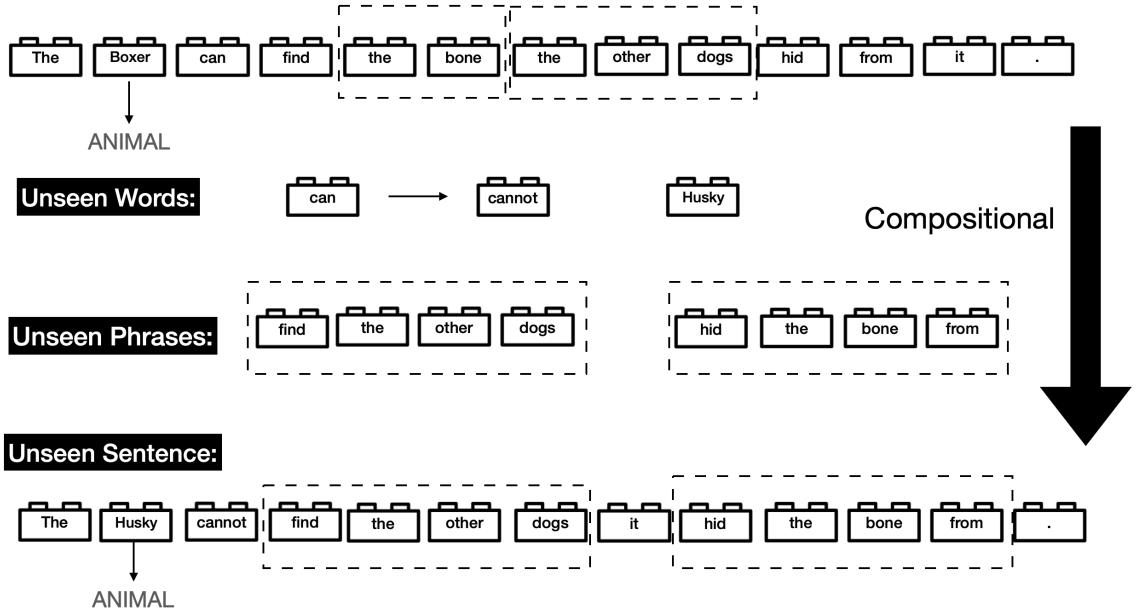

Figure 1.8: In the named entity recognition task, we hope the learned model can still recognize "*dog*" for new word "*Husky*" in unseen context with newly composed words, phases and sentences.

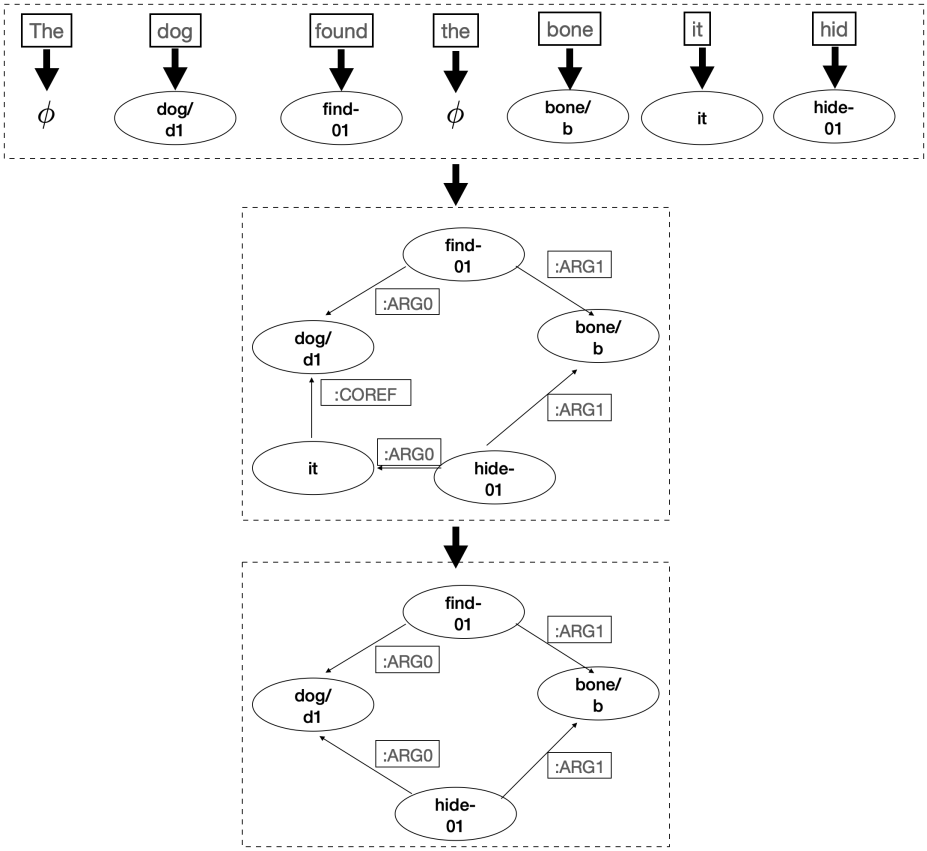

Figure 1.9: Independence Factorization for parsing a new sentence 'The dog found the bone it hid" into an AMR graph'

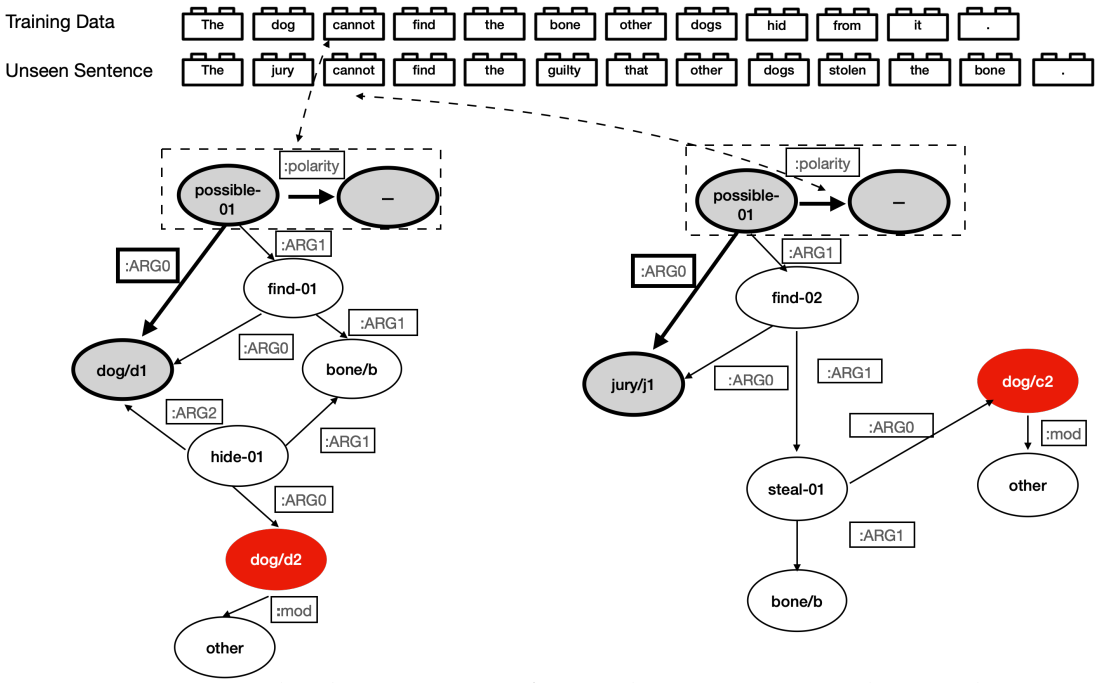

Figure 1.10: Structural Inductive Biases of AMR decomposition, and AMR alignments.

CHAPTER 2

BACKGROUND

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

2.1 Deep Linguistic Structured Prediction and Independent Factorization

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 dissertation, instead of focusing on a single task, we systematically study the representation learning challenges for multiple tasks based on the independent factorization assumption. In this section, We first introduce the recent advances in deep structured prediction with respect to representational formalism (Section 2.1.1) and introduce the independent factorization with factor graphs. Then we briefly summarize the progress on representation learning for natural language (Section 2.1.2) and show why the independent factorization is possible with the contextualized representations. Finally, we also summarize the recent advances on inference (Section 2.1.3) for linguistic structured prediction.

2.1.1 Formulations of Structural Interdependence

Structured prediction refers to machine learning models that learn a target function to predict mulitple interrelated and dependent outputs. For representing the target function, different formulations exists. In this section, we mainly review the recent advances of representations for modeling the structural interdependences.

2.1.1.1 Graphical Models

A graphical model is a probabilistic model using a graph to express 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 independence, but they differ in the set of independences they can encode and the factorization of the distribution that they induce.

2.1.1.2 Constrained Conditional Models

Besides the graphical models to declaratively represent the structural interdependence, constrained conditional models [CCM, 59] are 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 limits, penalizing unlikely assignments. One popular formalism to represent the constraints is to use an integer linear programming (ILP), which has been widely used to constrain learning in many NLP tasks [60]. The declarative linear objective functions, linear constraints, and the availabilities of the off-the-shelf solvers make this formalism very easy to use.

2.1.1.3 Declarative Constraints in Deep Learning

Recently, to inject known hand-crafted constraints between discrete variable assignments in the deep neural networks, one fundamental challange is how to represent the constraints in end-to-end differentiable ways [61]. For example, Li and Srikumar [62] propose to use differentiable fuzzy logic operators to augment the neural networks with boolean logic. Pacheco and Goldwasser [63] introduce a declarative Deep Relational Learning framework (DRAIL) integrating neural representation learners with probabilistic logic. Besides representing the constraints as logic forms, much recent work also studies representing constraints with discrete latent variable models, such as StructVAE for latent tree-structured variables [64, 65]. Our work on latent alignment models also falls into this category.

2.1.1.4 Learning Constraints in Deep Learning

Besides injecting declarative constraints, recent research also learn the constraints in end-to-end ways. Belanger and McCallum [SPEN, 66] 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 [67] was proposed to learn the constrained network for inference, which approximates the cost-augmented inference during training and then fine-tuning for test-time inference.

2.1.1.5 Independent Factorization

In this dissertation, we mainly use the undirected Markov random fileds to represent our independent factorization assumptions. As shown in Figure 2.1, each circle represents a variable, while each rectangle indicates a factor between the input sentence variable x and each decomposed segments of output structures y. The difference between the left and right figure is in the alignment variable a in the center. In the left figure, the shaded circle a means the alignment is explicitly observed after the decomposition. In the right figure, the alignment variable a is not observed.

As discuessed in Section 1.1.3, to apply independent factorization for each task, we need to resolve three main challenges to formulate the above factor graph as $E(x, y) = \sum_{c \in C} E(x, a(y_c), y_c)$, including (1) **Output Decomposition:** Decomposing the output y into a set of independent parts y_c. (2) **Input Decomposition and Alignments Discovery:** Decomposing x and derive the aligned input $x_{a_{y_c}}$ at the index a_{y_c}. (3) **Factor Modeling:** Modelling each y_c and its relevant parts $x_{a_{y_c}}$ to compute the energy score $E(x, a(y_c), y_c)$.

In the following subsection, we show that rapid progress in contextualized representation learning can offer discriminative features for modelling the relative parts of $x_{a_{y_c}}$, thus make the challenge (3) (factor modeling) of independent factorization possible. Then we provide the background knowledge about structures in NLP in Section 2.2, and we briefly show that such prior knowledge about anchoring and compositionality are the main source of inductive biases that guide us to find the decomposition and alignment in the challenge (1) and (2). We extend the detailed analysis about independent factorization in each application chapter.

2.1.2 Neural Representation Learning

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

The key challenge of trying to apply analytical and computational methods to text data are 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 [68], word embedding and neural 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 networks to transformer, and the corresponding pre-trained language models ELMo [69], BERT [70], GPT3 [40], etc.

2.1.2.1 Static Word Embedding

Word embeddings are commonly categorized into two types [71, 72, 73], 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, 74] and GloVe [72] are among the best-known models for the two types respectively. However, they create a single fixed representation for each word, a notable problem with the static word embeddings are that all senses of a polysemous word must share a single vector.

2.1.2.2 Contextualized Representation

To resolve the above issue of static word embedding, sequence encoders, such as LSTM [41], Transformer [44], 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 dynamically depends on the entire sentence. Furthermore, based on the neural sequence encoding architectures, pretraining language models with a large amount of text can create a more powerful contextualized representations [75]. ELMo [69] creates contextualized representations of each token by

concatenating the internal states of a 2-layer biLSTM trained on a bidirectional language modeling task. In contrast, BERT [39] and GPT-2 [76] are bi-directional and uni-directional transformer-based language models respectively. Peters et al. [77] shows that ELMo contextualized representation is more suitable to be used as a fixed word embedding, as also shown in our dissertation on lexical and phrasal anchoring based parsing (Chapter 3) and sentenctial anchoring based MISC code prediction (Chapter 4). While for BERT and GPT-2, finetuning them on downstream task will lead to better performance, which also inspired us on exploiting natural language description to understand each output components (such as intent, slot labels) in task-oriented dialogue (Chapter 5). One assumption behind the independent factorization is that local models with rich features may perform competitively or better than global models also exists in pre-deep learning era. Before the rising of deep learning in 2012, the well known MEMM-based Stanford pos tagger is the state-of-art model at that time. Instead of using a single normalized, global CRF model for the sequence modeling, richer features seem diminish the label biases problems in the MEMM [78, 79]. However, the rich global features required heavy engineering in ten years ago. Recently, according to the NLP-Progress website [1], which tracks the state-of-art models for each task, we found that the state-of-art models for many sequences tagging tasks (such as named entity recognization, part-of-speech tagging) are attention-based models without any CRF layer. In this dissertation, the contextualized representation learned from deep learning is the key to our independent factorization, which helps our decomposed factor models to make local decisions with a set of discriminative and global features, without any heavy feature engineering. Furthermore, large pre-trained language model also offers great power to use natural language in our modeling, such as prompt-based models [80, 81].

2.1.3 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 before

[1] `http://nlpprogress.com/english/named_entity_recognition.html`, visited on July 19, 2022

reducing overfitting. In this way, the neural models only learn a single set of parameters. However, the point estimation does not capture the associated uncertainties [82, 83]. Hence, we care about both MAP and marginal inference in structured prediction research.

2.1.3.1 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 [84] for hidden markov models, CKY for context-tree grammars [85, 86, 87] , Max Spanning Arborescence for spanning tree [88, 89], and so on), and Integer Linear Programming [60, 90, 91]. On the other side, approximate inference methods include various sampling methods [92, 93], search-based methods [94, 95, 96], and Linear Programming Relaxations [97, 98].

2.1.3.2 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 [99, 100], constrative estimation [101], Conditional Random Field [CRF, 102], max-margin training over all candidate targets [103]. For linguistic structured prediction, exact marginal inference methods include forward-backward algorithm for HMM [104], Inside-outside [99], Matrix-Tree Theorem for non-projective dependency structures [105, 106].

In this dissertation, we use variational inference to marginalize out the latent alignment variable in Section 3.2.3. While for the other parts of this dissertation, the assumption of independent factorization simplified the inference into either greedy (Chapter 4 and Chapter 5) or dynamic programming based exact MAP inference (such as the dynamic programming parsing the dependency and constituency structures in Section 3.2.2 and Section 3.3 respectively)

2.2 Symbolic Representations for Natural Language

In Chapter 1, we have listed several lexical, syntactic and semantic structures in NLP. In this section, we will mainly introduce more detailed background on the broad-coverage

semantic presentations (Section 2.2.1) and application-specific representations on dialogue (Section 2.2.2).

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 [107], 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:** We 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*.

2.2.1 Broad-coverage Semantic Representations

For linguistic analysis, structures have been studied from subword-level morphology [108], word-level lexicon semantics [109], to single sentence syntax/semantic representations [110, 111, 112], and multi-sentences discourse analysis [113, 114, 115]. A broad-coverage semantic representation is a general-purpose meaning representation language aiming to represent the multiple phenomena in a single structure for broad-coverage text.

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

This dissertation 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, 116] and Prague Semantic

Dependencies [PSD, 12, 13], Abstract Meaning Representation [AMR, 8]; While for phrasal-anchoring, we study Universal Conceptual Cognitive Annotation [UCCA, 10]. We consider a famous sentence (#20001001 in MRP Corpus) as a running example, which is also the first sentence from WallStreetJournal (WSJ) Corpus from the Penn Treebank [117]: *"Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov.29."* This sentence contains some interesting linguistic phenomes, such as morphology words, person, and date named entities. Taking it as an example, we will introduce the detailed properties for each of the representations. [2]

2.2.1.1 DELPH-IN MRS-Derived Bi-lexical Dependencies

It originates in a manual reannotation, dubbed DeepBank [118], with syntactic-semantic analyses of the LinGO English Resource Grammar [119] in logical forms. These logical forms are often referred to as English Resource Semantics [ERS, 120], and the underlying grammar is rooted in the general linguistic theory of Head-Driven Phrase Structure Grammar [HPSG, 121]. Then Ivanova et al. [116] propose a two-stage version to transform the ERS logical forms into bi-lexical semantic dependency graphs. ERS logical forms are firstly transformed into Elementary Dependency Structures [EDS, 122], then EDS are simplified into pure bi-lexical dependencies, dubbed DELPH-IN MRS Bi-Lexical Dependencies [DM, 116]. As shown in Figure 2.2, graph nodes in DM are anchoring to the surface tokens. Hence, DM is lexical-anchoring. However, the underlying tokens are not 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 [3], but there are some more specialized edge labels too. For example, it uses *compound* to reflect the name "*Pierre Vinken*" as a whole, and it uses BV (bound variable, e.g., the word "*a*") as a reflection of quantification in the underlying logic quantification. In summary, DM is lexical-anchoring, and it captures semantic phenomena including predicate-argument structures, word-sense differentiation, quantification, and scope.

---

[2]The summarization paper in MRP workshop [117] introduces another example-based comparative studies for different meaning representations.

[3]The annotation of predicate-argument 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`

2.2.1.2 Prague Semantic Dependencies

Besides DM, there is another line of research to simplify the richer syntactic-semantic 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, 12, 123] into PSD. The Prague Czech-English Dependency Treebank [PCEDT, 12] 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 the case of coordinate clauses, there are multiple top nodes per sentence. Figure 2.3 shows the PSD representation of our example sentence. One major differences are the role labels and verb frames, they are grounded in a machine-readable valency lexicon [124], and the frame values on verbal nodes indicate specific verbal senses in the lexicon. In a summary, PSD is also lexical-anchoring, and it captures the same semantic contents with DM (including predicate-argument structures, word-sense, quantification and scope), however, with a different formation and different frame lexicon.

2.2.1.3 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 represents the word "*nonexecutive*" with a negation "*(:polarity 0)*" with the concept "*executive*". Furthermore, unlike DM and PSD on predicate-argument representation, AMR extensively uses PropBank framesets [9, 111]. For example, It represents the verb "*join*" using the frame "*join-01*". At the meantime, 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 start with a conjectured alignment between tokens and the graph elements. Most AMR parsers [*e.g.* 48, 125, 126, 127, 128, 129, 130] use either the JAMR aligner [125] or the ISI aligner [131] for this purpose[4] We introduced more details about AMR alignment in Section 3.1.2, and we show that AMR nodes and subgraphs are implicitly anchored to the lexical tokens or entities. In summary, AMR is implicit lexical-anchorin, and it captures more semantic content than DM and PSD. Besides the common predicate-argument structure, word sense, qutification, and scope (by placing "*polarity*"), AMR also represents the lexical decomposion, anaphoric coreference, and entity-linking.

2.2.1.4 Universal Conceptual Cognitive Anotation

Similar to AMR, UCCA is designed to abstract the semantic scheme away from its surface 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). 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 edges and represent the descendant unit's role in forming the semantics of the parent unit. The foundational layer covers the entire text, so 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 fundamental 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*",

[4]Other aligners exist, *e.g.* Chen and Palmer [132] uses dependencies, but raw text alignments are more prevalent.

"*join*" with an incoming edge "*P*" denoting the category "*Process*", which is the primary 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. In sum, UCCA is phrasal-anchoring, and its foundational layer captures predicate-argument structures, anaphoric coreference, representing less semantic content than DM, PSD, and AMR. However, UCCA shows more benefits on cross-lingual, easy annotation, and extensible modularity.

2.2.1.5 Summary

For easy reference, in Table 2.1, we summarize the background of broad-coverage meaning representations for the their anchoring type and captured semantic phenomena. For more detailed comparative studies over the state-of-art semantic representation, please refer to the paper [133].

2.2.2 Application-specific Representation on Dialogue

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

2.2.2.1 Dialogue Act and MISC Codes

In the utterance-level, dialogue acts are designed to represent the function of each utterance in the dialogue. The key insight behind the dialogue 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 [14]. A dialogue act has two main components: a communicative function and a semantic content. Bunt et al. [15] provides an ISO project developing an international standard for annotating dialogue with semantic information, particularly 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, 56, 57] are also proposed to represent the functions of each client and therapist utterance in the psychotherapy dialogue. This part will mainly introduce the MISC Codes for psychotherapy dialogue.

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 MI's high efficacy and success in psychotherapy [134, 135, 136]. However, MI skills take practice to master and require ongoing coaching and feedback to sustain [137]. Given the emphasis on using specific types of linguistic behaviors in MI (*e.g.*, open questions and reflections), fine-grained behavioral coding plays an essential role in MI theory and training.

Motivational Interviewing Skill Codes (MISC, Table 2.2) is a framework for coding MI sessions. It facilitates evaluating therapy sessions via utterance-level labels akin to dialogue acts [138, 139], and are designed to examine therapist and client behavior in a therapy session.[5] Table 2.2 shows the distribution, description and examples of MISC labels for clients and therapists. Each of the MISC labels will be corresponding to a single utterance. Hence, MISC code prediction is a sentence-anchoring task, mainly capturing each utterance's function.

2.2.2.2 Dialog State Tracking

From the simple GUS [17] 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 [16]. 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.

As the dialogue goes on, a dialogue state tracker maintains the current state of the conversation (which includes 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 to the utterance of users.

Figure 1.6 shows a dialogue on restaurant booking service. For each user turn, the dialogue state tracking predicts the corresponding intent, requested slot, and slot values

[5]The original MISC description of Miller et al. [140] 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. [141]; Appendix A gives more details.

for that turn. The intent classification task is to understand what the user is trying to accomplish in this dialogue utterance; thus, the output intent label is anchored to the current user utterance. The requested slot is what the user is asking for more information, while the slot filling task is to extract the particular slots and fillers that the user offered to the system for more detailed arguments of their intent. The requested slots and slot-filling tasks can depend on any relevant tokens or phrases in the user utterance; thus, the anchoring is mixed.

