Dialogue Meaning Representation for Task-Oriented Dialogue Systems

Xiangkun Hu[1,*], Junqi Dai[2,*,†], Hang Yan[2], Yi Zhang[1], Qipeng Guo[1], Xipeng Qiu[2], Zheng Zhang[1]
[1]Amazon AWS AI
[2]School of Computer Science, Fudan University
{xiangkhu, yizhngn, gqipeng, zhaz}@amazon.com
jqdai22@m.fudan.edu.cn
{hyan19, xpqiu}@fudan.edu.cn

Abstract

Dialogue meaning representation formulates natural language utterance semantics in their conversational context in an explicit and machine-readable form. Previous work typically follows the intent-slot framework, which is easy for annotation yet limited in scalability for complex linguistic expressions. A line of works alleviates the representation issue by introducing hierarchical structures but challenging to express complex compositional semantics, such as negation and coreference. We propose Dialogue Meaning Representation (DMR), a pliable and easily extendable representation for task-oriented dialogue. Our representation contains a set of nodes and edges to represent rich compositional semantics. Moreover, we propose an inheritance hierarchy mechanism focusing on domain extensibility. Additionally, we annotated DMR-FastFood, a multi-turn dialogue dataset with more than 70k utterances, with DMR. We propose two evaluation tasks to evaluate different dialogue models and a novel coreference resolution model GNNCoref for the graph-based coreference resolution task. Experiments show that DMR can be parsed well with pre-trained Seq2Seq models, and GNNCoref outperforms the baseline models by a large margin.[1]

1 Introduction

A task-oriented dialogue (TOD) system aims to serve users to accomplish tasks in specific domains with interactive conversations. The modeling of converting natural language semantics in their conversational context into a machine-readable structured representation, also known as the dialogue meaning representation, is at the core of TOD system research. A meaning representation framework sets the stage for natural language understanding (NLU) and allows the system to communicate with other downstream components such as databases or web service APIs.

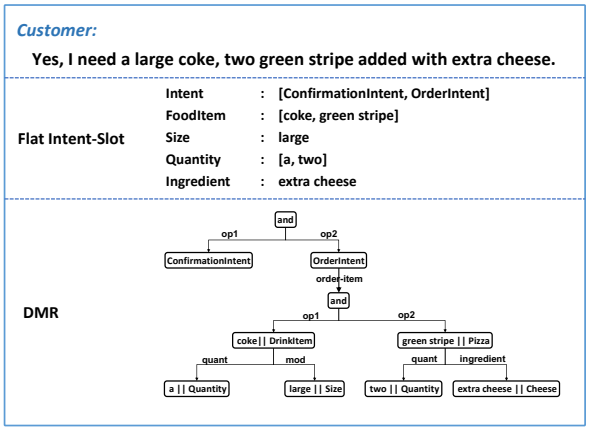


Figure 1: An example of a customer's utterance for food ordering. The flat intent-slot schema can not align the food items ("coke" and "green stripe") with the modifiers ("large", "a", "two" and "extra cheese") in conjunction constructions. These multiple conjunctions and modifiers require a good meaning representation to reveal the relations between attributes (e.g., size, quantity) and their corresponding entities. Here, we propose DMR with an example shown in the lower part of the figure, a meaning representation for TOD, which can resolve such compositional semantics.

Derived from the theoretical framework of Fillmore (1968), and wildly adopted in dialogue system designs as early as Bobrow et al. (1977), the classic flat intent-slot schema represents an utterance into one specific intent with several associated slots. Such schemas are convenient in annotation but limited in the expression for compositional semantics, such as conjunction, modification, negation, and coreference across dialogue turns. These complex patterns are not at all uncommon in real-world use cases. Take the fast-food domain data from MultiDoGO dataset (Peskov et al., 2019) as an example, where an agent is required to extract infor-

*Equal contribution.
† Work done during internship at Amazon Shanghai AI Lab.
[1]The dataset and code are available at https://github.com/amazon-research/dialogue-meaning-representation.

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 223 - 237

mation about ordering food (e.g., food name, quantity, size, and ingredients) from the conversation with the customer, about 12.3% of the utterances contain multi-intent and 11.2% are involved with coreference semantics. However, the flat intent-slot schema leaves these semantics uncovered. Moreover, 8.7% of the ordering utterances contain multiple objects and modifiers. For example in Figure 1, with the flat intent-slot schema, the extracted food items "coke" and "green stripe" (a pizza name) cannot be aligned with the size "large", the quantities "a" and "two", and the ingredient "extra cheese".

The vanilla flat intent-slot schema needs to be extended to support compositional semantics. Gupta et al. (2018) proposes TOP in the form of a hierarchical parsing tree to allow representation for nested intents. Its successor, SB-TOP (Aghajanyan et al., 2020), further simplifies the structure and supports coreference. Cheng et al. (2020) introduces TreeDST, which is also a tree-structured dialogue state representation to allow high compositionality and integrate domain knowledge. These studies imply the trend pointing to a balance in a better expression for complex compositional semantics and a lighter structure for extensibility.