2.2.2.3 Conversational Semantic Representations

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 [19]. Hence, the previous intent-slot dialog state representation has poor compositionality to represent complex conversational requests, such multiple intents in the same utterance, and nested intent slot structures.

Recently, conversational parsing has attracted much attention to represent the dialogue state in a more compositional way, such as the the hierarchical tree structure in the Task-Oriented dialogue Parsing [TOP, 54, 142], [TreeDST, 143], and [Dataflow, 144]. In summary, those conversation semantic representation offers richer intent slot compositions, and support complex conversational linguistic phenomena, such as dialogue state revision and recovering.

In this dissertation, we study the structures of single sentence representation TOP for dialogue representations. Figure 2.6 shows two examples of the nested structures of TOP structures. All intents and slots are non-terminal 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.

2.2.2.4 Summary of Application-specific Representations

Table 2.3 summarized the application-specific representations studied in our dissertation. For the application-specific representations on dialogue, we first introduced sentence-level

anchoring representation: dialogue acts, and we show a specific type of dialogue act for psychotherapy dialogue called MISC. Then we introduced the frame-based dialog state tracking, and we showed that intent classification is also sentence-anchoring tasks, while requested slot and slot filling tasks can be considered as mixed anchoring. Finally, to resolve the limitations on non-recursive intent and slot tagging, we briefly introduced a set of conversational semantic parsing representations. They offer richer intent slot compositions and support complex conversational linguistic phenomena, such as dialogue state revision and recovery. In this dissertation, we mainly focused on one of the tree-structural conversational dialogue representation called TOP, which shares the same phrasal anchoring due to the similar structure with constituent tree.

2.3 Chapter Summary

In this chapter, we first summarized the recent advances in deep structured prediction for representational formalism, learning and inference, respectively. For representational formalism, we introduce graphic model to represent the structural interdependence, and represent the main assumption of the independent factorization as two factor graphs in Figure 2.1. Then we overviewed the development of representation learning methods for natural language, from feature selection to deep learning based on contextualized representation learning methods. We also reviewed the recent advances of MAP inference and Marginal Inference in deep learning.

The second part of this section introduces the background of symbolic representation for natural language, including the broad-coverage meaning representations and application-specific representations. Instead of studying structured prediction for each linguistic structured prediction tasks separately, based on independent factorization, we firstly categorize those symbolic representations according to the anchoring types, summarized in Section 2.2.1.5 and Section 2.2.2.4. Then based on the background, in the following chapters, we study the detailed strategies for independent factorization by exploiting the structural inductive biases and natural language as inductive biases.

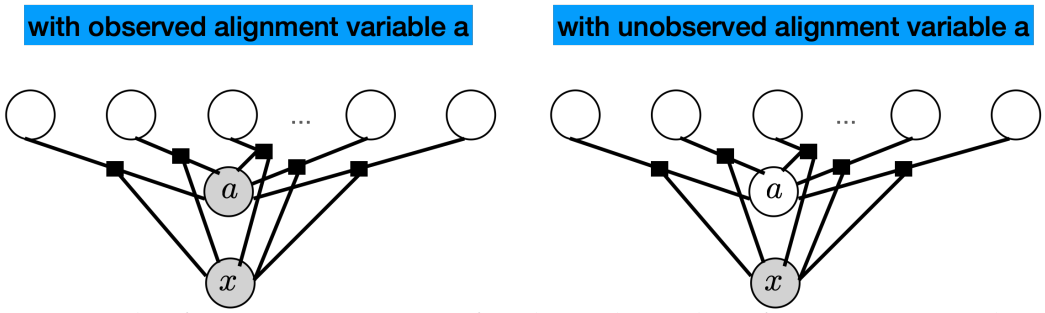

Figure 2.1: The factor representation for the independent factorization used in our dissertation.

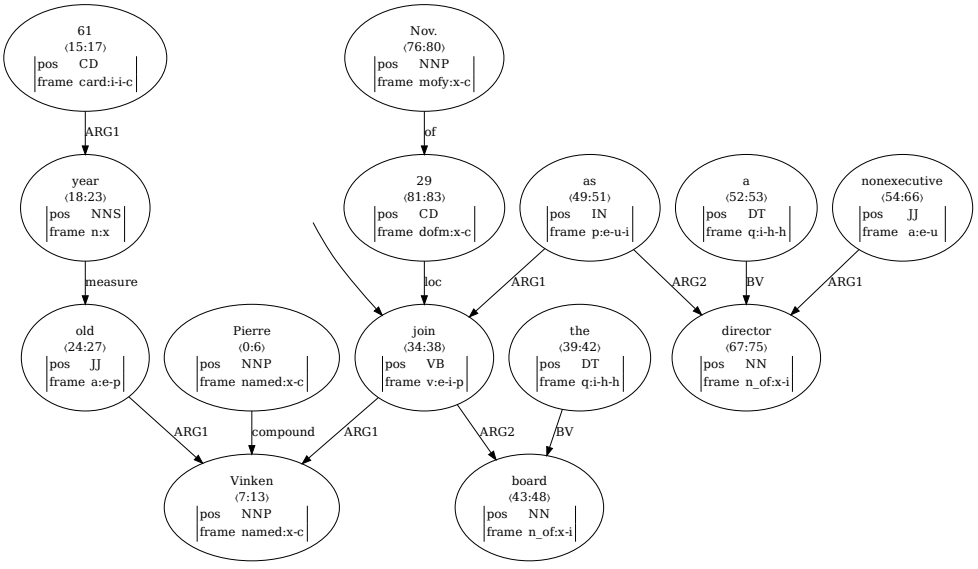

Figure 2.2: The DM representation for the sentence #20001001

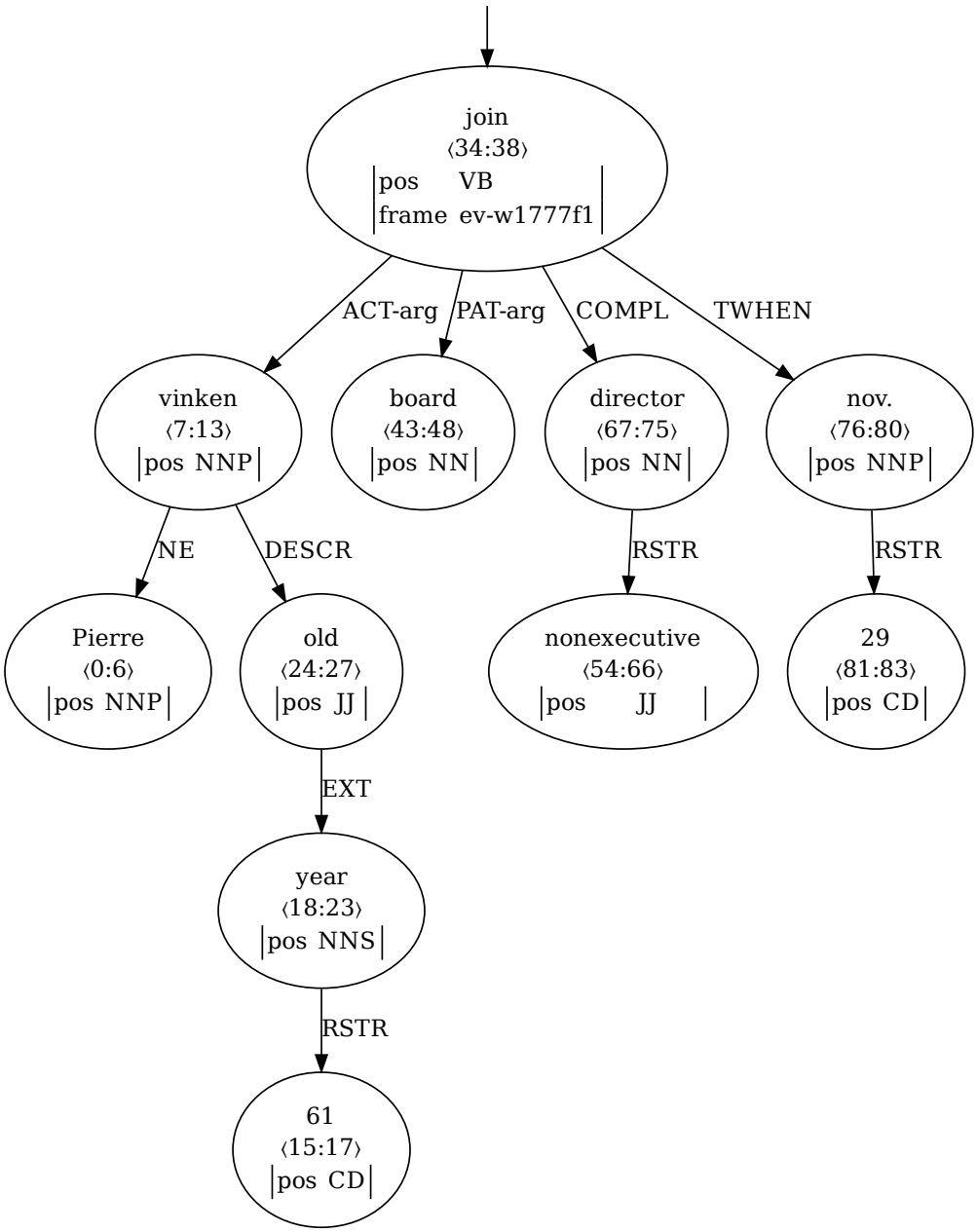

Figure 2.3: The PSD representation for the sentence #20001001

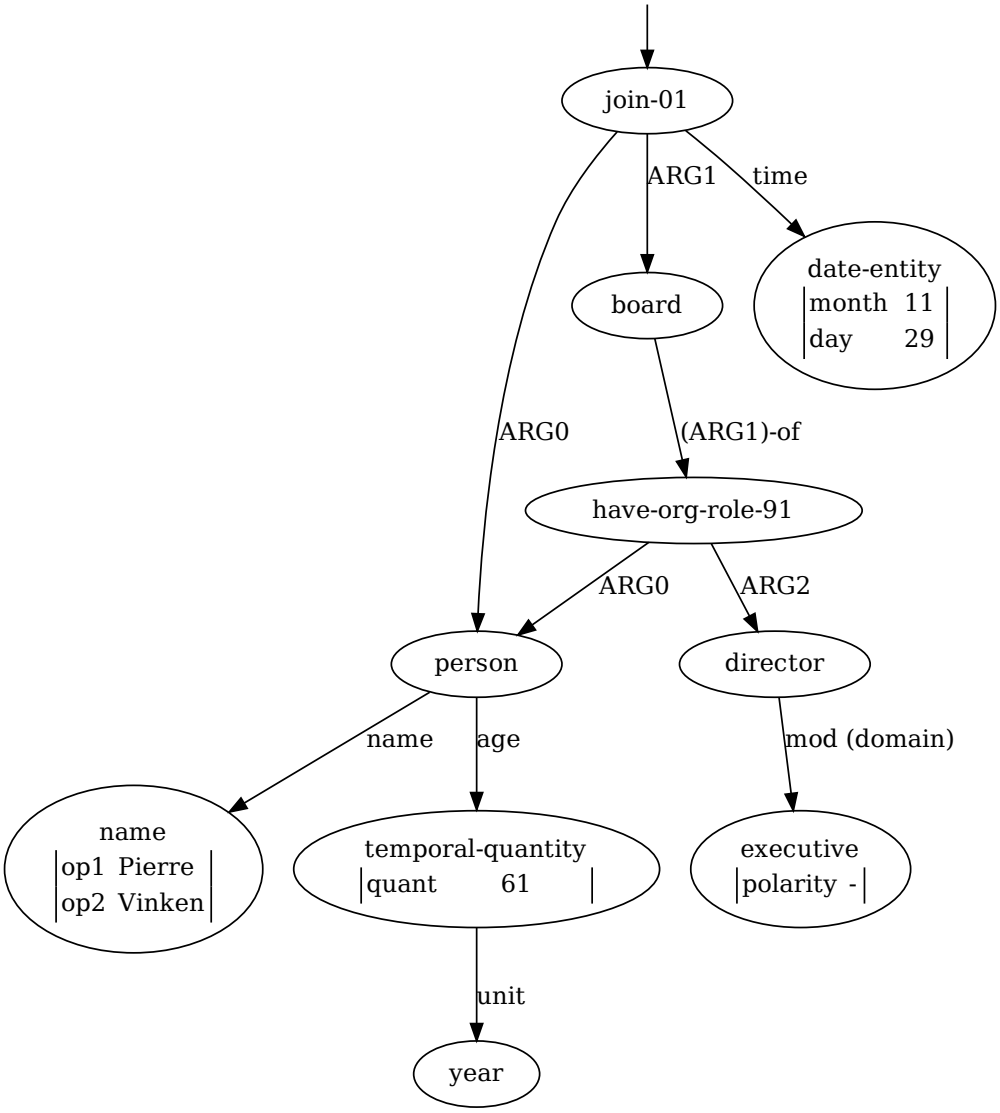

Figure 2.4: The AMR representation for the sentence #20001001

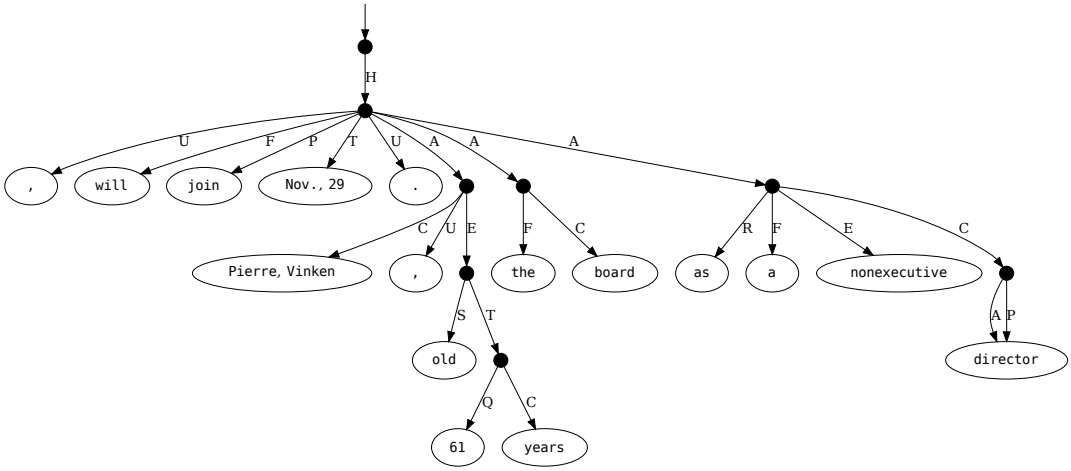

Figure 2.5: The UCCA representation for the sentence #20001001

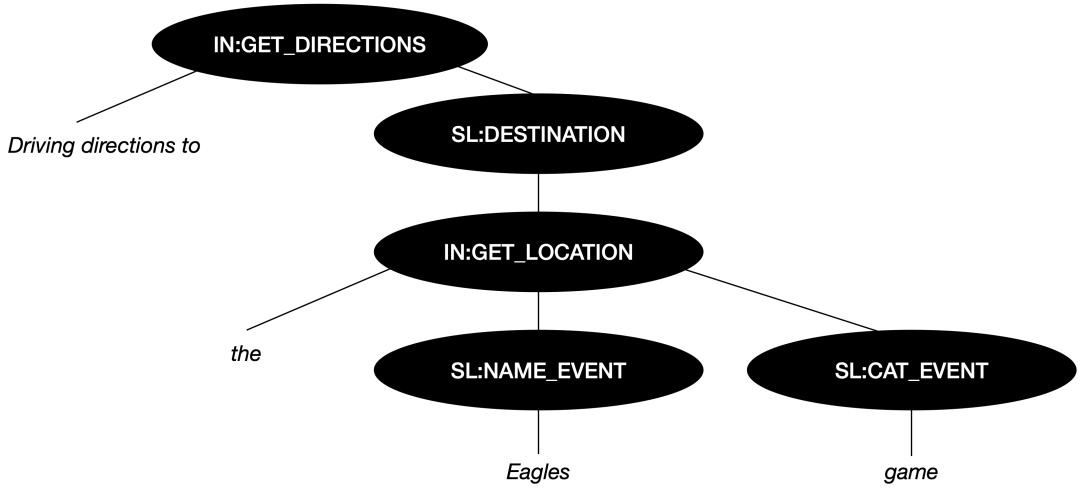

Figure 2.6: A TOP example for conversational semantic parsing (borrowed from the original paper [54])

Table 2.1: The summary of broad-coverage meaning representation, with respect to their anchoring type and captured semantic contents.

	DM	**PSD**	**AMR**	**UCCA**
Anchoring Type	Explicit Lexical	Explicit Lexical	Implicit Lexical	Phrasal
Predicate-Argument	Yes	Yes	Yes	Yes
Word Sense	Yes	Yes	Yes	No
Qualification and Scope	Partial	No	Yes	No
Presupposition/Focus	No	No	Yes	No
Lexical Decomposition	No	No	Yes	No
Anaphoric Coreference	No	No	Partial	Partial
Grounding	No	No	Yes	No

Table 2.2: Distribution, description and examples of MISC labels.

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 2.3: The summarization of the application-specific representations studied in this dissertation.

	MISC	**Dialogue State Tracking**	**TOP**
Application	Psychotherapy	Task Oriented	Task Orientied
Structures	Sequence Labelling	Intent/Requested Slot/Slot Filling	Tree Parsing
Anchoring Type	Sentential	Mixed	Phrasal

CHAPTER 3

STRUCTURAL INDUCTIVE BIASES FOR PARSING GRAPH-BASED REPRESENTATIONS

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 [145, 146, 147, 148]. Moreover, the difference 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 [133]. In some application scenarios, tasks-specific formal representations such as database queries, the arithmetic formula has 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. In this chapter, we mainly studied parsing graph-based symbolic representations that inspired by computational semantics, including four broad-coverage meaning representations (DM [116], PSD [12, 13], AMR Banarescu et al. [8], UCCA [10]) and an application-specific representation TOP [54].[1].

Anchoring is crucial in graph-based representation parsing. Training a statistical parser typically starts with a conjectured alignment between tokens (or 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 [48, 125, 131, 132, 149, 150], we propose a uniform handling of three meaning representations (DM, PSD, and AMR) into a new group referred to as the **lexical-anchoring** representations. It supports

[1]The main content of this chapter is published in our paper Cao et al. [50]

both explicit and implicit anchoring of semantic concepts to tokens. We also studied the parsing on the other two symbolic representations (UCCA and TOP), belonging to the group of **phrasal-anchoring** representations 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)[2]. By leveraging the linguistic knowledge about the anchoring analysis, we proposed two separate models for each anchoring type based on independent factorization. For lexical anchoring, 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 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 TOP. Our post-evaluation submission for MRP 2019 task ranked 5th on UCCA. Furthermore, we show that the same unified CKY-based model can be easily used to parse an application-specific dialogue representation TOP, which outperforms several baselines in TOP parsing. Most parts of this chaper is published in our paper [50].

3.1 Related Work and Anchoring Analysis

In this chapter, we mainly study the parsing of four meaning representations (DM, PSD, AMR, and UCCA) and an application-specific representation TOP on dialogue parsing. We have mentioned the anchoring type for each representation according to their brief introduction in Section 2.2.1. In this section, we will first revisit the anchoring analysis for them, especially we show the detailed evidence that AMR is implicit lexical-anchoring. Then, we group the five representations into two groups: Lexical Anchoring (DM, PSD, AMR) and Phrasal Anchoring (UCCA and TOP).

3.1.1 Flavors of Meaning Representation

The 2019 Conference on Computational Language Learning (CoNLL) hosted a shared task on Cross-Framework Meaning Representation Parsing [MRP 2019, 107], which encourage participants in building a parser for five different meaning representations in three

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

distinct flavors. `Flavor-0` includes the DELPH-IN MRS Bi-lexical Dependencies [DM, 11] and Prague Semantic Dependencies [PSD, 12, 13]. Both frameworks under this representation have a syntactic backbone (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. Hence, `Flavor-0` is actually explicit **lexical-anchoring**. Hence, `Flavor-0` is lexical-anchoring in our dissertation.

`Flavor-1` includes Elementary Dependency Structures [EDS, 151] and Universal Conceptual Cognitive Annotation framework [UCCA, 53], which shows an explicit, many-to-many anchoring of semantic concepts onto sub-strings of the underlying sentence. In this dissertation, we consider UCCA with another application-specific symbolic representation TOP, where the underlying sub-strings mainly form a tree-like structure[3]. We grouped UCCA and TOP as **phrasal-anchoring**.

Finally, `Flavor-2` includes Abstract Meaning Representation [AMR, 152], which is designed to abstract the meaning representation away from its surface token. But it leaves open the question of how these are derived. In the following part, we mainly analyze the detailed anchoring analysis of AMR.

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

3.1.2.1 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 [131] first linearizes the AMR graph into a sequence, and then use IBM word alignment model [153] 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

[3]We leave the parsing of EDS as future work

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 [125] 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.* 48, 125, 126, 127, 128, 129, 130]. 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 [48] 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.

3.1.2.2 AMR-to-Dependency Alignments

Chen and Palmer [132] first tries to align an AMR graph with a syntactic dependency tree. Szubert et al. [149] 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 [150] 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 [154]. However, whether the attention weights can be explained as AMR alignments needs more investigation in future.

3.1.2.3 Summary of Anchoring Analysis

According to the above factorization analysis, we group the broad-coverage meaning representations (DM, PSD, AMR, UCCA) and the conversational semantic representation (TOP) into the following three categories.

- **Explicit lexical-anchoring: DM and PSD.** As discussed in Section 2.2.1.1 and Section 2.2.1.2, DM and PSD aim to represent all the semantic dependencies between words, fully covering the sentence. For each node in the graph, there will be an explicit words and multiword expressions aligning to it. Hence, we call such kind of lexical anchoring explicit lexical-anchoring.

- **Implicit lexical-anchoring: AMR.** AMR tries to abstract the meaning representation away from the surface token. The absence of explicit anchoring can present difficulties for parsing. However, through extensive anchoring analysis on AMR alignments, we show that AMR nodes can be implicitly aligned to the lexical tokens or special entities in a sentence. Hence, we mainly consider the lexical-level input decomposition for AMR.

- **Phrasal-anchoring: UCCA and TOP.** According to the background of UCCA Section 2.2.1.4, UCCA non-terminal nodes are aligned to phrases in a sentence. Hence, we need to consider phrase-level input decomposition for UCCA.