This paper proposes Dialogue Meaning Representation (DMR) that significantly extends the intent-slot framework. DMR is a rooted, directed acyclic graph (DAG) composed of nodes of *Intent*, *Entity* and pre-defined *Operator* and *Keyword*, as well as edges between them. Entity is an extension of slot which wraps the slot value with the specific slot type defined in external knowledge. Such design allows arbitrary complex compositionality between slots and keeps the potential for type constraint. Operator and Keyword are components to represent linguistic knowledge (i.e. general semantics) such as conjunction, negation, quantification, coreference, etc. The details of DMR can be found in Section 2, and the example comparing DMR with flat intent-slot representation is shown in Figure 1. As described later in Section 2, many of the key designs are inspired by AMR(Banarescu et al., 2013) but specialized for TOD. Thus, DMR can be considered a dialect of AMR. From this perspective, DMR is powerful enough and easily extendable for TOD applications.

Moreover, DMR is capable of adapting to different domains. Unlike previous works, DMR utilizes a domain-agnostic ontology to define the structural constraints and representations of general semantics. It allows chatbot developers to derive domain-specific ontology from this for their applications through the *Inheritance Hierarchy* mechanism. This design improves both generalization and normalization of DMR.

To validate our idea, we propose a dataset, DMR-FastFood, with 7194 dialogues and 70328 annotated utterances. This dataset is extensively annotated with more linguistic semantics, including 16087 conjunctions and 557 negations, significantly more than other related datasets. We developed and evaluated a few baseline models to pinpoint where the challenges lie. We further propose GNNCoref for the coreference resolution task on DMR. In general, DMR parsing is not difficult, especially with a pre-trained model, though graphs with more complex (and often deeper) structures are naturally more challenging. Moreover, experiments show that GNNCoref performs better compared to baseline models.

2 Dialogue Meaning Representation

This section describes the structure of the DMR graph, the domain-agnostic ontology, and the representation of general semantics.

2.1 DMR Ontology

DMR ontology defines the nodes, the edges, and the rules for constructing DMR graphs. It also describes the inheritance hierarchy mechanism. DMR is a rooted directed acyclic graph with node and edge labels. Figure 2 shows an example of a DMR graph from the fast-food domain.

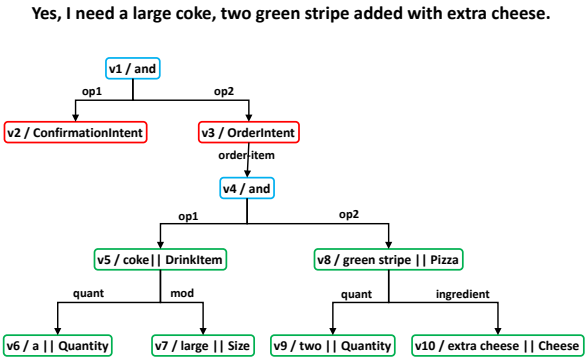


Figure 2: An example of DMR graph from fast food domain. Nodes in different types are in different box colors, namely, `Intent`, `Operator` and `Entity`.

2.1.1 Nodes

Different from the general purpose of the predicates and concepts defined in AMR, DMR utilizes the

specialized designed nodes for TOD. There are four types of nodes in DMR:

- `Intent`, denotes the intention of the speaker, such as `OrderIntent` in Figure 2;
- `Entity`, denotes the objects mentioned in the utterance. Generally, it is formed with "`<lexical_value>` || `<canonical_value>` || `Entity`", where `<lexical_value>` specifies the surface form value from the utterance, and `<canonical_value>` is the predefined value for the entity in ontology. `<lexical_value>` and `<canonical_value>` are optional;[2]
- `Operator`, supports compositional constructions, such as `and` for conjunction, and `reference` for cross-turn coreference. Details are described in the next section;
- `Keyword`, specifies keywords for some special semantics, such as "-" for negation.

Each node (except keywords) are assigned with an identifier as AMR does, such as "v1" for node `OrderIntent` in Figure 2. And the root node of a DMR graph is restricted to be an `Intent` node or a conjunction operator `and` for packing multiple intentions.

2.1.2 Edges

The nodes in DMR graph are linked with directed edges. For every edge, the node types it can reach are pre-defined. Moreover, all types of nodes have pre-defined arguments in the ontology, which constrains the argument type, namely the edge here, and node types the edge can reach. In a specific DMR graph, some arguments defined in ontology may not appear. For example, in fast food domain, intent `OrderIntent` has one argument `order-item` (see Figure 2). Entity type `DrinkItem` has pre-defined arguments `quant`, `mod` and `ingredient`, but in the example of Figure 2, no ingredients of the coke are mentioned in the utterance, thus, the edge `ingredient` is not shown in the graph.