3.2 Lexical-Anchoring: Latent Alignment Model for Graph-Based Parsing

In this section, we will first introduce the two-stage framework for parsing the DM, PSD, and AMR graphs (Section 3.2.2). Then we resolve the alignment problem with a latent alignment model (Section 3.2.3)

According to the linguistic analysis on anchoring, we have shown that DM, PSD and AMR belong to the lexical-anchoring, which indicates that their nodes in the output graph are explicitly or implicitly anchored to the lexical units of the corresponding input sentence. Before showing the unified design of independent factorization, let us introduce the fundamental concepts and notations.

We refer to words in a sentence as $x = (x_0; x_1; ...x_i; ...; x_n)$, where n is the sentence length. For all the graph-based representations in our lexical-anchoring category, we decompose the whole graph into nodes and edges. We denotes the labelled nodes (concepts) as $C = \{c_i \mid i \in [0, m]\}$, where m is the number of concepts. While the labelled edges (relations) between the m concepts are denoted as $R = \{r_{ij} \mid i \in [0, m], j \in [0, m]\}$. r_{ij} means the directional relation from the node i to the node j. We use $r_{ij} = \Phi$ to indicate no edge from the node i to the node j.

In this part, based on the above notations, we introduce how to do independent factorization on lexical-anchoring by addressing the three main chellenges in **Output Decomposition** (Section 3.2.1.1), **Input Decomposition and Alignments Discovery** (Section 3.2.1.2), and **Factor Modelling** (Two-stage Parsing (Section 3.2.2 and Latent Alignment Model Section 3.2.3)

3.2.1 Independent Factorization on Lexical-Anchoring

In the following, we focus on the first two main challenges in independent factorization: **output decomposition** and **input decomposition** on lexical-anchoring. We leave the **factor modelling** into the next part. We take a more complicated AMR graph in Figure 2.4 as an example, for the sentence, *Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov.29.* We introduce the details of independent factorization for AMR and other lexical-anchoring representations.

3.2.1.1 Output Decomposition

For the lexical-anchoring graph-based representations (DM, PSD, AMR), the target graph can be decomposed into independent nodes or subgraphs. For DM and PSD, each node in the target graph is strictly aligned to the corresponding token. Hence, we simply decompose the DM and PSD graph by nodes. However, when we introduce the necessary of structural inductive biases for independent factorization (Section 1.1.3.3), the AMR parsing running examples shows that we need to handle those special entities in AMR graph. From the training data alone, we cannot easily tell how to segment the AMR graphs. However, according to the annotation guideline of AMR and previous work on subgraph templating [46] or abstract concept label [48], the main intuition of grouping is to ensure that concepts are rarely lexically triggered (e.g., "*person*" and "have-org-role-91"

in Figure 2.4 get grouped together with lexically triggered nodes (e.g., "*Pierre Vinken*", and "director"). Finally, we adopt the templates proposed by Corro and Titov [65] including: "*thing*" (e.g., "*(opine-01 :ARG-01 thing)*" for "*opinion*"), "*person*" (e.g., "*(play-01 :ARG1 person)*" for "*player*"), "*more*", "*most*" (e.g., "*(have-degree-91 :ARG2 good-02) :ARG3 more)*" for "*better*"), "*date-entity*" (e.g.,"*(date-entity :month 2)*" for "*February*"), and all the special quantity entities, "*xx-quantity*" (e.g., "*(monetary-quantity :quant 20 :unit dollar)*" for "*$20*"). [4]

AMR output decomposition will be the rectangled segments shown in Figure 3.1.

3.2.1.2 Input Decomposition and Alignments Discovery

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 implicit 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, and 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.

The more difficult challenge exists in how to align the decomposed input with the decomposed output. According to the previous anchoring analysis, the training data of DM and PSD naturally contains the anchoring information, while the AMR training data doesn't offer the alignments. Hence, for AMR, we need to an extra model to resolve the alignments discovery problem. According to the AMR annotation guideline, a strong inductive bias about the alignment is that *each token or expressive is not overlapping and is exclusively aligned to the output decompositions.* We will introduce the details of the latent alignment model in Section 3.2.3.

3.2.2 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

[4] Please refer to the paper of Lyu and Titov [150] for more details about the recategorization.

loss (NLL), with independence assumption between each node and edge, we decompose it into node- and edge-identification pipelines.

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

Hence, we train a joint model to maximize the above probability for both node identification $P(c_i \mid h_{a_i})$ (Section 3.2.2.1) and edge identification $P(r_{ij} \mid h_{a_i,c_i,h_{a_j},c_j})$ (Section 3.2.2.2). 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. In DM, PSD, and AMR, every token will only be aligned once. We know the exact alignment information in DM and PSD, while we don't have the alignment information for AMR for training. Hence, we introduce the latent alignment model in Section 3.2.3 to resolve the learning with discrete latent alignment variable. Figure 3.2 summerize the unifed two-stage graph based parsing framework. In the following subsections, we will explain the framework in more details.

3.2.2.1 Node Identification

The stage of **node identification** predicts a concept c given a word. A concept can be either "Φ" (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 propose to use copy mechanism [155] in our neural network architecture to decide whether to copy the deterministic label given a word or directly estimate a classification probability from a fixed label set. As the first stage shown in Figure 3.2, each token will output its corresponding node. Here, the two article "*The*", "*the*" and final period will produce "Φ"

node. While other tokens will copy themself and transform to the corresponding lemma or frames.

3.2.2.2 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 [156]. As shown in Figure 3.2

3.2.2.3 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* (MSCG) from all the relations. We use MSCG [125] to greedily select the most valuable edges from the identified nodes and their relations connecting them. As shown in Figure 3.2, an input sentence goes through preprocessing, node identification, edge identification, root identification, and MSCG to generate a final connected graph as structured output.

3.2.3 Latent Alignment Model

As the two-stage probablistic model shown in Equation 3.1, 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 $P(a|a \neq a^*) = 0.0$. Hence, in this case, with known alignment information, we don't need to worry the intractable marginalization problem. Further more, after reducing the latent alignment variable, we can easily learn the parameters for models of node identifiaction and edge identification.

- **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 [150]. In the following section, we introduce the details of latent alignment model via variational inference (Section 3.2.3.1), Perturb-and-Max (Section 3.2.3.2), Gumbel-Sinkhorn networks (Section 3.2.3.3).

Hence, with the alignment models for both explicit and implicit alignments, our unified two-stage parsing framework can support the parsing on DM, PSD and AMR.

3.2.3.1 Variational Inference

First, as shown in Equation 3.2, latent alignment prior $P(a_)$ is unknown for the two-stage parsing. We want estimate the latent alignment model to ensure that it is consistent with the obversed w, C, and R. Furthermore, we also need to marginal it out and jointly learn the parameters θ and Φ to generate the concepts and relations respectively. Hence, we apply variational inferenc to reduce the marginal likelihood into Evidence Lower Bound (ELBO) [157].

$$
\begin{aligned}
&\log(P(C, R \mid w)) \\
&= \log\left(\sum_{a} P(a) P(C \mid w, a) P(R \mid w, a, C)\right) \\
&\geq E_Q[\log(P_\theta(C \mid w, a) P_\Phi(R \mid w, a, C))] - D_{KL}(Q_\Psi(a \mid C, R, w) \mid\mid P(a)) \\
&= \sum_{a} Q_\Psi(a \mid C, R, w) \log(P_\theta(C \mid w, a) P_\Phi(R \mid w, a, C))] \\
&- D_{KL}(Q_\Psi(a \mid C, R, w) \mid\mid P(a))
\end{aligned}
\tag{3.2}
$$

where $Q_\Psi(a|c, R, w)$ is the alignment model, parameterized by a nerual network Ψ to approximate the intractable posterior alignment model $P_{\theta,\Phi}(a \mid C, R, w)$. The second term D_{KL} is the Kullback-Leibler divergence [158], measuring the difference between the approximated posterior and the true posterior. Please refer to the original variational auto-encoder paper [157] for more details.

3.2.3.2 Perturb-and-Map

To jointly learn the models according the Equation 3.2, the posterior alignment model Q can be computed as shown in Equation 3.3.

$$
Q_\Psi(a \mid c, R, w) = \frac{\exp(\sum_{i=1}^{n} \phi(g_i, h_{a_i}))}{Z_\Psi(c, w)} \tag{3.3}
$$

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. However, the denominator Z_Ψ is an intratable partition fuction for the global normalization, which has $n!$ combinations. Perturb-and-Max(MAP) techinique [51] can help this intratable sampling with two steps: (1) perturbing the alignment score ($\phi(g_i, h_{a_i})$) with an independent noise. (2) compute the *argmax* to sample the discrete alignment variable a.

3.2.3.3 Gumbel Sinkhorn

Finally, the remaining challenge lies in the discrete *argmax* in the second step of Perturb-and-Max, which is not differtiable. Mena et al. [52] resolve this issue with a simple differentiable operator called Gumbel-Sinkhorn operator. As the exlcusive alignment assumption when we introduce the input decomposition of lexical-anchoring parsing (Section 3.2.1.2), the *argmax* over exclusive alignment can be relaxiated into a differentiable gumbel softmax over $n * n$ matrix scores. For further details of the estimating, we refer the reader to the original paper [52, 150].

3.3 Phrasal-Anchoring: Minimal Span-Based CKY Parsing

Let us now see our phrasal-anchoring parser for UCCA and TOP. We introduce the transformation (Section 3.3.1) used to reduce UCCA and TOP parsing into a unified constituent tree parsing task. Then based on the above transformation, we introduce how to do independent factorization on phrasal-anchoring by addressing the three main challenges in **Output Decomposition** (Section 3.3.2.1), **Input Decomposition and Alignments Discovery** (Section 3.3.2.2), and **Factor Modelling**, including CKY parsing (Section 3.3.3) and span representation (Section 3.3.4)

3.3.1 Graph to Constituent Tree Transformation

Nodes in UCCA do not have node labels or node properties, but all the nodes are anchored to the spans of the underlying sentence. However, to have a unifed representation with previous lexical-anchoring, we do the following transformation to form a constituent tree for UCCA, where every node has a label, which is suitable for the independent facrtorization to map each input decompositions into the output nodes.

3.3.1.1 UCCA to Consistuent Tree Transformation

We propose to transform a graph into a constituent tree structure for parsing, which is also used in recent work [159]. Figure 3.3 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, to simplify the model, 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.

3.3.1.2 TOP to Consistuent Tree

For the hierarchical dialog representation TOP, we also can tranform it to a constituent tree structure. Figure 3.4 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.

3.3.2 Independent Factorization on Phrasal-Anchoring

In the following, we focus on the first two main challenge in independent factorization: **output decomposition** and **input decomposition** on phrasal-anchoring. We leave the **factor modelling** into the next part. We consider the same sentence example "Pierre Vinken, 61 years old, will join the board as a nonexecutive director Nov.29." and its corresponding UCCA graph as shown in Figure 2.5. We introduce the details of independent factorization for UCCA and other pharsal-anchoring representations.

3.3.2.1 Output Decomposition

After the above transformation, as shown in the Figure 3.5 the labelled 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 dashed line to the node "*:A*", which is the second parent of node "*director*"), the majority of the UCCA graphs are tree-like structures. In this disseration, because the remote edge are rare, we ignore all the exception to simplfy the modeling. According to the position as well as the anchoring style, nodes in UCCA can naturally 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.

Hence, it is natural that we can decompose the transformed UCCA tree into two set of nodes, both of them can be produced from the corresponding lexical units or phrases. However, more challenges exists on how to generate the input segments that can produce those output decompostions.

3.3.2.2 Input Decomposition and Alignments Discovery

Different from the lexical anchoring without overlapping, UCCA may align to larger overlapped word spans which involves syntactic or semantic pharsal structure. Further mode, the phrasal decomposition and the alignment model is provided in training data. However, we have no idea on how to factorize the input during inference.

In lexical anchoring, the input decomposition are handled with a rule-based preprocessing step. However, there is no clear rules how to decompose a sentence into a structure

similar with consistuent tree structure. Luckily, we have many previous studies on how to parse a sentence into a constituent tree, which inspired us that we can model the input decomposition jointly with the factor modelling part. We leave more details of the model for joint input decomposition and factor modeling in Section 3.3.3.

3.3.3 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 [159], 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 3.3. 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 [160].

3.3.3.1 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 3.4. However, this assumption does not hold for UCCA. Please see more error analysis in Section 3.4.5

$$\begin{aligned} T^* &= \arg\max_T s(T) \\ s(T) &= \sum_{(i,j,l)\in T} s(i,j,l) \end{aligned} \tag{3.4}$$

3.3.3.2 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 3.5.

$$\begin{aligned} s_{\text{best}}(i,i+1) &= \max_l s(i,i+1,l) \\ s_{\text{best}}(i,j) &= \max_l s(i,j,l) \\ &\quad + \max_k [s_{\text{best}}(i,k) + s_{\text{best}}(k,j)] \end{aligned} \tag{3.5}$$

3.3.4 Span Encoding

For each span (i, j), we represent the span encoding vector $v_{(i,j)} = [\vec{y}_j - \vec{y}_i] \oplus [\overleftarrow{y}_{j+1} - \overleftarrow{y}_{i+1}]$. Here, $\oplus$ 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 Section 3.4.3.

3.4 Experiments and Results

This section will first introduce the dataset and the evaluation used in our experimental study (Section 3.4.1). Then we summarize the implementation details when using our proposed frameworks for each parsing task (Section 3.4.2), with respect to the modeling on root, node, and edge. Then we introduce the model setup (Section 3.4.3), including sentence encoders, embedding usage, and training strategies. Finally, we demonstrate the experimental results with both the evaluation results (Section 3.4.4) and detailed error analysis (Section 3.4.5).

3.4.1 Dataset and Evaluation

For DM, PSD, 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 [107],

3.4.2 Summary of Implementation

We summarize our implementation for lexical-anchoring in Table 3.1, and phrasal-anchoring in Table 3.2. As we mentioned in the previous sections, we use latent-alignment graph-based parsing for lexical-anchoring rerepsentations (DM, PSD, AMR), and use CKY-based constituent parsing phrasal anchoring rerepsentations (UCCA, TOP). This section gives information about various decision for our models.

3.4.2.1 Root

The first row "*Root*" 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 predict one and only one root node with a N (N is size of already identified nodes) way classifier, and then fix this with a post-processing strategy. When our model predicts one node as the root 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 representations 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, 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. 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 representation, 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 predict the node label based on each 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 representations: 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. We ignored all the discontinuous span and remote edges in UCCA.

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 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 [160], 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 [161], 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 in MRP'2019 shared task, we submitted the unified lexical anchoring parser for DM, PSD and AMR, 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. For the results of TOP Parsing, we use a seperate table for it.

3.4.4.1 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. However, the 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. Our submitted parser ranked 6th on PSD and 7th on DM. Extensive error analysis shows that our parser perform worse on the root and edge prediction. Please refer to more details in Section 3.4.5.1.

3.4.4.2 Official Results on Phrasal Anchoring (UCCA)

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 [162] into our model, it can further improve almost 3 points on it. However, it still performs 4 points worse than the top 1 model from [163], which realize stack LSTM with more advanced contextualized word embeddings BERT [39]. We also conduct extensive error analysis in Section 3.4.5.2.

3.4.4.3 Results on Phrasal Anchoring (TOP)

Table 3.5 shows the results on TOP parsing. We use our CKY model skeleton but with different underlying contextualized span representation from ELMo, Roberta, and SpanBERT. Below the line are the baseline models for TOP parsing. Our plain model with glove-based transformer model can outperform all the previous baseline models. Contualized presentation from pretrained language model can boost the performance further.

3.4.5 Error Breakdown

Table 3.6, 3.7, 3.8 and 3.9 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 [164] and ERG [165].

3.4.5.1 Error Analysis on Lexical Anchoring

As shown in Table 3.6, our AMR parser is good at predicting node properties and consistently perform better than other models in all subcomponent, except for top prediction. 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 [70] may also boost the performance of our paser in future.

3.4.5.2 Error Analysis on Phrasal Anchoring

Our UCCA parser in post-evaluation ranked 5th according to the original official evaluation results. According to Table 3.9, 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 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, where each output node can implict be aligned to a token or entity in the input sentence. Thus we propose to regroup five representations as two groups: lexical anchoring (DM,PSD,AMR) and phrasal anchoring (UCCA,TOP). Furthermore, based on the assumption of independent factorization and the structural inductive biases about different anchoring-types, we proposed two parsing framework to support lexical and phrasal anchoring respectively. For lexical anchoring, we suggest to parse DM, PSD, and AMR in a unified parsing framework based on latent alignment model, which supports both **explicit lexical-anchoring** and **implicit lexical anchoring**. Our submission ranked top 1 in AMR sub-task, ranked 6th and 7th in PSD and DM tasks. For phrasal anchoring, by reducing UCCA and TOP graph into constituent tree structures, and we use the same span-based CKY parser to predict their diverse tree structures. Our phrasal-anchoring parser ranked 5th in UCCA official post-evaluation stage, and outperform serveral baselines on TOP parsing.

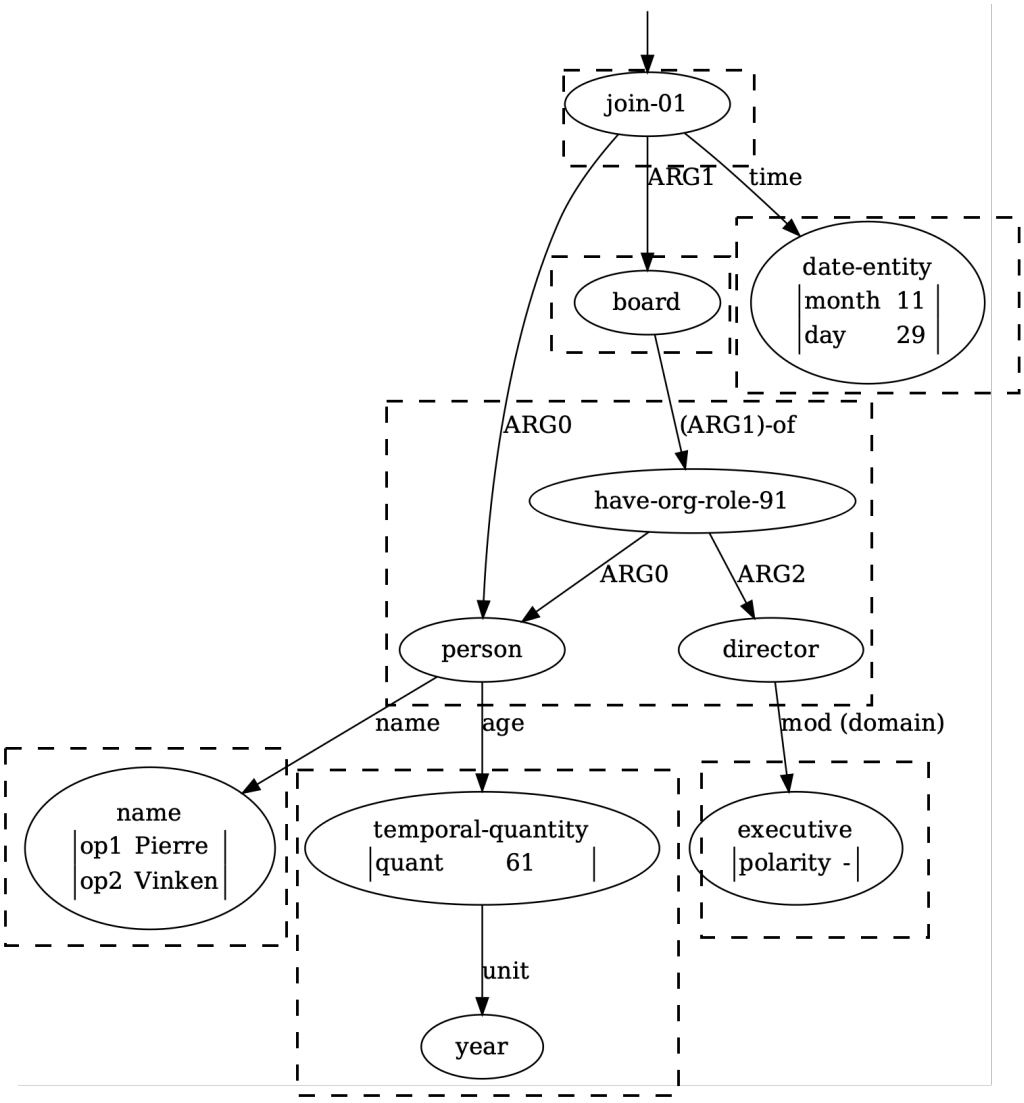

Figure 3.1: AMR output decomposition for the sentence #20001001

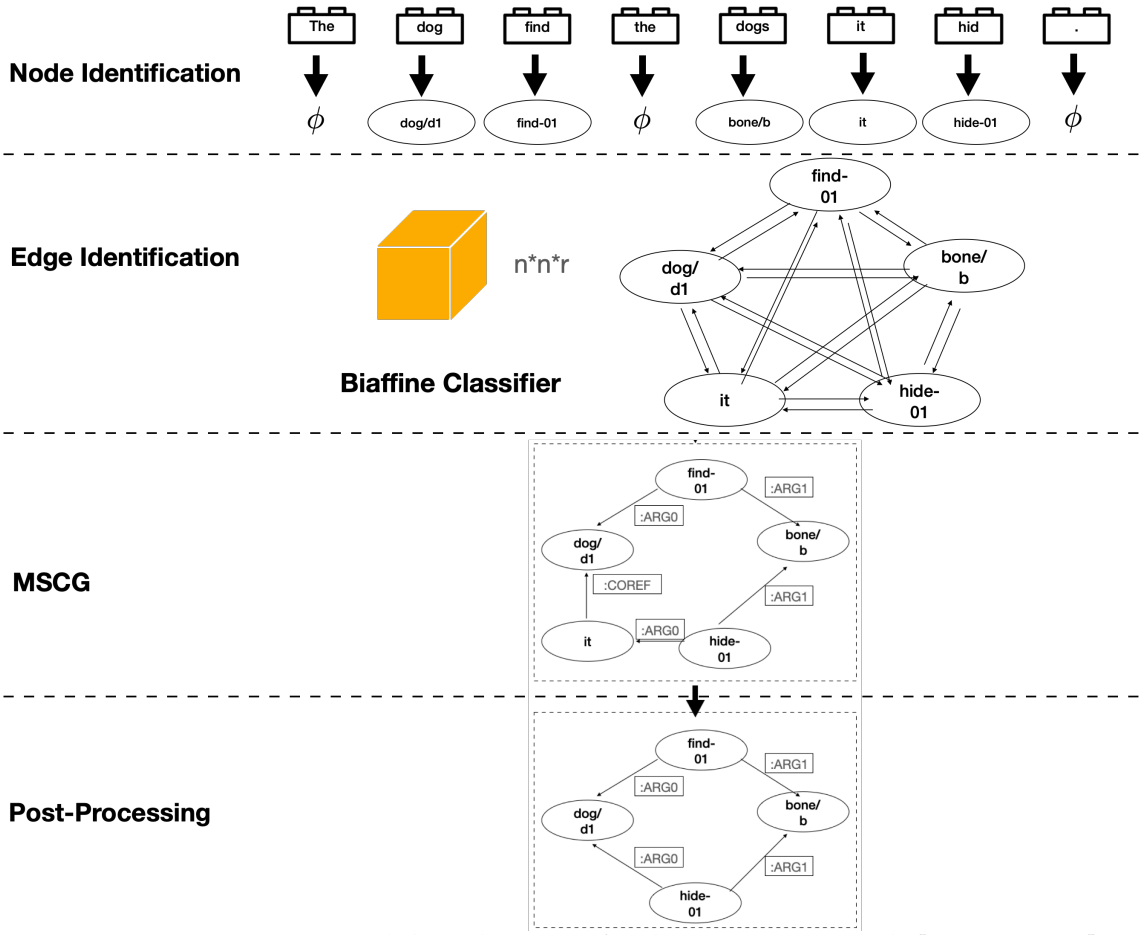

Figure 3.2: Two-stage graph-based parsing for the running exmaple [wsj#0209013]

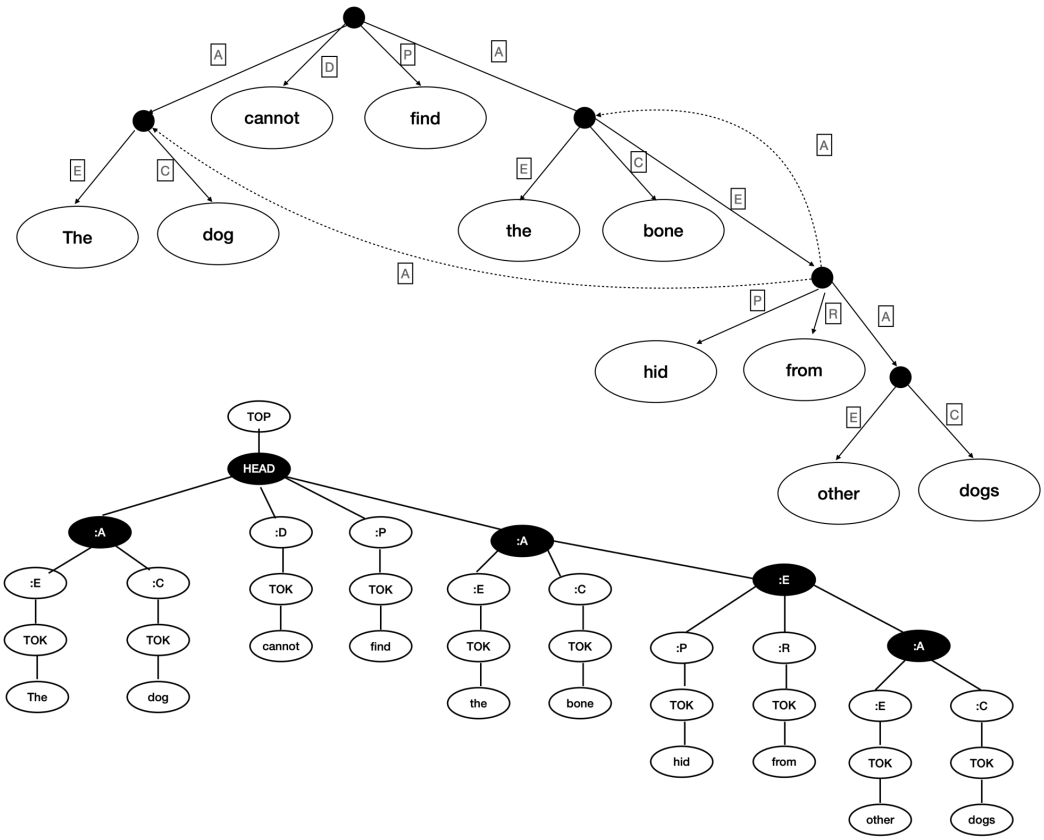

Figure 3.3: UCCA to Constituent Tree Transformation for [wsj#0209013]

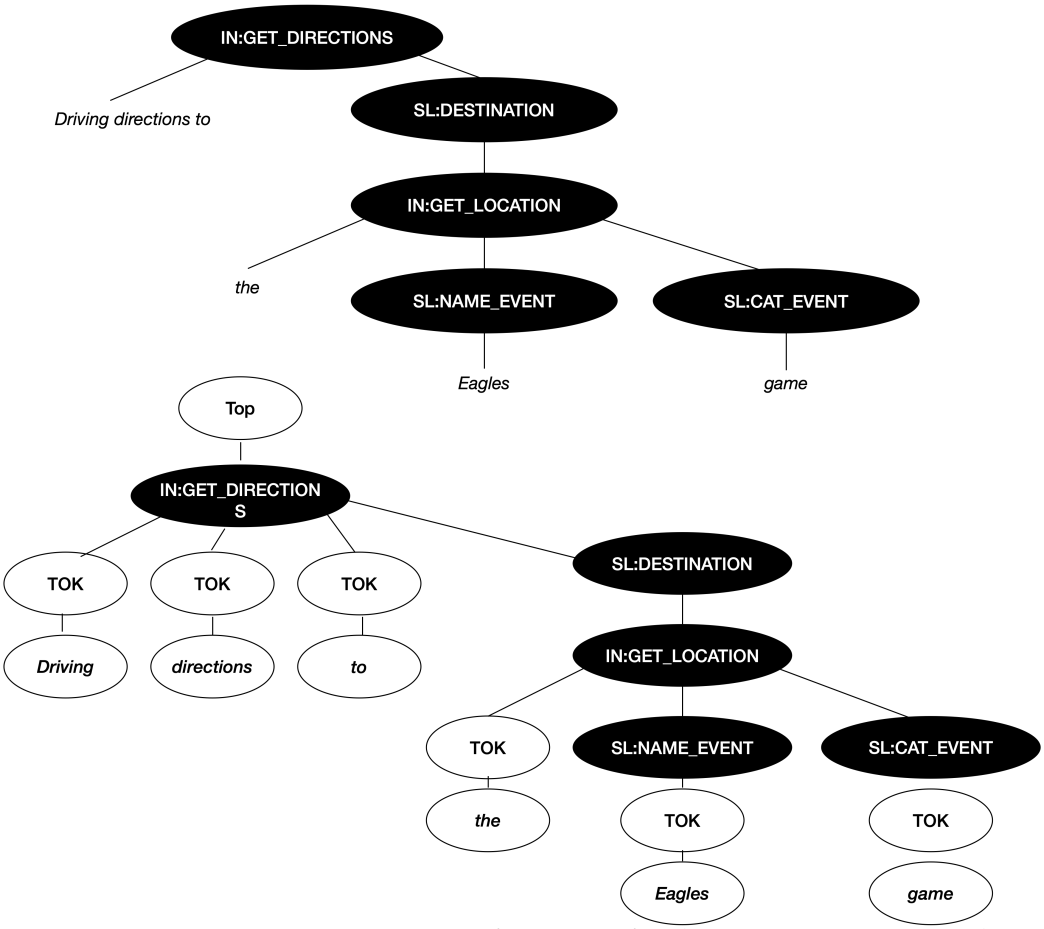

Figure 3.4: TOP to Constituent Tree Transformation for the utterance “Driving directions to the Eagles game"

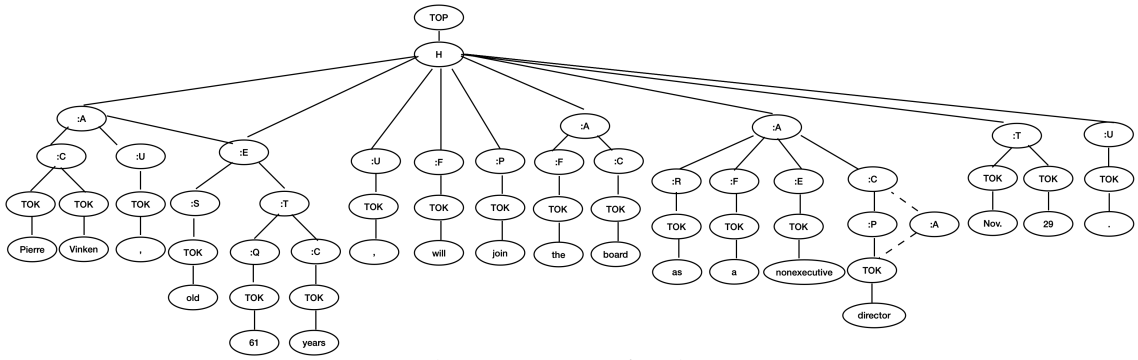

Figure 3.5: UCCA decomposition for the sentence #20001001

Table 3.1: The detailed lex-anchoring classifiers in our model. The round bracket means the number of ouput classes of our classifiers, * means copy mechanism is used in our classifier.

	Lexicon Anchoring		
	DM	PSD	AMR
Root	1	$\geq$ 1 (11.56%)	1
Node Label	Lemma	Lemma(*)	Lemma(*) + NeType(143+)
Node Properties	POS SEM-I(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 3.2: The detailed lex-anchoring classifiers in our model. The round bracket means the number of ouput classes of our classifiers, * means copy mechanism is used in our classifier.

	Phrase Anchoring	
	UCCA	TOP
Root	1	1
Node Label	N/A	Intent(25), Slot(36)
Node Properties	N/A N/A	N/A N/A
Edge Label	(15)	N/A
Edge Properties	“remote"	N/A
Connectivity	True	True
Training Data	6485	35741
Test Data	1131	9042

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

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 3.4: 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.

MR	Ours (P/R/F1)	Top 1/3/5 (F1)
UCCA(5)	80.83/73.42/**76.94**	81.67/77.80/73.22

Table 3.5: Results on TOP Parsing

Model	Exact Match	P	R	F1
Ours	80.85	92.77	91.05	91.9
Ours + ELMo	83.04	93.57	92.37	92.97
Ours + Roberta	85.89	96.07	93.08	94.92
Ours + SpanBERT	85.82	95.55	94.44	94.99
RNNG	78.51	90.62	89.84	90.23
Seq2Seq-CNN	75.87	89.25	87.88	88.56
Seq2Seq-LSTM	75.31	88.35	87.03	87.69
Seq2Seq-Transformer	72.2	87.09	86.11	86.6

Table 3.6: Our parser on AMR ranked 1st. This table shows the error breakdown when comparing to the baseline TUPA model and top 2 [163] in official results

	data	tops	labels	prop	edges	all
TUPA single	all	63.95	57.20	22.31	36.41	44.73
	lpps	71.96	55.52	26.42	36.38	47.04
TUPA multi	all	61.30	39.80	27.70	27.35	33.75
	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 3.7: Our parser on DM ranked 7th. This table shows the error breakdown when comparing to the model ranked Top 1 [166] in official results

	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 3.8: Our parser on PSD ranked 6th. This table shows the error breakdown when comparing to the model ranked top 1 [167] 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 3.9: The error breakdown of our UCCA parser. * denotes the ranking of post-evaluation results

	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. [163]	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

CHAPTER 4

STRUCTURAL INDUCTIVE BIASES FOR OBSERVING DIALOGUE IN THERAPY

Conversational agents have long been studied in the context of psychotherapy, going back to chatbots such as ELIZA [100] and PARRY [168]. 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[1]. 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, e.g., Crisis Text Line (`https://www.crisistextline.org`), 7 Cups (`https://www.7cups.com`), etc.

We ground our study in a style of therapy called Motivational Interviewing [MI, 56, 57], 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, 140]. Table 2.2 shows standard therapist and patient (*i.e.*, client) codes with examples. Recent NLP work [18, 141, 169, 170, *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 complementary dialogue observers:

1. **Categorization**: Monitoring an ongoing session by predicting MISC labels for thera-

[1]The main content of this chapter is published in Cao et al. [55]

pist 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 2.2). 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 provides 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 2.2 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.*, 18, 141, 169].

---

[2]The code is available online at `https://github.com/utahnlp/therapist-observer`. The majority content of this chapter is published in our paper [55]

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 Independent Factorization for Motivational Interviewing

In this section, we will formally define the two NLP tasks corresponding to the vision in Section 4.1 using the conversation in Table 4.1 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). We will define two classification tasks over a fixed dialogue history with n elements — *categorization* Section 4.2.1 and *forecasting* Section 4.2.2. Then we make a comparative analysis on the independent factorization for the two tasks Section 4.2.3.

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 previous post-hoc dialogue analysis setting, we have the following definition for the categorization task: *Given the dialogue history $H_n = \{u_1, u_2, ..., u_n\}$ which includes the speaker information, output a sequence of MISC label $L_n = \{l_1, l_2, ..., l_n\}$ for each utterance u_i.*

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 MISC forecasting task is a previously unstudied problem in the posthoc dialogue analysis.

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}.*

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.*, 171, 172, 173], 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 [174]. The forecasting task is also related to recent work on detecting antisocial comments in online conversations [175] whose goal is to provide an early warning for such events.

4.2.3 Independent Factorization for Categorization and Forcasting Task

Take the dialogue segment in Table 4.1 as a running example, we compare the categorization and forecasting task for each turn, with respect to the independent factorization for the dialogue history and predicting target. Both categorization and forecasting tasks are taking a sequence of dialogue utterances as input, and then predict a sequence of MISC labels as output. They share similar sequence labeling structures and seem to share the same independent factorization. However, they have different predicting goals when considering

each dialogue turn. In the following, we first introduce how we decompose the sequence of output MISC labels according to the speaker, and then we present the sliding window to decompose the input dialogue history. Table 4.2 shows the clear differences when we choose dialogue history window size as 3.

4.2.3.1 Output Decomposition

The outputs of both tasks are a sequence of MISC labels. Because each MISC label is naturally associated with a corresponding utterance, representing different functions as shown in Table 2.2. Hence for independent factorization, it is natural to decompose the whole sequence prediction into a set of continuous single MISC label prediction tasks as defined previously. In such a way, each MISC prediction in the categorization or forecasting task is independent of the MISC prediction in the previous dialogue turn. However, they have different goals for each dialogue turn. When the dialogue goes to turn 3, both the categorization and forecasting task will 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 forecasting task is to guess the future MISC code l_{n+1} for the unseen utterance u_{n+1}. To model this difference on output decomposition, we introduce four components for dialogue and sentence representation learning and search for the best model for each task Section 4.3. Furthermore, the MISC codes for the clients and therapist represent different meanings and functions for motivational skills. For example, the client MISC codes (ST, CT, and FN) discuss the client's intention to change or sustain problematic behavior or be neutral. While the therapist codes represent the actions the therapist takes for a motivational interview (e.g., giving information GI, simple reflection RES). To simplify the modeling, we also decompose the output according to the speakers. For example, for the categorization task, we build two models C_c and C_t for client and therapist codes separately. Similar for the forecasting task, we study the F_c and F_t for client and therapist. After such output decomposition, every model only needs to classify on a unified set of MISC labels, representing coherent goals for the clients or the therapist.

4.2.3.2 Input Decomposition and Alignments Discovery

For the input decomposition in previous lexical-anchoring and phrasal-anchoring, the corresponding output label can be directly derived from each input segment. For example, in lexical anchoring, each concept or subgraph in an AMR graph is aligned to each token or special entity. In phrasal anchoring, a non-terminal intent and slot label in TOP is aligned to a phrase in the input sentence. In MISC code prediction, the MISC code is assigned to a utterance. However, the MISC code can not be derived by only a single token , phrase or the aligned utterance. According to the MISC annotation guidedline, the identification of relevant words or utterances may depend on the whole dialogue history. For example, reflection related MISC (RES, REC) may need to discover which utterance in the dialogue history the therapist are reflecting to. Hence, more than the previous hard-aligned lexical or phrasal-anchoring, it requires to discovering the relevant details in the meaning of the sentence **within the context of dialogue history**. In Section 4.3, we proposed a hierarchical dialogue encoder to model the nest structures of dialogue, and we also offered word-level and utterance-level attention to help discover the relevant parts in it.

Due to the realtime setting, the input dialogue is naturally decomposed by time steps, and forms a sequence of incremental dialogue history $\{u_1\}$, $\{u_1, u_2\}$, and $\{u_1, u_2, u_3\}$. 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 dissertation study Section 4.5.2, we compare the window size as 8 and 16 for our models. We leave the extremely long dialogue context encoding as the future work.

4.3 Models for MISC Prediction

Modeling the categorization and forcasting tasks defined in Section 4.2 requires addressing four questions:

- How do we encode a dialogue and its utterances?
- Can we discover discriminative words in each utterance?
- Can we discover which of the previous utterances are relevant?

- 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 [176, 177, 178, 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 $\boldsymbol{v}_i$. Also, we will use the hidden states of each word as inputs to the attention components in Section 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 2.2 illustrates this. Word attention offers a natural mechanism for discovering such patterns.

---

[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.

We can unify a broad collection of attention mechanisms in NLP under a single high level architecture [179]. 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:

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

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 [180] and gated matchLSTM [181]. For simplicity, we replace the sequence encoder in the latter with a BiGRU and refer to it as GMGRU. Table 4.3 shows the corresponding definitions of f_c and f_m. 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.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 4.1, 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 [44]. 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:

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

$$\text{head}_i = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{W}_i^Q \left(\boldsymbol{K}\boldsymbol{W}_i^K\right)^T}{\sqrt{d_k}}\right)\boldsymbol{V}\boldsymbol{W}_i^V \tag{4.5}$$

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 $Q = [v_n]$ and $K = V = [v_1 \cdots v_n]$. Self-attention is defined by setting all three matrices to $[v_1 \cdots v_n]$. 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

From top 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 $\boldsymbol{v}_n^{selfatt}$.

We summarize the option input encodings for inference in Table 4.4. 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 2.2, 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 182] to the multiclass case. For a label l with probability produced by a model p_t, the loss is defined as

$$\text{FL}(p_t) = -\alpha_t(1 - p_t)^{\gamma} \log(p_t) \tag{4.6}$$

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 [183], outpatient clinics [184], college alcohol interventions [185, 186, 187, 188]. All sessions were annotated with the Motivational Interviewing Skills Codes (MISC) [189]. We use the train/test split of Tanana et al. [18], Can et al. [190] to give 243 training MI sessions and 110 testing sessions. We used 24 training sessions for development. As mentioned in Section 4.1, all our experiments are based on the MISC codes grouped by Xiao et al. [141].

4.4.1 Preprocessing and Model Setup

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 [191]. To embed words, we concatenated 300-dimensional Glove embeddings [72] with ELMo vectors [192].

We use 300-dimensional Glove embeddings pre-trained on 840B tokens from Common Crawl [72]. 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 [161] 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 4.5 with precision, recall and F_1 for each codes. Then, we also show the performance of these models against various baselines.

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.

4.4.2.1 Results on Categorization

Tables 4.6 and 4.7 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. [141], BiGRU$_{\text{generic}}$, Can et al. [190] and Tanana et al. [18] 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. BiGRU$_{\text{ELMo}}$, CONCAT$^{\text{C}}$, GMGRU^H and BiDAF^H are new baselines we define below.

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. [141] 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. [190] and Tanana et al. [18] use well-studied linguistic features and then tagging the current utterance with both past and future utterance with CRF and MEMM, respectively.

[4]Other related work in no context exists [e.g., 169, 193], but they either do not outperform [141] or use different data.

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. [170], but was reimplemented to take advantage of ELMo.

For categorizing client codes, 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 MLP(H_n) + MLP(v_n) performs better than just MLP(H_n). As shown in Table 4.6, 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 4.7 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. The set of baseline models are the same as in Table 4.6, but tuned for therapist codes. For the two grouped MISC set MIA and MIN, the results for Can et al. [190] and Tanana et al. [18] are not reported in their original work due to different setting. 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$_{\text{ELMo}}$.

4.4.2.2 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 the GMGRU^{H} model

that uses a gated matchGRU for word attention.[5]

Table 4.8 and Table 4.9 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 both the Table 4.8 and Table 4.9 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.

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 4.10.

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

[5]The forecasting task bears similarity to the next utterance selection task in dialogue state tracking work [173]. In preliminary experiments, we found that the Dual-Encoder approach used for that task consistently underperformed the other baselines described here.

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

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. [141], 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 4.12 and 4.13 summarize our results.

4.5.2.1 History Size

Increasing the history window size generally helps. The biggest improvements are for categorizing therapist codes (Table 4.13), especially for the RES and REC. However, increasing the window size beyond 8 does not help to categorize client codes (Table 4.12) or forecasting (Table 4.11).

4.5.2.2 Word-Level Attention

Only the model $\mathcal{C}_T$ uses word-level attention. As shown in Table 4.13, 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).

4.5.2.3 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 4.12 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 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 4.14, 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.

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 4.15

4.5.5 Can We Suggest Empathetic Responses?

Our forecasting models are trained on regular MI sessions, according to the label distribution on Table 2.2, 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 Related Work

In this section, we review the related work on MISC code prediction, dialogue representation and attention mechanism.

4.6.1 MISC Code Prediction

MEMM and CRF with handcrafted features are firstly proposed by Can et al. [190, 194] for the MISC code prediction task . Then Tanana et al. [195] improved the model by incorporating richer dependency relation features, which even outperformed a proposed recursive neural network baseline model. Recently, [141] proposed to use GRU with domain-specific word embedding and weighted cross-entropy loss to resolve the label imbalance problem. This paper studies the solutions drawn from recent work in dialogue representation, memory attention, and imbalanced classification. Besides that, other improvements exist, such as topic models based domain adaptio [170, 189], and prosodic features [196] have been proposed to improve the prediction tasks.

4.6.2 Dialogue Representation and Hierarchical Encoder

End-to-End neural networks and attention mechanisms have been widely used in many natural language tasks, such as text classification, question answering, dialogue

systems, etc. RNN, CNN, and Transformer has been quite effective for modeling the sequence of words [197, 198]. Combinations of them have also been used to model the hierarchical structure of document and dialogue context. They first use CNN or LSTM to get a sentence vector, and then a BiGRU to compose a sentence vector to get a document vector [176, 177, 178, 199, 200, 201]. In our paper, we use hierarchical GRU as skeletons. Other hierarchical combinations may also help, we leave it for future studies.

4.6.3 Attention Mechanism

Attention mechanism was first proposed by Bahdanau et al. [202] in machine translation, then various of extensions has been invented and widely used in other tasks, especially for question answering and dialogue system[180, 203, 204, 205, 206]. Attention on sentence-level representation also helps in many recent works such as abstractive summarization [207], dialogue state tracking [208, 209], document classification [200].

Previous work on hierarchical attention[200] tries always to use both word-level and sentence-level attention in a model, In this chapter, we argue that, for different tasks, we need different levels attentions. Always adding two-level attention may be not necessary. Recently, multi-heads multi-hop attention used in Transformer [44] became a more attractive attention mechanism. All the above advances in NLP inspired us to systematically analyze whether or how they impact our tasks.

4.7 Chapter Summary

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. Ablation studies on history size, word-level and sentence-level attention, focal loss and generic or domain-specific embeddings, shows how each of them helps in our tasks. Extensive analysis shows that our model can decrease the label confusion compared to previous work, especially for reflections and rare labels. The experimental results show that word-level and sentence-level attention mechanism can help our independent factorization to discover the relevant parts for our local factor modelling without the hard alignment.

Furthermore, our model trained on regular MI sessions with both MI adherent and non-adherent utterances can still achieve a recall@3 of 76.9 on selected good MI sessions. Hence our model potentially helps for therapist training on psychotherapy domain. Beyond the strategy of mimic good therapy dialogue session, in the future, we also can extend the study to other strategies. For example, one possibility is to use some sort of an external idea of checklist on how the dialogue should go. In this way, the checklist can supervise the dialogue flow on the going. Another possiblity is that we can use reinforcement learning to model the reward and policy towards the final goal of motivational interview: facilitating behaviour change.

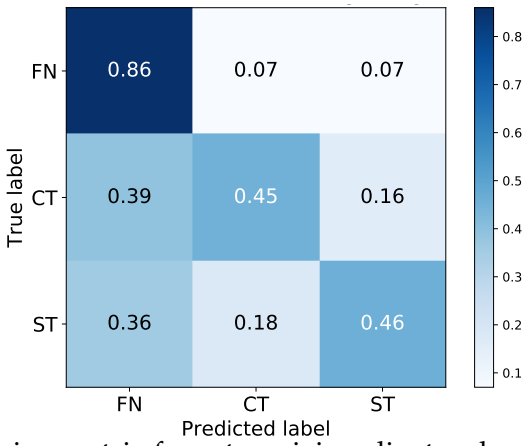

Figure 4.1: Confusion matrix for categorizing client codes, normalized by row.

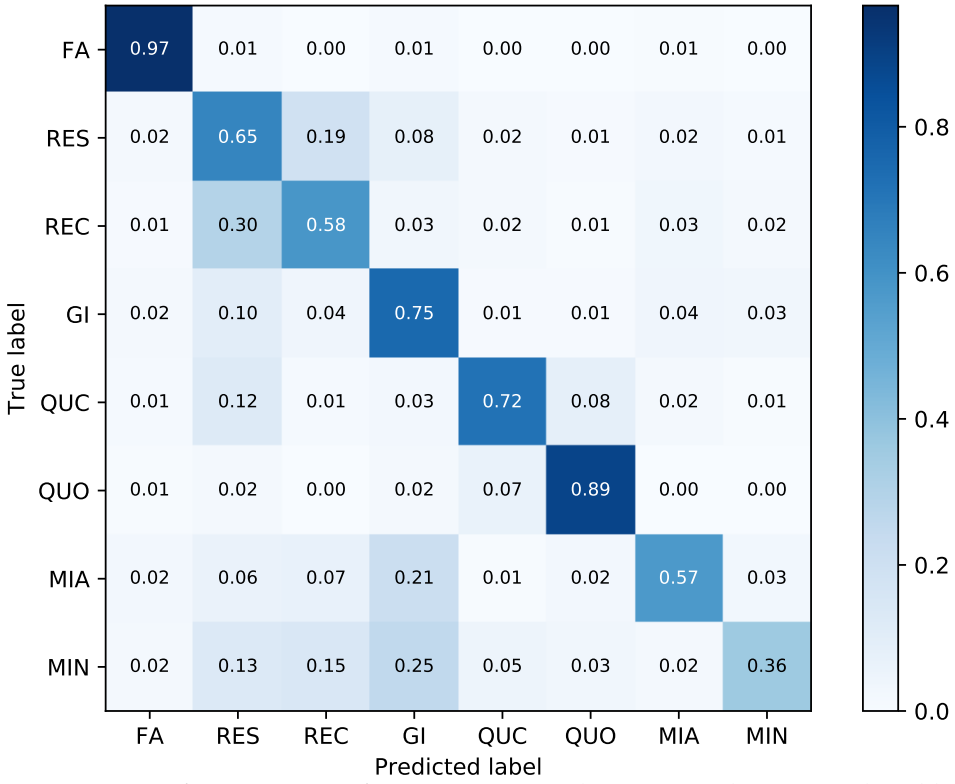

Figure 4.2: Confusion matrix for categorizing therapist codes, normalized by row.

Table 4.1: An example of ongoing therapy session

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

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: Summary of word attention mechanisms.

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

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

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: Main results on categorizing client codes, in terms of macro F_1, and F_1 for each client code.

Method	macro	FN	CT	ST
Majority	30.6	**91.7**	0.0	0.0
Xiao et al. [141]	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. [190]	44.0	91.0	20.0	21.0
Tanana et al. [18]	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 therapist codes, in terms of macro F_1, and F_1 for each therapist code.

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. [141]	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. [190]	-	94.0	49.0	45.0	74.0	72.0	81.0	-	-
Tanana et al. [18]	-	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 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	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 therapist codes, in terms of Recall@3, macro F_1, and F_1 for each label on 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: 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.

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: Ablation study on forecasting task on both client and therapist code. $*$ row are results of our best forecasting model $\mathcal{F}_C$, and $\mathcal{F}_T$. $\setminus$ 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	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	$-\ \text{SELF}_{42}$	**24.9**	22.5	76.0	71.4	12.7	24.9	58.3	28.8	5.9	**17.4**	9.7
	$\setminus\ \text{ANCHOR}_{42}$	22.9	22.9	76.2	72.2	15.5	**24.6**	59.5	27.1	7.7	16.3	8.3
	$+\ \text{GMGRU} \setminus \text{ANCHOR}_{42}$	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 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.

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 therapist codes, $*$ is our proposed model $\mathcal{C}_T$. $\setminus$ 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.

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
	$\setminus$ BiDAF	63.5	54.2	51.3	22.6
sentence attention	- ANCHOR_{42}	64.9	56.0	54.4	21.8
	$\setminus$ SELF_{42}	63.4	55.5	48.2	21.1

Table 4.14: 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.

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

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**
	$ELMo_{psyc}$	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
	$Glove^{pysc}$	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
	$ELMo_{psyc}$	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
	$Glove^{pysc}$	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**

CHAPTER 5

NATURAL LANGUAGE AS INDUCTIVE BIASES FOR TRACKING DIALOGUE STATE

From early frame-driven dialog system GUS [17] 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 [19, 210, 211] and transferable dialog state tracking to unseen intent/slots [212, 213, 214].

As shown in Figure 5.1, the intent classification task is to understand what the user is 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 5.1, bolded text is the evidence for different slot values. For example, “economic" implies the value of the slot seat class, and “June 10" indicates the departure time. In Figure 5.1, two flight services present two different ontologies 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 states for each user dialogue turn and request different commands for downsteam information retrieval components. When there is a new flight booking service, then there will be new ontology for the new domain. Hence, even if it shares a lot of overlapping functionalities with the previous two flight services, we still need to annotate new data for the new service and retraining the new model on the newly annotated data. Such re-annotating and retraining on new services are costly.

Recently, Rastogi et al. [215] 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 5.2, 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) [216]. 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 chapter[1], we focus our investigation on the study of three aspects in schema-guided dialog state tracking: (1) schema encoding model architectures (2) supplementary 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, besides the independent factorization and the attention mechansiam in previous chapters, we show that natural language description can further offer discriminative features to our factor modelling. Esepcially, these descriptions can connect semantically similar concepts across heterogeneous APIs, thus allowing a unified model to handle unseen services and APIs in data-poor cases. 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 bench-

[1]The main content of this chapter is published in our paper [217]

marking datasets with variations in schema description for both SG-DST and MULTIWOZ 2.2.

5.1 Independent Factorization for Dialogue 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 5.2, the key difference between schema-guided dialog state tracking and the classic paradigm is the newly added natural language descriptions. Beyond the classic independent factorization, we also add a natural language description to describe each output labels. Hence, the independent factorization for schema-guided dialog state tracking can be represented as Equation 5.1. Instead of using the label of decomposed part y_c only, we also use the natural language description of y_c.

$$E(x, y) = \sum_{c \in C} E(x, y_c) = \sum_{c \in C} E(x, a(y_c), desc(y_c)) \tag{5.1}$$

In this section, we will analyze the task structure of dialogue state tracking and the design for independent factorization. First, we will examine the decomposition of dialogue state, and split the dialogue state tracking into four independent subtasks. In addition to the output decomposition, we propose to add a natural language description for each part of the output label to support unseen new services. We introduce the schema components in schema-guided dialog state tracking. Then, we show how each output part can be derived from the input decomposition. Finally, we outline the research questions in our paper and address each of them in the rest sections of this chapter.

5.1.1 Output Decomposition: Four Subtasks

As shown in Figure 5.2, 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 [218]. 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).

Beyond the independent factorization of 4 subtasks, we also consider describing each decomposed intent/slot label with natural language. By using natural language description to define a dynamic set of service schema. As shown in Figure 5.2, we hope 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. Hence, in this independent factorization, we require matching the current dialog history with candidate schema descriptions for each subtask and multiple times. Such matches are computation-heavy, which raises new challenges to our modeling.

Figure 5.2 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 a set of predefined candidate values (Boolean, numeric or text). For instance, *has_live_music* in Figure 5.2 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.

5.1.2 Input Decomposition and Alignments Discovery: Attention Mechanism

In lexical-anchoring, the input sentence is decomposed exclustively into tokens and special entities. In phrasal-anchoring, the input sentence is decomposed into nested phrases which forms a tree structure. In sentential-anchoring, we show that the input dialogue are naturally decomposed into a sliding dialogue history window. The input decomposition for dialogue state tracking is the same as sentential-anchoring MISC prediction tasks. Because for both MISC prediction and dialogue state tracking tasks, we cannot simply find relevant parts to predict the intent and slots, which requires jointly considering the current dialogue utterance and the previous dialogue history. Hence, due to the success of attention mechanisms in the MISC prediction, we also use the attention mechanism to discover the more detailed relevant parts for our dialogue state tracking. We ground our study on the pre-trained transformer model: BERT [70]. BERT uses the self-attention mechanism to dynamically learn the relevant parts from a large amount of text, which has been shown great success for many NLP tasks, including dialogue state tracking [219, 220]

5.1.3 New Questions

According the output decomposition with four independent subtasks, and the alignments analysis, we summerize the design for independent factorization in Table xx.

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?(Section 5.3)
Q2. How do different supplementary trainings impact each subtask? (Section 5.4)
Q3. How do different description styles impact the state tracking performance? (Section 5.5)

5.2 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.

5.2.1 Schema-Guided Dialog Dataset

SG-DST dataset [2] is especially designed as a test-bed for schema-guided dialog, which contains well-designed heterogeneous APIs with overlapping functionalities between services [215]. In DSTC8 [216], SG-DST was introduced as the standard benchmark dataset for schema-guided dialog research. SG-DST covers 20 domains, 88 intents, 365 slots.[3] 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 Section 5.5, and then we perform both homogenous and hetergenous evalution on it.

5.2.2 Remixed MultiWOZ 2.2 Dataset

To eliminate potential bias from the above single SG-DST dataset, we further add MULTIWOZ 2.2 [221] to our study.

---

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

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

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

Among various extended versions for MultiWOZ dataset [2.0-2.3, 19, 221, 222, 223] , 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 Table 5.2. 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. [213]. 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.

5.2.3 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, 224] with dialog flows as seeds, then paraphrased by crowd-workers. While MULTIWOZ 2.2 are human-to-human dialogs [H2H, 225], 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 dialogue.

5.2.4 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.

5.3 Dialogue and Schema Modeling

In this section, we focus on the model architecture for matching dialog history with schema descriptions using pretrained BERT [39] [4]. We first compare different encoder architectures in Section 5.3.1. Then for each of four subtasks, we show the classification head (Section 5.3.2) and the results (Section 5.3.3)

5.3.1 Encoder Architectures

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 5.3.

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

5.3.1.1 Dual-Encoder

It consists of two separate BERTs to encode dialog history and schema description respectively, as Figure 5.3 (a). We follow the setting in the official baseline provided by DSTC8 Track4 [216]. 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 $\mathbf{CLS}_S$ with each token embedding in the dialog history, denoted as $\mathbf{TOK}^{DE}$.[5] Because the candidate schema embeddings are encoded independently from the dialog context, they can be pre-computed once and cached for fast inference.

5.3.1.2 Cross-Encoder

Another popular architecture as Figure 5.3 (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 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.

5.3.1.3 Fusion-Encoder

In Figure 5.3 (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

[5] A schema-aware dialog token embedding can also be computed by attention or other method for span-based detection tasks [220, 226]

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.

5.3.2 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.

5.3.2.1 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.

5.3.2.2 Requested Slot

As in Figure 5.2, 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 $P_{req}^{active} > 0.5$.

5.3.2.3 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.

5.3.2.4 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^i_{start} and f^i_{end} 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.

5.3.3 Experiments on Encoder Comparison

To fairly compare all three models, we follow the same settings of schema input as in Table 5.3. We trained separate models for SG-DST and the remixed MultiWOZ datasets for all the experiments according to the detailed setup in Section 5.2.4 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 this dissertation.

5.3.3.1 Comparison on Accuracy

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

5.3.3.2 Comparison on 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 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.

5.4 Supplementary Training

Besides the pretrain-fintune framework used in Section 5.3, Phang et al. [227] 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 different intermediate tasks for supplementary training. However, how to choose those intermediate tasks is a challenge problem. Our inductive biases on choosing intermediate tasks are: *Intermediate tasks that sharing similar structures with downsteam tasks may help the supplementary training.* Hence, for natural language description modelling, we will first show the similarity between our proposed **Natural Language inference** and **Question Ansering** with our four subtasks, then analyze the results on how they performed.

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

5.4.1 Intermediate Tasks

As described in § 5.3.2, all our 4 subtasks take a pair of dialog and schema description as input, and predict with the summerized sentence-pair **CLS** representation. While NONCAT 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 hypodissertation/premise sentence pair, natural language inference is a task to determine whether a hypodissertation 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 Section 5.3.2, 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.

5.4.2 Results on Supplementary Training

Table 5.5 and Table 5.6 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 NONCAT task, while neither of them helps much on CAT task. Recently, Namazifar et al. [228] 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.

5.5 Impact of Description Styles

Previous work on schema-guided dialog [216] are only based on the provided descriptions in SG-DST dataset. Recent work on modeling dialog state tracking as reading comprehension [229] 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.

5.5.1 Benchmarking Styles and Experiment Setup

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. [229]. 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 [230] 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. Table 5.7 shows examples for different styles of schema descriptions.

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 in both homogeneous (Section 5.5.2) and heterogeneous settings (Section 5.5.3)

5.5.2 Homogeneous Evaluation

In this section, Table 5.8 summarizes the performance for homogeneous evaluation, while Table 5.9 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 5.10.

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

5.5.2.1 Is name-based description enough?

As shown in Table 5.8, 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 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".

5.5.2.2 Does question format help?

As shown in Table 5.8, 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 Q-NAME and NAMEONLY, there is no consistent improvement on both of the two datasets. Further more,

---

[8] Only exception happens in CAT on MULTIWOZ 2.2. When creating MULTIWOZ 2.2 [221], the slots with less than 50 different slot values are classified as categorical slots, which leads to inconsistencies. 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., 231, OPINE].

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 5.9 shows that, after applying the supplementary training on SQuAD2 (Section 5.4), 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.

5.5.3 Heterogeneous Evaluation

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 5.10. 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 (Section 5.5.2), 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 5.10, 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.

5.6 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.

5.6.1 Multi-Sentence Encoder Strategies

Similar to the recent study on encoders for response selection and article search tasks Humeau et al. [226], we also conduct our comparative study on the two typical architectures *Cross-Encoder* [232, 233] and *Dual-Encoder* [234, 235]. 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 Section 5.3.1, which improves efficiency without sacrificing much accuracy.

5.6.2 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 [19], M2M [211], accompanied by studies on key issues such as in/cross-domain carry-over [236]. 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** [213]. Additionally, explicit cross-domain carryover [237] 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 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.

5.6.3 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 [238, 239], neural belief tracking [212], generative dialog state tracking [214, 240], modeling DST as a question answering task [218, 229, 241, 242]. 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 in Section 5.5.1.

5.7 Chapter Summary

In this chapter, beyond the independent factorization and the attention mechansiam in previous sentential anchroing modelling, we show that natural language description can further offer discriminative features to our factor modelling. Esepcially, 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 in data-poor cases.

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. Natural language description can also be used for other tasks where the natural language can be used to describe the overlapping functionalities and differences. However, how to efficient design the natural language description will be a challenge problem, recent ideas on prompt-tuning may potentially help in the future [243, 244]

Figure 5.1: Different Flight Service Ontology for Dialogue State Tracking

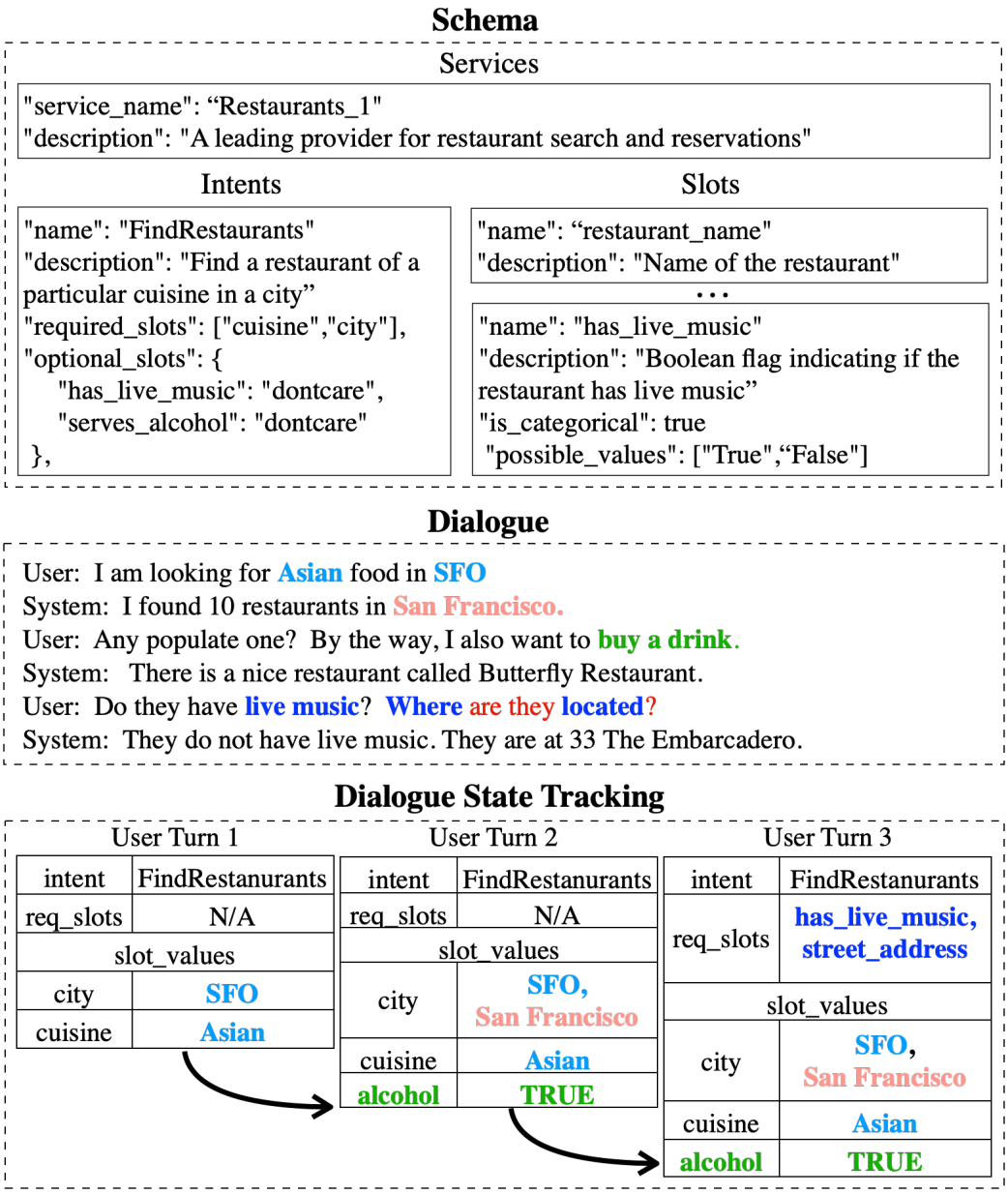

Figure 5.2: An example dialog from Restaurant_1 service, along with its service/intent/slot descriptions and dialog state representation.

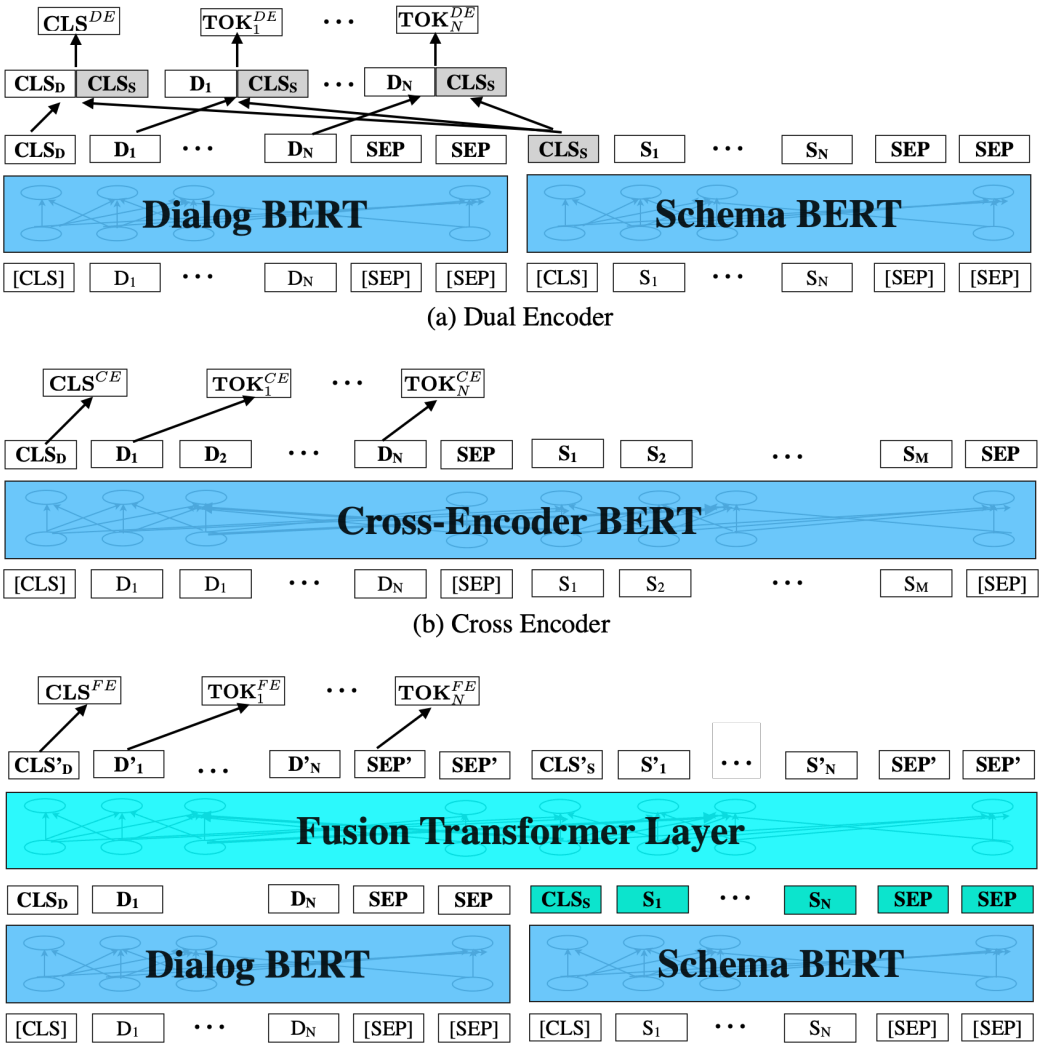

Figure 5.3: Three kinds encoders for schema encoding: Dual-Encoder, Cross-Encoder and Fusion Encoder. Shaded block will be cached during training.

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

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

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 5.3: 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.

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

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. [215]. Other models are trained with the architecture described in Section 5.3.2.

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

	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 5.6: Relative performance improvement of different supplementary training on MULTIWOZ 2.2 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 5.7: Different extensions of schema descriptions

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 account_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	The type of the user account

Table 5.8: 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.

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

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 5.10: Results on unseen service with heterogeneous description styles on SG-DST dataset. More results and qualitative analysis are in the Appendix C.2.

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**

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 Research Summary and Contributions

In this dissertation, we claim that by designing *Structural Inductive Bias* and *Natural Language as Inductive Biases*, models with naive independent factorization can achieve strong performance on predicting the natural language structures across multiple broad-coverage and application-specific representations.

The main contributions of this dissertation are as follows:

1. Based on the assumption of independent factorization and the structural inductive biases about different anchoring-types, we proposed a unified parsing framework to support both **explicit lexical-anchoring** (including DELPH-IN MRS Bi-lexical Dependencies [DM, 11] and Prague Semantic Dependencies [PSD, 12, 13]), and **implicit lexical anchoring** (AMR). For the **phrasal-anchoring** Universal Conceptual Cognitive Annotation [UCCA, 53] and Task-oriented Dialog Parsing [TOP, 54], 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 [50] *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 and modeling for two speaker seperately: (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 [55]. 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

In this dissertation, by utilizing structural inductive biases and natural language as inductive biases, we designed effient deep linguistic structured prediction methods with indepdent factorization. Though these methods achieved better performance on various broad-coverage and application-specific symbolic representations, there is still room for further improvment. In this section, we demonstrate the future directions including extending to other kinds of factorizations beyond independent factorization (Section 6.2.1), applying to other symbolic representations (Section 6.2.2), enhancing contextualized representation (Section 6.2.3), repressenting other kind of biases (Section 6.2.4), and more ways of learning and transferring the inductive biaes (Section 6.2.5)

6.2.1 Beyond Independent Factorization

This dissertation mainly focuses on independent factorization, which ignores the interdependence between output parts. Our experiments show that the contextualized representation can capture the interdependence within the input parts; thus, they can

offer discriminative features to predict each output part independently. However, we also can extend our work on other factorizations 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.

6.2.2 Apply to Other Symbolic Representations

As shown in Chapter 3, the above structural inductive bias in lexical, phrasal, and sentential anchoring can be easily extended to other linguistic structured prediction tasks, such as coreference resolution, semantic role labeling, where the main task structures has been studied in our broad-coverage meaning representation parsing. Taking the *coreference resolution* tasks as an example, we show how to apply the independent factorization on a new task in Section 6.2.2.1. Then we show the potential future application on other symbolic representations in Section 6.2.2.2.

6.2.2.1 Coreference Resolution

Coreference resolution is the task of clustering mentions in text that refer to the same underlying real-world entities or events. As shown in the left bottom of Figure 1.4, “dog" and “it" are pointed to the same dog.

First, we consider the **output decomposition** for coreference resolution task. A classic problem formula for coreference resolution task is to define a set of antecedent assignments y_i for each of span x_i in the given document. The set of possible assignments for each $y_i \in \Phi, 1, ...i-1$. Φ means dummy antecent or the span i is not a mention, and every

y_i can be assigned with the preceding spans. Hence, then there are three factors for the pair-wise coreference core. (1) predict $I_m(i)$, whether span i is a mention. (2) predict $I_m(j)$, whether span j is a mention. (3) search for y_i, based on the previous mention predictions $I_m(i)$ and $I_m(j)$. One problem with the above factorization is that the third factor y_i may have multiple values, which are antecedent for multiple preceding spans. Furthermore, it is not easy to do N ways classification directly, because for each y_i the output candidate labels are different, and the features only from span i may not enough to produce the y_i. Instead, a better independence factorization can model this into a two-stage pipeline model as shown in AMR parsing: Stage 1, predicting the possible mentions via local classifier for each local factor $I_m(i)$. Stage 2, predicting the edges labels between all pairs as local binary classification $I_a(i, j)$, which means whether span i and j are in the same cluster without caring about the preceding or not. In this output factorization, we can model two part of contextualized representation from span i and span j. Such a pair-wise factorization will offer more discriminative features and simplifer all the classification into binary classification. We still can use a bi-affine classifier to model the pair-wise binary classification. Furthermore, logic constraints can be added here to enforce the consistency between local decisions.

Then, we consider the **input decomposition and alignment discovery** for coreference resolution. According to the above analysis on output decomposition, we need to decompose the inputs into candidate spans. Assuming there are N words in the input document, then there will be $\frac{T(T+1)}{2}$ possible spans. However, we don't need to consider the spans cross the sentence boundaries, and we only mainly consider the pronouns, nouns , and other entity-related or event-related spans. Hence, such inductive biases about extracting potential spans will also simplify the input decomposition. At the same time, the pruning of the spans will reduce the computation for the second stage on binary edge label prediction.

Hence, our proposed methods for two-stage graph-based parsing can be used in coreference resolution with task-specific output decompositions and input decompositions. The contextualized representation for each span can also benefit from span representation we studied in phrasal-anchoring parsing Section 3.3.4

6.2.2.2 Independent Factorization for Other Tasks

For broad-coverage symbolic representations, our dissertation only covers the representation for a single sentence. While in the future, we also study on multi-sentence and document-level representations such as MS-AMR [245], Doc-AMR [246], discourse parsing [247], and soon on.

For application-specific symbolic representations, besides the single sentence representation in [TOP, 54], we also can extend our structured prediction models into session-based conversational representation such as session-based TOP [142], [TreeDST, 143], and [Dataflow, 144]. 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 Contextualized Representation

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

6.2.3.1 Extreme Long Context

First, we need to resolve the extrem long text encoding problem. Our current models of psychotherapy dialogue and schema-guided dialog only consider 8 to 16 utterances as the dialogue history window. However, we have more than 500 utterance in a single therapy session. Furthermore, a psychotherapy treatment may last for months and years which involves multiple dialogue sessions. The extended context problem also exists in other domains, such as scientific document analysis and threaded conversations in social media. The extreme long context modeling has already attracted a lot of attentions [248, 249]. However, how to joinly model the long text in interactive form (especially in multi-party dialogue) is still not well studied.

6.2.3.2 Contextualized Representation Beyond Text

In this dissertation, we mainly utilize the contextualized text representation to model the structured prediction. However, human acquisition of information and communication with the world does not occur based on pure language input. From a cognitive perspective, processing language in isolation without information in other modalities seems insufficient. Recently, multimodal contextualized representation has also been widely used in the natural language processing task. Beinborn et al. [250] shows that multimodal grouding of verbs play a crucial role for the compositional power of language. Jointly considering both visual and textual models has been widely used in many tasks, such as Multimodel Aspect-Based Sentimental Analysis (MABSA) [251], Named Entity Recognization [252] and so on. Some tasks consider both multimodal inputs and multimodal outputs, e.g., Question Answering [253]. More recently, vision-language pre-training further boosts the power of contextualized representation [254, 255]. Recent works on multimodality above raise challenges for future structured prediction tasks and their independent factorization.

6.2.4 Other Biases in Other Formalism

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

6.2.4.1 Declaritive Constraints and Other Latent Models

In this dissertation, we mainly consider the interdepenence with prior knowledges on how to decompose the input, output and their alignments. For example, in Section 3.2.3, we model the exclusive alignement by relaxing the discrete alignement variable with soft score matrix, and resolving the intractable marginal inference with Perturb-and-Max (Section 3.2.3.2) and Gumbel-Sinkhorn networks (Section 3.2.3.3).

In the future, we also can inject other structural interdependence/constraints with declarative tools, such as integer linear programming [60], probabilistic neural logic rules [61, 62, 63, 256]. However, many existing work on constrained learning is to design regularizer to optimize the training objective in the output space. In the future, modeling the constraints on the state space or action space will offer more direct supervision both the model structured learning and inference [257].

Furthermore, continuing the research line of modeling latent variables, we plan to study more ways to relax structural inductive biases as continuous and differentiable latent variables in the end2end deep learning [64, 65].

6.2.4.2 Other Biases: Casuality and 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 [25], which are potentially helpful for machine intelligence.

Beside the compositionality-related inductive biases, casuality is another important inductive bias for human intelligence, which may also help deep learning. Especially, current factorization and Markov random field-based formalism only can capture the undirectional correlation between different variables. In the future, we will extend the formalism to bayesian networks and intervention-based casual inference [258, 259].

Furthermore, we also plan to model the uncertainty and approximate biases for the learning and inference in deep learning.

First, approximate inference algorithms can be derived by approximating the underlying exact inference problem. For example, in this dissertation, the partition function in the posterior alignment model is intractable, to make it tractable and differentiable, we leverage the Peturb-and-Max (Section 3.2.3.2) and Gumble-Sinkhorn network (Section 3.2.3.3). The similar ideas on esitimating the marginal inference can also be used in other latent structured predictin models mentioned in Section 6.2.4.1.

Furthermore, to handle the uncertainty issues in practical machine learning, beyond the point estimation of learning a single set of best weights, bayesian deep learning seeks to equip deep learning with uncertainty estimation [260]. Finally, the error tolerance in machine learning also can be leveraged to design efficient machine learning methods, such as stochastic distributed optimization [261], hardware-aware efficient quantization [262], and so on.

6.2.5 Learning and Transfering the Inductive Biases

In this dissertation, we mainly design inductive biaes with prior knowledge about structured prediction tasks including the input decomposition, output decomposition and factor mdoeling. Besides designing the inductive bias by hand, we also can explore more on how to learn the inductive biases Section 6.2.5.1 and transfer to other new tasks Section 6.2.5.2.

6.2.5.1 Learning the Inductive Biases

Inspired by self-supervised pretraining in ELMo and BERT [39], our VLDB'2022 paper [263] 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* [264] 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. In the future, such self-supervied method can also be used for other data beyond natural language, such web page, programming language, documents, and so on.

6.2.5.2 Transferring Inductive Biases

Learning to learn is an essential inductive bias in human intelligence [29], 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. Our previous work on schema-guided dialogue state tracking [217] 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

[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.

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.

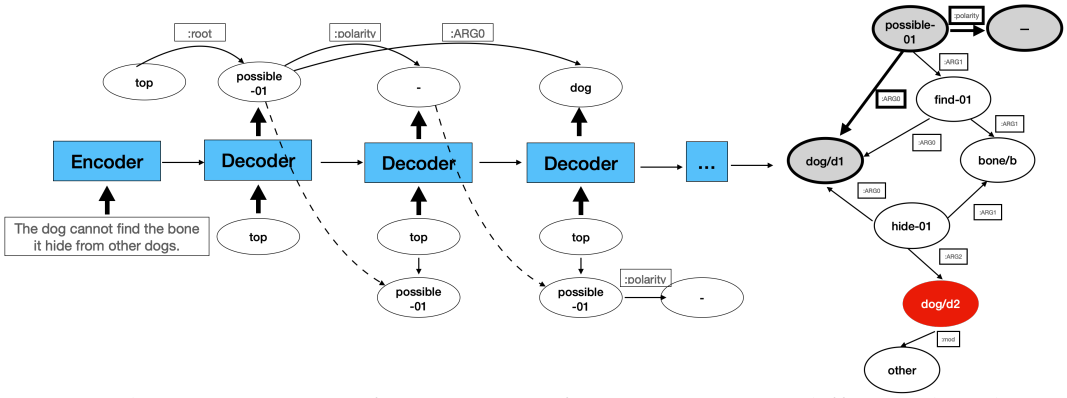

Figure 6.1: The autoregressive factorization of AMR Parsing in different decoding time step.

APPENDIX A

CLUSTERING STRATEGIES ON MISC CODES

Motivational Interview Skill Codes (MISC) [56, 57] are designed to represent the functions of each client and therapist utterance in the psychotherapy dialogue. The original MISC annotation guideline [140] included 28 labels (9 client codes, 19 therapist codes). Due to data scarcity and label confusion, various strategies are proposed to merge the labels into a coarser set. In this dissertation, we adopt the grouping proposed by Xiao et al. [141]. The detailed clustering has been introduced in Table 2.2 in Section 2.2.2.1. This appendix first introduces the existing clustering strategies in Section A.1, and then demonstrates the detailed label distribution and examples of the grouped MIA and MIN labels (Section A.2) in our strategy *MISC-11*.

A.1 Main Clustering Strategies

This section summarizes the existing clustering stategies of MISC codes as follows:

- *MISC-28:* The original MISC description of Miller et al. [140] included 28 labels (9 client codes, 19 therapist codes). However, among them, there are 13 therapist codes are too rare to learn an efficient model on predicting them.

- *MISC-8:* Hence, Can et al. [190] retain 6 original codes FA, GI, QUC, QUO, REC, RES, and merge the remaining 13 rare therapist codes into a single COU label. Furthermore, they merge all 9 client codes into a single CLI label. In total, they have 7 labels for therapists and 1 label for clients.

- *MISC-15:* Instead, Tanana et al. [18] merge only 8 of 13 rare labels in therapist codes into a customized label named OTHER, and then they cluster client codes into 3 labels according to the valence of changing, sustaining or being neutral on the addictive

behavior [189]. Hence, in total, they have 12 labels for therapists and 3 labels for clients.

- *MISC-11:* Then Xiao et al. [141] combine and improve above the two clustering strategies (MISC-8 and MISC-15) by splitting all the 13 rare labels according to whether the code represents MI-adherent(MIA) and MI-nonadherent (MIN). In total, they have 8 therapist codes and 3 client codes as shown in Table 2.2 (Section 2.2.2.1). We show more details about the original labels in the group MIA and MIN in Table A.1 (Section A.2).

A.2 Details of MISC-11

In *MISC-11*, 13 rare labels are grouped into two set: 8 labels representing MI-adherent(MIA) and 5 labels representing MI-nonadherent (MIN). Table A.1 shows in the distribution of the grouped MIA and MIN labels. Compared to the other labels in Table 2.2, the grouped MIA and MIN are still relative less frequent. Hence, we use the focal loss to resolve the label imbalance issue, and improve the overall performance. The examples in Table A.1 will help to understand the meanings of the rare labels. For more details about each of the MISC code, please refer to the latest MISC annotation guideline [265].

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

Code	Count	Description	Examples
MIA	3869	Group of 8 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) "It's 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 5 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)

APPENDIX B

COMBINATIONS OF SCHEMA DESCRIPTIONS

In schema-guided dialogue modeling, as shown in Figure 5.2 and Section 5.1.1, it shows three main schema components: service, intent, slot. Natural language descriptions are also added to the three components. This appendix includes the ablation study on the combinations of the three descriptions.

B.1 Combinatory Settings

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.

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.

B.2 Results on Description Combinations

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.

Table B.1: Models using different composition of schema, results on test set of SG-DST and our remixed MULTIWOZ 2.2. For zero-shot performance on MultiWOZ, we simply merge the zero-shot performance on each domain.

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

APPENDIX C

SUPPLEMENTARY TRAINING AND DESCRIPTION STYLES

This appendix shows more analysis and results on how to model the natural language description in schema-guided dialogue, including supplimentary training Section C.1, homogeneous and heterogeneous evaluation on different description styles Section C.2.

C.1 More Results of Supplementary Training

Table C.1 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.

To evaluate show supplementary training impact on different styles, we also show the detailed results when we apply supplementary training of SQuAD2 on the NONCAT tasks (Section C.1.1).

C.1.1 More details on SQuAD2 Results on Different Styles

For homogeneous evaluation, Table C.2 shows the detailed performance when we apply SQuAD2-finetuned BERT on our NONCAT models. We found that supplementary training generally helps the NONCAT task in most setting except for NAMEONLYon SG-DST dataset. Furthermore, the SQuAD2 supplementary training helps more on the rich description than name-based description, because the questions in SQuAD2 share more distribution similary with natural language descriptions than the intent or slot names. Besides that, another

observation is that supplementary training helps more on unseen services, while slightly hurting the performance on the seen services.

C.2 Homogeneous and Heterogeneous Evaluation on Different Styles

Section 5.5.1 introduces the benchmarking description styles used in our dissertation. This appendix section shows more detailed evaluation results on both homogeneous and heterogeneous setting.

C.2.1 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.

C.2.1.1 Results on Intent Classification

The results on SG-DST dataset are shown in Table C.3. 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 Section 5.5.2, we believe the name template is good enough to describe the core functionality of an intent in SG-DST dataset.

C.2.1.2 Requested Slot

Table C.4 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.

C.2.1.3 Categorical Slot

The results on SG-DST and MULTIWOZ 2.2 dataset are shown in Table C.5. When creating MULTIWOZ 2.2 [221], 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 subtasks, we can further verify our guess when more datasets are created for the research of schema-guided dialog in the future.

C.2.1.4 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 C.6. Overall, the rich description performs better on both homogeneous and heterogeneous evaluations.

C.2.2 Qualitative Analysis On Heterogeneous Evaluation

We conduct qualitative analysis on heterogeneous evaluation on named-based description. Table C.7 shows how paraphrasing the named-based description impact on the categorical and non-categorical slot prediction tasks.

The first three 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 three 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 four 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.

Table C.1: Results of different supplementary training on SG-DST and MULTIWOZ 2.2 datasets.

		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
	$\Delta_{\textbf{SNLI}}$	**+0.51**	+0.02	**+0.68**	-0.19	+0.38	-0.38	-1.63	-2.87	-1.23	-4.7	-0.1	-6.25	+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
	$\Delta_{\textbf{SQuAD}}$	-1.81	-0.17	-1.32	-0.25	-0.01	-0.33	-2.87	-3.02	-5.17	**1.99**	-1.79	**3.25**	+0.04	-0.71	+0.41	**1.93**	-2.21	**+4.27**

Table C.2: Results on different description style on SG-DST and MULTIWOZ 2.2 dataset, when performing SQuAD2 supplementary training.

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 C.3: 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	*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 C.4: 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
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 C.5: 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	*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 C.6: 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.

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 C.7: 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.

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 C.8: Results on unseen service with heterogeneous description styles on SG-DST dataset. More results and qualitative analysis are in the appendix C.2

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**

REFERENCES

[1] N. A. Smith, "Linguistic structure prediction," *Synthesis lectures on human language technologies*, vol. 4, no. 2, pp. 1–274, 2011.

[2] R. Johansson and P. Nugues, "The effect of syntactic representation on semantic role labeling," in *Proceedings of the 22nd International Conference on Computational Linguistics*, Manchester, UK, 2008, pp. 393 – 400.

[3] D. Hovy, S. Tratz, and E. Hovy, "What's in a preposition? dimensions of sense disambiguation for an interesting word class," in *Coling 2010: Posters*, 2010, pp. 454–462.

[4] V. Punyakanok, D. Roth, and W.-t. Yih, "The importance of syntactic parsing and inference in semantic role labeling," *Computational Linguistics*, vol. 34, no. 2, pp. 257–287, 2008.

[5] N. S. Moosavi and M. Strube, "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*. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 193–203. [Online]. Available: https://aclanthology.org/D18-1018

[6] E. Strubell, P. Verga, D. Andor, D. Weiss, and A. McCallum, "Linguistically-informed self-attention for semantic role labeling," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 5027–5038. [Online]. Available: https://aclanthology.org/D18-1548

[7] S. R. Bowman, J. Gauthier, A. Rastogi, R. Gupta, C. D. Manning, and C. Potts, "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)*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 1466–1477. [Online]. Available: https://aclanthology.org/P16-1139

[8] L. Banarescu, C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider, "Abstract Meaning Representation for sembanking," in *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, Sofia, Bulgaria, Aug. 2013, pp. 178 – 186.

[9] P. Kingsbury and M. Palmer, "From TreeBank to PropBank," in *Proceedings of the 3rd International Conference on Language Resources and Evaluation*, Las Palmas, Spain, 2002, pp. 1989 – 1993.

[10] O. Abend and A. Rappoport, "Universal Conceptual Cognitive Annotation (UCCA)," in *Proceedings of the 51th Meeting of the Association for Computational Linguistics*, Sofia, Bulgaria, Aug. 2013, pp. 228 – 238.

[11] A. Ivanova, S. Oepen, L. Øvrelid, and D. Flickinger, "Who did what to whom?: A contrastive study of syntacto-semantic dependencies," in *Proceedings of the sixth linguistic annotation workshop*. Association for Computational Linguistics, 2012, pp. 2–11.

[12] J. Hajic, E. Hajicová, J. Panevová, P. Sgall, O. Bojar, S. Cinková, E. Fucíková, M. Mikulová, P. Pajas, J. Popelka *et al.*, "Announcing prague czech-english dependency treebank 2.0." in *LREC*, 2012, pp. 3153–3160.

[13] Y. Miyao, S. Oepen, and D. Zeman, "In-house: An ensemble of pre-existing off-the-shelf parsers," in *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, 2014, pp. 335–340.

[14] L. Wittgenstein, *Philosophical investigations*. John Wiley & Sons, 2010.

[15] H. Bunt, J. Alexandersson, J. Carletta, J.-W. Choe, A. C. Fang, K. Hasida, K. Lee, V. Petukhova, A. Popescu-Belis, L. Romary *et al.*, "Towards an iso standard for dialogue act annotation," in *Seventh conference on International Language Resources and Evaluation (LREC'10)*, 2010.

[16] C. J. Fillmore, "The case for case," in *Universals in Linguistic Theory*, E. Bach and R. T. Harms, Eds. London: Holt, Rinehart and Winston, 1968, pp. 1 – 88.

[17] D. G. Bobrow, R. M. Kaplan, M. Kay, D. A. Norman, H. Thompson, and T. Winograd, "Gus, a frame-driven dialog system," *Artificial intelligence*, vol. 8, no. 2, pp. 155–173, 1977.

[18] M. Tanana, K. A. Hallgren, Z. E. Imel, D. C. Atkins, and V. Srikumar, "A comparison of natural language processing methods for automated coding of motivational interviewing," *Journal of substance abuse treatment*, vol. 65, pp. 43–50, 2016.

[19] P. Budzianowski, T.-H. Wen, B.-H. Tseng, I. Casanueva, S. Ultes, O. Ramadan, and M. Gasic, "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*, 2018, pp. 5016–5026.

[20] F. Dernoncourt and J. Y. Lee, "Pubmed 200k rct: a dataset for sequential sentence classification in medical abstracts," *IJCNLP 2017*, p. 308, 2017.

[21] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill *et al.*, "On the opportunities and risks of foundation models," *arXiv preprint arXiv:2108.07258*, 2021.

[22] L. Song, D. Gildea, Y. Zhang, Z. Wang, and J. Su, "Semantic neural machine translation using amr," *Transactions of the Association for Computational Linguistics*, vol. 7, pp. 19–31, 2019.

[23] F. Liu, J. Flanigan, S. Thomson, N. Sadeh, and N. A. Smith, "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*, 2015, pp. 1077–1086.

[24] P. Kapanipathi, I. Abdelaziz, S. Ravishankar, S. Roukos, A. Gray, R. F. Astudillo, M. Chang, C. Cornelio, S. Dana, A. Fokoue-Nkoutche *et al.*, "Leveraging abstract meaning representation for knowledge base question answering," in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 2021, pp. 3884–3894.

[25] S. J. Gershman, *What Makes Us Smart*. Princeton University Press, 2021.

[26] E. S. Spelke, "Principles of object perception," *Cognitive Science*, vol. 14, pp. 29–56, 1990.

[27] E. Bienenstock, S. Geman, and D. Potter, "Compositionality, mdl priors, and object recognition," in *NIPS*, 1996.

[28] B. Rehder, "A causal-model theory of conceptual representation and categorization." *Journal of experimental psychology. Learning, memory, and cognition*, vol. 29 6, pp. 1141–59, 2003.

[29] H. F. Harlow, "The formation of learning sets." *Psychological review*, vol. 56, no. 1, p. 51, 1949.

[30] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman, "Building machines that learn and think like people," *Behavioral and Brain Sciences*, vol. 40, 2016.

[31] J. Baxter, "A model of inductive bias learning," *Journal of artificial intelligence research*, vol. 12, pp. 149–198, 2000.

[32] D. H. Wolpert, W. G. Macready *et al.*, "No free lunch theorems for search," Technical Report SFI-TR-95-02-010, Santa Fe Institute, Tech. Rep., 1995.

[33] T. M. Mitchell, *The need for biases in learning generalizations*. Department of Computer Science, Laboratory for Computer Science Research . . . , 1980.

[34] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[35] R. Zhang, "Making convolutional networks shift-invariant again," in *International conference on machine learning*, 2019, pp. 7324–7334.

[36] G. Cheng, P. Zhou, and J. Han, "Rifd-cnn: Rotation-invariant and fisher discriminative convolutional neural networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2884–2893.

[37] M. Schuster and K. Nakajima, "Japanese and korean voice search," in *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2012, pp. 5149–5152.

[38] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, pp. 1715–1725.

[39] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "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)*, 2019, pp. 4171–4186.

[40] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[41] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[42] L. D. Gregoire Mesnil, Xiaodong He and Y. Bengio, "Investigation of recurret neural-network architectures and learning methods for language understanding," in *Proceedings of Interspeech*, 2013.

[43] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *NIPS Deep Learning and Representation Learning Workshop*, 2014.

[44] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 5998–6008. [Online]. Available: http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

[45] Z. S. Harris, "Distributional structure," *Word*, vol. 10, no. 2-3, pp. 146–162, 1954.

[46] K. Werling, G. Angeli, and C. Manning, "Robust Subgraph Generation Improves Abstract Meaning Representation Parsing," *arXiv.org*, pp. 982–991, Jun. 2015.

[47] W. Foland and J. H. Martin, "Abstract Meaning Representation parsing using LSTM recurrent neural networks," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 463–472. [Online]. Available: https://aclanthology.org/P17-1043

[48] C. Wang and N. Xue, "Getting the most out of amr parsing," in *Proceedings of the 2017 conference on empirical methods in natural language processing*, 2017, pp. 1257–1268.

[49] D. Gildea, N. Xue, X. Peng, and C. Wang, "Addressing the Data Sparsity Issue in Neural AMR Parsing." *EACL*, 2017.

[50] J. Cao, Y. Zhang, A. Youssef, and V. Srikumar, "Amazon at mrp 2019: Parsing meaning representations with lexical and phrasal anchoring," in *Proceedings of the Shared Task on MRP at the CoNLL 2019*, 2019.

[51] G. Papandreou and A. Yuille, "Perturb-and-map random fields: Using discrete optimization to learn and sample from energy models–iccv 2011 paper supplementary material–," *ICCV*, 2011.

[52] G. Mena, D. Belanger, S. Linderman, and J. Snoek, "Learning latent permutations with gumbel-sinkhorn networks," *arXiv preprint arXiv:1802.08665*, 2018.

[53] O. Abend and A. Rappoport, "Universal conceptual cognitive annotation (ucca)," in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2013, pp. 228–238.

[54] S. Gupta, R. Shah, M. Mohit, A. Kumar, and M. Lewis, "Semantic parsing for task oriented dialog using hierarchical representations," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, Oct.-Nov. 2018, pp. 2787–2792. [Online]. Available: https://www.aclweb.org/anthology/D18-1300

[55] J. Cao, M. Tanana, Z. Imel, E. Poitras, D. Atkins, and V. Srikumar, "Observing dialogue in therapy: Categorizing and forecasting behavioral codes," in *Proceedings of ACL 2019*, 2019.

[56] W. Miller and S. Rollnick, "Motivational interviewing: Preparing people for change," *Journal for Healthcare Quality*, vol. 25, no. 3, p. 46, 2003.

[57] W. R. Miller and S. Rollnick, *Motivational interviewing: Helping people change*. Guilford press, 2012.

[58] M. Marcus, G. Kim, M. A. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, K. Katz, and B. Schasberger, "The Penn Treebank: Annotating predicate argument structure," in *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*, 1994. [Online]. Available: https://aclanthology.org/H94-1020

[59] M.-W. Chang, L. Ratinov, and D. Roth, "Structured learning with constrained conditional models," *Machine learning*, vol. 88, no. 3, pp. 399–431, 2012.

[60] D. Roth and W.-t. Yih, "Global inference for entity and relation identification via a linear programming formulation," *Introduction to statistical relational learning*, pp. 553–580, 2007.

[61] S. H. Bach, M. Broecheler, B. Huang, and L. Getoor, "Hinge-loss markov random fields and probabilistic soft logic," *Journal of Machine Learning Research*, vol. 18, pp. 1–67, 2017.

[62] T. Li and V. Srikumar, "Augmenting Neural Networks with First-order Logic," in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

[63] M. L. Pacheco and D. Goldwasser, "Modeling content and context with deep relational learning," *Transactions of the Association for Computational Linguistics*, vol. 9, pp. 100–119, 2021.

[64] P. Yin, C. Zhou, J. He, and G. Neubig, "Structvae: Tree-structured latent variable models for semi-supervised semantic parsing," *ACL*, 2018.

[65] C. Corro and I. Titov, "Learning latent trees with stochastic perturbations and differentiable dynamic programming," *ACL*, 2019.

[66] D. Belanger and A. McCallum, "Structured prediction energy networks," in *International Conference on Machine Learning*. PMLR, 2016, pp. 983–992.

[67] L. Tu and K. Gimpel, "Learning approximate inference networks for structured prediction," *ICLR*, 2018.

[68] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of documentation*, 1972.

[69] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[70] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[71] M. Baroni, G. Dinu, and G. Kruszewski, "Don't 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*, vol. 1, 2014.

[72] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[73] S. Li, J. Zhu, and C. Miao, "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*, 2015, pp. 1599–1609.

[74] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems (NIPS)*, 2013, pp. 3111–3119.

[75] K. Ethayarajh, "How contextual are contextualized word representations? comparing the geometry of bert, elmo, and gpt-2 embeddings," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 55–65.

[76] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "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*, 2018.

[77] M. E. Peters, S. Ruder, and N. A. Smith, "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)*, 2019, pp. 7–14.

[78] K. Toutanvoa and C. D. Manning, "Enriching the knowledge sources used in a maximum entropy part-of-speech tagger," in *2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora*, 2000, pp. 63–70.

[79] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, 2003, pp. 252–259.

[80] T. Shin, Y. Razeghi, R. L. Logan IV, E. Wallace, and S. Singh, "Autoprompt: Eliciting knowledge from language models with automatically generated prompts," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 4222–4235.

[81] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, "Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing," *arXiv preprint arXiv:2107.13586*, 2021.

[82] K. Murphy, "Probabilistic machine learning: Advanced topics," 2022.

[83] A. G. Wilson and P. Izmailov, "Bayesian deep learning and a probabilistic perspective of generalization," *Advances in neural information processing systems*, vol. 33, pp. 4697–4708, 2020.

[84] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[85] T. Kasami, "An efficient recognition and syntax-analysis algorithm for context-free languages," *Coordinated Science Laboratory Report no. R-257*, 1966.

[86] D. H. Younger, "Recognition and parsing of context-free languages in time n3," *Information and control*, vol. 10, no. 2, pp. 189–208, 1967.

[87] J. Cocke, *Programming languages and their compilers: Preliminary notes*. New York University, 1969.

[88] Y.-J. Chu, "On the shortest arborescence of a directed graph," *Scientia Sinica*, vol. 14, pp. 1396–1400, 1965.

[89] J. Edmonds *et al.*, "Optimum branchings," *Journal of Research of the national Bureau of Standards B*, vol. 71, no. 4, pp. 233–240, 1967.

[90] D. Roth and W.-t. Yih, "Integer linear programming inference for conditional random fields," in *Proceedings of the 22nd international conference on Machine learning*, 2005, pp. 736–743.

[91] J. Berant, V. Srikumar, P.-C. Chen, A. Vander Linden, B. Harding, B. Huang, P. Clark, and C. D. Manning, "Modeling biological processes for reading comprehension," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1499–1510.

[92] J. R. Finkel, T. Grenager, and C. D. Manning, "Incorporating non-local information into information extraction systems by gibbs sampling," in *Proceedings of the 43rd annual meeting of the association for computational linguistics (ACL'05)*, 2005, pp. 363–370.

[93] S. Singh, M. Wick, and A. McCallum, "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*, 2012, pp. 1104–1113.

[94] H. Daumé, J. Langford, and D. Marcu, "Search-based structured prediction," *Machine learning*, vol. 75, no. 3, pp. 297–325, 2009.

[95] S. Ross, G. Gordon, and D. Bagnell, "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*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.

[96] K.-W. Chang, A. Krishnamurthy, A. Agarwal, H. Daumé III, and J. Langford, "Learning to search better than your teacher," in *International Conference on Machine Learning*. PMLR, 2015, pp. 2058–2066.

[97] A. M. Rush and M. Collins, "A tutorial on dual decomposition and lagrangian relaxation for inference in natural language processing," *Journal of Artificial Intelligence Research*, vol. 45, pp. 305–362, 2012.

[98] T. Werner and D. Pruˇsa, "The power of lp relaxation for map inference," *Advanced Structured Prediction*, p. 19, 2014.

[99] J. K. Baker, "Trainable grammars for speech recognition," *The Journal of the Acoustical Society of America*, vol. 65, no. S1, pp. S132–S132, 1979.

[100] J. Weizenbaum, "ELIZA – a computer program for the study of natural language communication between man and machine," *Communications of the ACM*, vol. 9, no. 1, pp. 36–45, 1966.

[101] N. A. Smith and J. Eisner, "Contrastive estimation: Training log-linear models on unlabeled data," in *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, 2005, pp. 354–362.

[102] J. D. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proceedings of the Eighteenth International Conference on Machine Learning*, 2001, pp. 282–289.

[103] B. T. C. G. D. Koller, "Max-margin markov networks," *Advances in neural information processing systems*, vol. 16, p. 25, 2004.

[104] J. Binder, K. Murphy, and S. Russell, "Space-efficient inference in dynamic probabilistic networks," *Bclr*, vol. 1, p. t1, 1997.

[105] T. Koo, A. Globerson, X. Carreras, and M. Collins, "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)*. Prague, Czech Republic: Association for Computational Linguistics, Jun. 2007, pp. 141–150. [Online]. Available: https://aclanthology.org/D07-1015

[106] Y. Liu and M. Lapata, "Learning structured text representations," *Transactions of the Association for Computational Linguistics*, vol. 6, pp. 63–75, 2018.

[107] S. Oepen, O. Abend, J. Hajič, D. Hershcovich, M. Kuhlmann, T. O'Gorman, N. Xue, J. Chun, M. Straka, and Z. Urešová, "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*, Hong Kong, China, 2019, pp. 1 – 27.

[108] K. R. Beesley and L. Karttunen, "Finite-state morphology: Xerox tools and techniques," *CSLI, Stanford*, 2003.

[109] G. A. Miller, *WordNet: An electronic lexical database*. MIT press, 1998.

[110] C. F. Baker, C. J. Fillmore, and J. B. Lowe, "The berkeley framenet project," in *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, 1998, pp. 86–90.

[111] M. Palmer, D. Gildea, and P. Kingsbury, "The proposition bank: An annotated corpus of semantic roles," *Computational linguistics*, vol. 31, no. 1, pp. 71–106, 2005.

[112] M. Collins, "Head-driven statistical models for natural language parsing," *Computational linguistics*, vol. 29, no. 4, pp. 589–637, 2003.

[113] L. Carlson, D. Marcu, and M. E. Okurowski, "Building a discourse-tagged corpus in the framework of rhetorical structure theory," in *Current and new directions in discourse and dialogue*. Springer, 2003, pp. 85–112.

[114] F. Wolf and E. Gibson, "Representing discourse coherence: A corpus-based study," *Computational linguistics*, vol. 31, no. 2, pp. 249–287, 2005.

[115] R. Prasad, N. Dinesh, A. Lee, E. Miltsakaki, L. Robaldo, A. K. Joshi, and B. L. Webber, "The penn discourse treebank 2.0." in *LREC*. Citeseer, 2008.

[116] A. Ivanova, S. Oepen, L. Øvrelid, and D. Flickinger, "Who did what to whom? A contrastive study of syntacto-semantic dependencies," in *Proceedings of the 6th Linguistic Annotation Workshop*, Jeju, Republic of Korea, 2012, pp. 2 – 11.

[117] M. Marcus, B. Santorini, and M. A. Marcinkiewicz, "Building a large annotated corpora of English. The Penn Treebank," *Computational Linguistics*, vol. 19, pp. 313 – 330, 1993.

[118] D. Flickinger, V. Kordoni, Y. Zhang, A. Branco, K. Simov, P. Osenova, C. Carvalheiro, F. Costa, and S. Castro, "ParDeepBank. Multiple parallel deep treebanking," in *Proceedings of the 11th International Workshop on Treebanks and Linguistic Theories*. Lisbon, Portugal: Edições Colibri, 2012, pp. 97 – 108.

[119] S. Oepen, D. Flickinger, K. Toutanova, and C. D. Manning, "LinGO Redwoods. A rich and dynamic treebank for HPSG," *Research on Language and Computation*, vol. 2, no. 4, pp. 575 – 596, 2004.

[120] E. M. Bender, D. Flickinger, S. Oepen, W. Packard, and A. Copestake, "Layers of interpretation. On grammar and compositionality," in *Proceedings of the 11th International Conference on Computational Semantics*, London, UK, 2015, pp. 239 – 249.

[121] C. Pollard and I. A. Sag, *Head-Driven Phrase Structure Grammar*, ser. Studies in Contemporary Linguistics. Chicago, USA: The University of Chicago Press, 1994.

[122] S. Oepen and J. T. Lønning, "Discriminant-based MRS banking," in *Proceedings of the 5th International Conference on Language Resources and Evaluation*, Genoa, Italy, 2006, pp. 1250 – 1255.

[123] P. Sgall, E. Hajičová, and J. Panevová, *The Meaning of the Sentence and Its Semantic and Pragmatic Aspects*. Dordrecht, The Netherlands: D. Reidel Publishing Company, 1986.

[124] Z. Urešová, E. Fučíková, and J. Šindlerová, "Czengvallex: a bilingual czech-english valency lexicon," *The Prague Bulletin of Mathematical Linguistics*, vol. 105, no. 1, p. 17, 2016.

[125] J. Flanigan, S. Thomson, J. G. Carbonell, C. Dyer, and N. A. Smith, "A Discriminative Graph-Based Parser for the Abstract Meaning Representation." *ACL*, 2014.

[126] C. Wang, N. Xue, and S. Pradhan, "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)*, pp. 1–6, Jul. 2015.

[127] Y. Artzi, K. Lee, and L. Zettlemoyer, "Broad-coverage CCG Semantic Parsing with AMR." *EMNLP*, 2015.

[128] M. Pust, U. Hermjakob, K. Knight, D. Marcu, and J. May, "Using Syntax-Based Machine Translation to Parse English into Abstract Meaning Representation," *arXiv.org*, Apr. 2015.

[129] X. Peng, L. Song, and D. Gildea, "A Synchronous Hyperedge Replacement Grammar based approach for AMR parsing," in *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2015, pp. 32–41.

[130] I. Konstas, S. Iyer, M. Yatskar, Y. Choi, and L. Zettlemoyer, "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)*, 2017, pp. 146–157.

[131] N. Pourdamghani, Y. Gao, U. Hermjakob, and K. Knight, "Aligning English Strings with Abstract Meaning Representation Graphs," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2014, pp. 425–429.

[132] W.-T. Chen and M. Palmer, "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*, vol. 1, 2017, pp. 558–567.

[133] O. Abend and A. Rappoport, "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)*, 2017, pp. 77–89.

[134] B. L. Burke, C. W. Dunn, D. C. Atkins, and J. S. Phelps, "The emerging evidence base for motivational interviewing: A meta-analytic and qualitative inquiry," *Journal of Cognitive Psychotherapy*, vol. 18, no. 4, pp. 309–322, 2004.

[135] R. K. Martins and D. W. McNeil, "Review of motivational interviewing in promoting health behaviors," *Clinical psychology review*, vol. 29, no. 4, pp. 283–293, 2009.

[136] B. W. Lundahl, C. Kunz, C. Brownell, D. Tollefson, and B. L. Burke, "A meta-analysis of motivational interviewing: Twenty-five years of empirical studies," *Research on social work practice*, vol. 20, no. 2, pp. 137–160, 2010.

[137] C. S. Schwalbe, H. Y. Oh, and A. Zweben, "Sustaining motivational interviewing: a meta-analysis of training studies." *Addiction (Abingdon, England)*, vol. 109, no. 8, pp. 1287–94, aug 2014. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/24661345

[138] A. Stolcke, K. Ries, N. Coccaro, E. Shriberg, R. Bates, D. Jurafsky, P. Taylor, R. Martin, C. V. Ess-Dykema, and M. Meteer, "Dialogue act modeling for automatic tagging and recognition of conversational speech," *Computational linguistics*, vol. 26, no. 3, pp. 339–373, 2000.

[139] D. Jurafsky and J. H. Martin, *Speech and language processing*. Pearson, 2019.

[140] W. R. Miller, T. B. Moyers, D. Ernst, and P. Amrhein, "Manual for the motivational interviewing skill code (misc)," *Unpublished manuscript. Albuquerque: Center on Alcoholism, Substance Abuse and Addictions, University of New Mexico*, 2003.

[141] B. Xiao, D. Can, J. Gibson, Z. E. Imel, D. C. Atkins, P. G. Georgiou, and S. S. Narayanan, "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*, 2016, pp. 908–912.

[142] A. Aghajanyan, J. Maillard, A. Shrivastava, K. Diedrick, M. Haeger, H. Li, Y. Mehdad, V. Stoyanov, A. Kumar, M. Lewis *et al.*, "Conversational semantic parsing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 5026–5035.

[143] J. Cheng, D. Agrawal, H. M. Alonso, S. Bhargava, J. Driesen, F. Flego, D. Kaplan, D. Kartsaklis, L. Li, D. Piraviperumal *et al.*, "Conversational semantic parsing for dialog state tracking," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 8107–8117.

[144] J. Andreas, J. Bufe, D. Burkett, C. Chen, J. Clausman, J. Crawford, K. Crim, J. DeLoach, L. Dorner, J. Eisner *et al.*, "Task-oriented dialogue as dataflow synthesis," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 556–571, 2020.

[145] S. Oepen, M. Kuhlmann, Y. Miyao, D. Zeman, D. Flickinger, J. Hajič, A. Ivanova, and Y. Zhang, "SemEval 2014 Task 8. Broad-coverage semantic dependency parsing," in *Proceedings of the 8th International Workshop on Semantic Evaluation*, Dublin, Ireland, 2014, p. 63 – 72.

[146] S. Oepen, M. Kuhlmann, Y. Miyao, D. Zeman, S. Cinková, D. Flickinger, J. Hajič, and Z. Urešová, "SemEval 2015 Task 18. Broad-coverage semantic dependency parsing," in *Proceedings of the 9th International Workshop on Semantic Evaluation*, Bolder, CO, USA, 2015, p. 915 – 926.

[147] J. May, "SemEval-2016 Task 8: Meaning Representation Parsing," in *Proceedings of SemEval*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2016, pp. 1063–1073.

[148] D. Hershcovich, Z. Aizenbud, L. Choshen, E. Sulem, A. Rappoport, and O. Abend, "SemEval-2019 task 1: Cross-lingual semantic parsing with UCCA," in *Proceedings of the 13th International Workshop on Semantic Evaluation*. Minneapolis, Minnesota, USA: Association for Computational Linguistics, Jun. 2019, pp. 1–10. [Online]. Available: https://www.aclweb.org/anthology/S19-2001

[149] I. Szubert, A. Lopez, and N. Schneider, "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)*, 2018, pp. 1169–1180.

[150] C. Lyu and I. Titov, "Amr parsing as graph prediction with latent alignment," *ACL*, 2018.

[151] S. Oepen and J. T. Lønning, "Discriminant-based mrs banking." in *LREC*, 2006, pp. 1250–1255.

[152] L. Banarescu, C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider, "Abstract Meaning Representation for Sembanking." *LAW@ACL*, 2013.

[153] P. F. Brown, V. J. D. Pietra, S. A. D. Pietra, and R. L. Mercer, "The mathematics of statistical machine translation: Parameter estimation," *Computational linguistics*, vol. 19, no. 2, pp. 263–311, 1993.

[154] S. Zhang, X. Ma, K. Duh, and B. Van Durme, "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, Jul. 2019.

[155] M.-T. Luong, I. Sutskever, Q. V. Le, O. Vinyals, and W. Zaremba, "Addressing the rare word problem in neural machine translation," *IJCNLP*, 2014.

[156] T. Dozat and C. D. Manning, "Deep biaffine attention for neural dependency parsing," *ICLR*, 2016.

[157] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *ICLR*, 2013.

[158] S. Kullback and R. A. Leibler, "On Information and Sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79 – 86, 1951. [Online]. Available: https://doi.org/10.1214/aoms/1177729694

[159] W. Jiang, Y. Zhang, Z. Li, and M. Zhang, "Hlt@ suda at semeval 2019 task 1: Ucca graph parsing as constituent tree parsing," *arXiv preprint arXiv:1903.04153*, 2019.

[160] N. Kitaev and D. Klein, "Constituency parsing with a self-attentive encoder," *arXiv preprint arXiv:1805.01052*, 2018.

[161] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the International Conference on Learning Representations*, 2015.

[162] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," *arXiv preprint arXiv:1802.05365*, 2018.

[163] W. Che, L. Dou, Y. Xu, Y. Wang, Y. Liu, and T. Liu, "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*, Hong Kong, China, 2019, pp. 76 – 85.

[164] D. Hershcovich and O. Arviv, "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*, Hong Kong, China, 2019, pp. 28 – 39.

[165] S. Oepen and D. Flickinger, "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*, Hong Kong, China, 2019, pp. 40 – 44.

[166] Z. Li, H. Zhao, Z. Zhang, R. Wang, M. Utiyama, and E. Sumita, "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*, Hong Kong, China, 2019, pp. 45 – 54.

[167] L. Donatelli, M. Fowlie, J. Groschwitz, A. Koller, M. Lindemann, M. Mina, and P. Weißenhorn, "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*, Hong Kong, China, 2019, pp. 66 – 75.

[168] K. M. Colby, *Artificial Paranoia: A Computer Simulation of Paranoid Process*. Pergamon Press, 1975.

[169] V. Pérez-Rosas, R. Mihalcea, K. Resnicow, S. Singh, L. Ann, K. J. Goggin, and D. Catley, "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*, vol. 1, 2017, pp. 1128–1137.

[170] X. Huang, L. Liu, K. Carey, J. Woolley, S. Scherer, and B. Borsari, "Modeling temporality of human intentions by domain adaptation," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018, pp. 696–701.

[171] J. Schatzmann, K. Georgila, and S. Young, "Quantitative evaluation of user simulation techniques for spoken dialogue systems," in *6th SIGdial Workshop on DISCOURSE and DIALOGUE*, 2005.

[172] R. Lowe, N. Pow, I. V. Serban, and J. Pineau, "The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems," in *Proceedings of SIGDIAL*, 2015.

[173] K. Yoshino, C. Hori, J. Perez, L. F. D'Haro, L. Polymenakos, C. Gunasekara, W. S. Lasecki, J. Kummerfeld, M. Galley, C. Brockett, J. Gao, B. Dolan, S. Gao, T. K. Marks, D. Parikh, and D. Batra, "The 7th dialog system technology challenge," *arXiv preprint*, 2018.

[174] Z. E. Imel, D. D. Caperton, M. Tanana, and D. C. Atkins, "Technology-enhanced human interaction in psychotherapy." *Journal of counseling psychology*, vol. 64, no. 4, p. 385, 2017.

[175] J. Zhang, J. P. Chang, C. Danescu-Niculescu-Mizil, L. Dixon, Y. Hua, N. Thain, and D. Taraborelli, "Conversations Gone Awry: Detecting Early Signs of Conversational Failure," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018.

[176] J. Li, M.-T. Luong, and D. Jurafsky, "A hierarchical neural autoencoder for paragraphs and documents," *arXiv preprint arXiv:1506.01057*, 2015.

[177] A. Sordoni, Y. Bengio, H. Vahabi, C. Lioma, J. Grue Simonsen, and J.-Y. Nie, "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*. ACM, 2015, pp. 553–562.

[178] I. V. Serban, A. Sordoni, Y. Bengio, A. C. Courville, and J. Pineau, "Building end-to-end dialogue systems using generative hierarchical neural network models." in *AAAI*, vol. 16, 2016, pp. 3776–3784.

[179] A. Galassi, M. Lippi, and P. Torroni, "Attention, please! a critical review of neural attention models in natural language processing," *arXiv preprint arXiv:1902.02181*, 2019.

[180] M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Bidirectional attention flow for machine comprehension," in *ICLR*, 2016.

[181] W. Wang, N. Yang, F. Wei, B. Chang, and M. Zhou, "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)*, vol. 1, 2017, pp. 189–198.

[182] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[183] P. Roy-Byrne, K. Bumgardner, A. Krupski, C. Dunn, R. Ries, D. Donovan, I. I. West, C. Maynard, D. C. Atkins, M. C. Graves *et al.*, "Brief intervention for problem drug use in safety-net primary care settings: a randomized clinical trial," *Jama*, vol. 312, no. 5, pp. 492–501, 2014.

[184] J. S. Baer, E. A. Wells, D. B. Rosengren, B. Hartzler, B. Beadnell, and C. Dunn, "Agency context and tailored training in technology transfer: A pilot evaluation of motivational interviewing training for community counselors," *Journal of substance abuse treatment*, vol. 37, no. 2, pp. 191–202, 2009.

[185] S. J. Tollison, C. M. Lee, C. Neighbors, T. A. Neil, N. D. Olson, and M. E. Larimer, "Questions and reflections: the use of motivational interviewing microskills in a peer-led brief alcohol intervention for college students," *Behavior Therapy*, vol. 39, no. 2, pp. 183–194, 2008.

[186] C. Neighbors, C. M. Lee, D. C. Atkins, M. A. Lewis, D. Kaysen, A. Mittmann, N. Fossos, I. M. Geisner, C. Zheng, and M. E. Larimer, "A randomized controlled trial of event-specific prevention strategies for reducing problematic drinking associated with 21st birthday celebrations." *Journal of consulting and clinical psychology*, vol. 80, no. 5, p. 850, 2012.

[187] C. M. Lee, J. R. Kilmer, C. Neighbors, D. C. Atkins, C. Zheng, D. D. Walker, and M. E. Larimer, "Indicated prevention for college student marijuana use: A randomized controlled trial." *Journal of consulting and clinical psychology*, vol. 81, no. 4, p. 702, 2013.

[188] C. M. Lee, C. Neighbors, M. A. Lewis, D. Kaysen, A. Mittmann, I. M. Geisner, D. C. Atkins, C. Zheng, L. A. Garberson, J. R. Kilmer *et al.*, "Randomized controlled trial of a spring break intervention to reduce high-risk drinking." *Journal of consulting and clinical psychology*, vol. 82, no. 2, p. 189, 2014.

[189] D. C. Atkins, M. Steyvers, Z. E. Imel, and P. Smyth, "Scaling up the evaluation of psychotherapy: evaluating motivational interviewing fidelity via statistical text classification," *Implementation Science*, vol. 9, no. 1, p. 49, 2014.

[190] D. Can, D. C. Atkins, and S. S. Narayanan, "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*, 2015.

[191] M. Honnibal and I. Montani, "spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing," *To appear*, 2017.

[192] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proc. of NAACL*, 2018.

[193] J. Gibson, D. Can, P. Georgiou, D. C. Atkins, and S. S. Narayanan, "Attention networks for modeling behaviors in addiction counseling," in *Proceedings of the 2016 Conference of the International Speech Communication Association INTERSPEECH*, 2017.

[194] D. Can, P. G. Georgiou, D. C. Atkins, and S. S. Narayanan, "A case study: Detecting counselor reflections in psychotherapy for addictions using linguistic features," in *Thirteenth Annual Conference of the International Speech Communication Association*, 2012.

[195] M. Tanana, K. Hallgren, Z. Imel, D. Atkins, P. Smyth, and V. Srikumar, "Recursive neural networks for coding therapist and patient behavior in motivational interviewing," in *Proceedings of the 2nd Workshop on Computational Linguistics and Clinical Psychology: From Linguistic Signal to Clinical Reality*, 2015, pp. 71–79.

[196] F. Weber, L. Manganaro, B. Peskin, and E. Shriberg, "Using prosodic and lexical information for speaker identification," in *Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on*. IEEE, 2002, pp. I–141.

[197] Y. Kim, "Convolutional neural networks for sentence classification," in *Proceedings of Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1746–1751.

[198] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," in *Advances in neural information processing systems*, 2015.

[199] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015 conference on empirical methods in natural language processing*, 2015.

[200] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2016, pp. 1480–1489.

[201] I. V. Serban, T. Klinger, G. Tesauro, K. Talamadupula, B. Zhou, Y. Bengio, and A. C. Courville, "Multiresolution recurrent neural networks: An application to dialogue response generation." in *AAAI*, 2017, pp. 3288–3294.

[202] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[203] S. Wang and J. Jiang, "Learning natural language inference with lstm," *arXiv preprint arXiv:1512.08849*, 2015.

[204] S. Sukhbaatar, J. Weston, and R. Fergus, "End-to-end memory networks," in *Advances in neural information processing systems*, 2015.

[205] F. Liu and J. Perez, "Gated end-to-end memory networks," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, 2017.

[206] X. Liu, Y. Shen, K. Duh, and J. Gao, "Stochastic answer networks for machine reading comprehension," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 2018, pp. 1694–1704. [Online]. Available: http://aclweb.org/anthology/P18-1157

[207] W.-T. Hsu, C.-K. Lin, M.-Y. Lee, K. Min, J. Tang, and M. Sun, "A unified model for extractive and abstractive summarization using inconsistency loss," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018.

[208] X. Zhou, L. Li, D. Dong, Y. Liu, Y. Chen, W. X. Zhao, D. Yu, and H. Wu, "Multi-turn response selection for chatbots with deep attention matching network," in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, 2018.

[209] X. Zhou, D. Dong, H. Wu, S. Zhao, D. Yu, H. Tian, X. Liu, and R. Yan, "Multi-view response selection for human-computer conversation," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016, pp. 372–381.

[210] B. Byrne, K. Krishnamoorthi, C. Sankar, A. Neelakantan, D. Duckworth, S. Yavuz, B. Goodrich, A. Dubey, A. Cedilnik, and K.-Y. Kim, "Taskmaster-1: Toward a realistic and diverse dialog dataset," *arXiv preprint arXiv:1909.05358*, 2019.

[211] P. Shah, D. Hakkani-Tür, B. Liu, and G. Tür, "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)*. New Orleans - Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 41–51. [Online]. Available: https://www.aclweb.org/anthology/N18-3006

[212] N. Mrkšić, D. Ó. Séaghdha, T.-H. Wen, B. Thomson, and S. Young, "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)*, 2017, pp. 1777–1788.

[213] C.-S. Wu, A. Madotto, E. Hosseini-Asl, C. Xiong, R. Socher, and P. Fung, "Transferable multi-domain state generator for task-oriented dialogue systems," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 808–819.

[214] E. Hosseini-Asl, B. McCann, C.-S. Wu, S. Yavuz, and R. Socher, "A simple language model for task-oriented dialogue," *arXiv preprint arXiv:2005.00796*, 2020.

[215] A. Rastogi, X. Zang, S. Sunkara, R. Gupta, and P. Khaitan, "Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset," *AAAI 2019*, 2019.

[216] ——, "Schema-guided dialogue state tracking task at dstc8," *Workshop on DSTC8, AAAI 2020*, 2020.

[217] J. Cao and Y. Zhang, "A comparative study on schema-guided dialogue state tracking," in *Proceedings of NAACL 2021*, 2021.

[218] J.-G. Zhang, K. Hashimoto, C.-S. Wu, Y. Wan, P. S. Yu, R. Socher, and C. Xiong, "Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking," *arXiv*, pp. arXiv–1910, 2019.

[219] G.-L. Chao and I. Lane, "Bert-dst: Scalable end-to-end dialogue state tracking with bidirectional encoder representations from transformer," *arXiv preprint arXiv:1907.03040*, 2019.

[220] V. Noroozi, Y. Zhang, E. Bakhturina, and T. Kornuta, "A fast and robust bert-based dialogue state tracker for schema-guided dialogue dataset," *Workshop on Conversational Systems Towards Mainstream Adoption at KDD 2020*, 2020.

[221] X. Zang, A. Rastogi, S. Sunkara, R. Gupta, J. Zhang, and J. Chen, "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*. Online: Association for Computational Linguistics, Jul. 2020, pp. 109–117. [Online]. Available: https://www.aclweb.org/anthology/2020.nlp4convai-1.13

[222] M. Eric, R. Goel, S. Paul, A. Sethi, S. Agarwal, S. Gao, A. Kumar, A. Goyal, P. Ku, and D. Hakkani-Tur, "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*, 2020, pp. 422–428.

[223] T. Han, X. Liu, R. Takanobu, Y. Lian, C. Huang, W. Peng, and M. Huang, "Multiwoz 2.3: A multi-domain task-oriented dataset enhanced with annotation corrections and co-reference annotation," *arXiv e-prints*, pp. arXiv–2010, 2020.

[224] P. Shah, D. Hakkani-Tür, G. Tür, A. Rastogi, A. Bapna, N. Nayak, and L. Heck, "Building a conversational agent overnight with dialogue self-play," *arXiv preprint arXiv:1801.04871*, 2018.

[225] J. F. Kelley, "An iterative design methodology for user-friendly natural language office information applications," *ACM Transactions on Information Systems (TOIS)*, vol. 2, no. 1, pp. 26–41, 1984.

[226] S. Humeau, K. Shuster, M.-A. Lachaux, and J. Weston, "Poly-encoders: Architectures and pre-training strategies for fast and accurate multi-sentence scoring," in *ICLR*, 2019.

[227] J. Phang, T. Févry, and S. R. Bowman, "Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks," *arXiv preprint arXiv:1811.01088*, 2018.

[228] M. Namazifar, A. Papangelis, G. Tur, and D. Hakkani-Tür, "Language model is all you need: Natural language understanding as question answering," *arXiv preprint arXiv:2011.03023*, 2020.

[229] S. Gao, A. Sethi, S. Agarwal, T. Chung, D. Hakkani-Tur, and A. A. AI, "Dialog state tracking: A neural reading comprehension approach," in *20th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 2019, p. 264.

[230] R. Sennrich, O. Firat, K. Cho, A. Birch, B. Haddow, J. Hitschler, M. Junczys-Dowmunt, S. Läubli, A. V. Miceli Barone, J. Mokry, and M. Nădejde, "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*. Valencia, Spain: Association for Computational Linguistics, Apr. 2017, pp. 65–68. [Online]. Available: https://www.aclweb.org/anthology/E17-3017

[231] N. Vedula, N. Lipka, P. Maneriker, and S. Parthasarathy, "Open intent extraction from natural language interactions," in *Proceedings of The Web Conference 2020*, 2020, pp. 2009–2020.

[232] A. Bordes, J. Weston, and N. Usunier, "Open question answering with weakly supervised embedding models," in *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2014, pp. 165–180.

[233] R. Lowe, N. Pow, I. Serban, and J. Pineau, "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*, 2015.

[234] Y. Wu, W. Wu, C. Xing, M. Zhou, and Z. Li, "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)*, 2017, pp. 496–505.

[235] L. Yang, M. Qiu, C. Qu, J. Guo, Y. Zhang, W. B. Croft, J. Huang, and H. Chen, "Response ranking with deep matching networks and external knowledge in information-seeking conversation systems," *arXiv preprint arXiv:1805.00188*, 2018.

[236] S. Kim, S. Yang, G. Kim, and S.-W. Lee, "Efficient dialogue state tracking by selectively overwriting memory," *arXiv preprint arXiv:1911.03906*, 2019.

[237] C. Naik, A. Gupta, H. Ge, M. Lambert, and R. Sarikaya, "Contextual slot carryover for disparate schemas," *Proc. Interspeech 2018*, pp. 596–600, 2018.

[238] M. Henderson, B. Thomson, and S. Young, "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)*, 2014, pp. 292–299.

[239] J. Williams, A. Raux, and M. Henderson, "The dialog state tracking challenge series: A review," *Dialogue & Discourse*, vol. 7, no. 3, pp. 4–33, 2016.

[240] B. Peng, C. Li, J. Li, S. Shayandeh, L. Liden, and J. Gao, "Soloist: Few-shot task-oriented dialog with a single pre-trained auto-regressive model," *arXiv*, pp. arXiv–2005, 2020.

[241] H. Lee, J. Lee, and T.-Y. Kim, "Sumbt: Slot-utterance matching for universal and scalable belief tracking," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 5478–5483.

[242] S. Gao, S. Agarwal, T. Chung, D. Jin, and D. Hakkani-Tur, "From machine reading comprehension to dialogue state tracking: Bridging the gap," *arXiv preprint arXiv:2004.05827*, 2020.

[243] N. Schucher, S. Reddy, and H. de Vries, "The power of prompt tuning for low-resource semantic parsing," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2022, pp. 148–156.

[244] H. Ye, N. Zhang, S. Deng, X. Chen, H. Chen, F. Xiong, X. Chen, and H. Chen, "Ontology-enhanced prompt-tuning for few-shot learning," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 778–787.

[245] T. O'Gorman, M. Regan, K. Griffitt, U. Hermjakob, K. Knight, and M. Palmer, "AMR beyond the sentence: the multi-sentence AMR corpus," in *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 3693–3702. [Online]. Available: https://aclanthology.org/C18-1313

[246] T. Naseem, A. Blodgett, S. Kumaravel, T. O'Gorman, Y.-S. Lee, J. Flanigan, R. Astudillo, R. Florian, S. Roukos, and N. Schneider, "DocAMR: Multi-sentence AMR representation and evaluation," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, Jul. 2022, pp. 3496–3505. [Online]. Available: https://aclanthology.org/2022.naacl-main.256

[247] L. Carlson, D. Marcu, and M. E. Okurovsky, "Building a discourse-tagged corpus in the framework of Rhetorical Structure Theory," in *Proceedings of the Second SIGdial Workshop on Discourse and Dialogue*, 2001. [Online]. Available: https://aclanthology.org/W01-1605

[248] Y. Tay, M. Dehghani, S. Abnar, Y. Shen, D. Bahri, P. Pham, J. Rao, L. Yang, S. Ruder, and D. Metzler, "Long range arena: A benchmark for efficient transformers," in *International Conference on Learning Representations*, 2020.

[249] A. Gu, K. Goel, and C. Re, "Efficiently modeling long sequences with structured state spaces," in *International Conference on Learning Representations*, 2021.

[250] L. Beinborn, T. Botschen, and I. Gurevych, "Multimodal grounding for language processing," in *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, pp. 2325–2339. [Online]. Available: https://aclanthology.org/C18-1197

[251] X. Ju, D. Zhang, R. Xiao, J. Li, S. Li, M. Zhang, and G. Zhou, "Joint multi-modal aspect-sentiment analysis with auxiliary cross-modal relation detection," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 4395–4405.

[252] D. Zhang, S. Wei, S. Li, H. Wu, Q. Zhu, and G. Zhou, "Multi-modal graph fusion for named entity recognition with targeted visual guidance," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, 2021, pp. 14 347–14 355.

[253] H. Singh, A. Nasery, D. Mehta, A. Agarwal, J. Lamba, and B. V. Srinivasan, "MIMOQA: Multimodal input multimodal output question answering," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Online: Association for Computational Linguistics, Jun. 2021, pp. 5317–5332. [Online]. Available: https://aclanthology.org/2021.naacl-main.418

[254] J. Lu, D. Batra, D. Parikh, and S. Lee, "Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks," *Advances in neural information processing systems*, vol. 32, 2019.

[255] Y. Ling, J. Yu, and R. Xia, "Vision-language pre-training for multimodal aspect-based sentiment analysis," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Dublin, Ireland: Association

for Computational Linguistics, May 2022, pp. 2149–2159. [Online]. Available: https://aclanthology.org/2022.acl-long.152

[256] G. Marra, F. Giannini, M. Diligenti, and M. Gori, "Integrating learning and reasoning with deep logic models," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2019, pp. 517–532.

[257] X. Lu, S. Welleck, P. West, L. Jiang, J. Kasai, D. Khashabi, R. Le Bras, L. Qin, Y. Yu, R. Zellers, N. A. Smith, and Y. Choi, "NeuroLogic a*esque decoding: Constrained text generation with lookahead heuristics," in *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Seattle, United States: Association for Computational Linguistics, Jul. 2022, pp. 780–799. [Online]. Available: https://aclanthology.org/2022.naacl-main.57

[258] J. Pearl, "Causal inference," *Causality: objectives and assessment*, pp. 39–58, 2010.

[259] M. Glymour, J. Pearl, and N. P. Jewell, *Causal inference in statistics: A primer*. John Wiley & Sons, 2016.

[260] H. Wang and D.-Y. Yeung, "A survey on bayesian deep learning," *ACM Computing Surveys (CSUR)*, vol. 53, no. 5, pp. 1–37, 2020.

[261] M. Assran, N. Loizou, N. Ballas, and M. Rabbat, "Stochastic gradient push for distributed deep learning," in *International Conference on Machine Learning*. PMLR, 2019, pp. 344–353.

[262] X. Huang, Z. Shen, S. Li, Z. Liu, H. Xianghong, J. Wicaksana, E. Xing, and K.-T. Cheng, "Sdq: Stochastic differentiable quantization with mixed precision," in *International Conference on Machine Learning*. PMLR, 2022, pp. 9295–9309.

[263] D. Paul*, J. Cao*, F. Li, and V. Srikumar, "Database workload characterization with query plan encoders," *VLDB'22*, 2021.

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

[265] J. Houck, T. Moyers, W. Miller, L. Glynn, and K. Hallgren, "Motivational interviewing skill code (misc) version 2.5," 2012.