2.1.3 Inheritance Hierarchy

DMR is featured with inheritance hierarchy. With this mechanism, chatbot developers can derive domain-specific ontology easily and organize it hierarchically. For example, in the fast-food domain, we can derive the ontology like:

```
Intent ⟵ OrderIntent | PaymentIntent | ThankYouIntent
Entity ⟵ FoodItem | DrinkItem
FoodItem ⟵ Pizza | Burger | Sandwich
```

It defines three intents `OrderIntent`, `PaymentIntent` and `ThankYouIntent`; two base entity types `FoodItem` and `DrinkItem`; and three `FoodItem` types `Pizza`, `Burger` and `Sandwich` that inherits from `FoodItem`. The derived intents and entity types inherit their parents' arguments by default. In the above example, `FoodItem` and `DrinkItem` inherit arguments of `Entity` such as `mod` and `quant`; `Pizza`, `Burger` and `Sandwich` inherit arguments of `FoodItem` such as `ingredient`, and so on. We describe more details on this in Appendix C.

With inheritance hierarchy, the domain-specific and domain-agnostic knowledge are well separated: the general semantics that is common in all domains, such as quantification, and negation, are defined in the domain-agnostic ontology, while the domain-specific part inherits these representations and can focus on the application. Further, it reduces the burden of constructing ontology, as the intent and entity types inherit their parents' arguments by default, and the ontology is organized hierarchically.

2.2 Compositional Semantics

Here we describe the general compositional semantics defined in the domain-agnostic ontology. It is worth noting that we do not cover all the general semantics in TOD, though this set can be extended in the future. The examples used are taken from the fast-food domain; we just show a sub-graph of the DMRs and omit the variables for simplicity.

Modification refers to the semantics where a specific adjective modifies some entities. Modifiers, such as the size or color of an object, are non-essential descriptive content compared to regular arguments (Dowty, 1982, 1989, 2003). The modification semantics is expressed by `mod`, for example:

Veg Out, large

```
(Veg Out || Pizza
  :mod (large || Size))
```

Quantification is also a common semantics in TOD. Quantification is expressed by the edge labeled with `quant`. T3 in Figure 3 shows such case similar to the following example:

2 burgers

```
(burgers || Burger
   :quant (2 || Quantity))
```

[2] In our DMR-FastFood dataset, there are no pre-defined canonical values, we instead use the form "`<lexical_value>` || `Entity`" for simplicity.

Conjunction refers to the semantic construction that connects elements, e.g., *"A and B"*. Inspired by AMR, DMR resolves conjunction with `Operator` node, such as `and`, `or`. Conjunction is important when multiple intentions or cumulative entities are expressed. A nested conjunction is like follows:

sandwich and soda, thank you.

```
(and
    :op1 (OrderIntent
            :order-item (and
                            :op1 (sandwich || Sandwich)
                            :op2 (soda || DrinkItem)))
    :op2 ThankYouIntent)
```

Cross-Turn Coreference is a common phenomenon in dialogues. Since DMR represents semantics in a graph, the implementation of coreference in DMR is to link corresponding nodes, rather than simple text mentions. DMR introduces a special operator `Reference` and edge `refer` to represent cross-turn coreferences. The reference node is with form "reference || `<lexical_value>`", along with an argument `refer` that points to another node. For example, T7 in Figure 3 contains a reference node "v2" that points to "T:5 N:v4", which means this node refers to the node "v4" in the T5 turn's DMR.

Negation is the construction which ties a negative polarity to another element, reversing the state of an affair or discontinuing an act. For instance, the utterance *"Please cancel the burger"* conveys a cancel action to an order. Inspired by AMR, we notice that negation can be seen as a binding act attached to an element. Therefore, instead of representing negation via additional `Intent`, DMR resolves negation by edge `polarity` and keyword "-". For example, T9 in Figure 3, node "v4" is negated. Moreover, one tricky issue about negation is its scope. A negation act to an order item can be confused as one to an (enclosing) order intent, leading to an unintended "overkill". At this stage of development, we make a simplification and restrict the negation to attach only to `Entity`.

3 Related Work

There is a rising interest in developing more flexible representation for TOD other than the slots representation (Bobrow et al., 1977). In this section, we briefly introduce them and compare the most related works with DMR.

AMR and Dialogue-AMR Using AMR for semantic parsing has been studied from a very early time (Banarescu et al., 2013). There are several

T1 *Customer:* **Hello, good morning!**

```
(v1 / OpeningGreeting)
```

T2 *Agent:* **Hello, what would you like to order today?**

T3 *Customer:* **I want a veg out and one GREEN STRIP, each with 4 slices.**

```
(v1 / OrderIntent
   :order-item (v2 / and
      :op1 (v3 / veg out || Pizza
         :quant (v4 / a || Quantity)
         :mod (v5 / 4 slices || Unit))
      :op2 (v6 / GREEN STRIP || Pizza
         :quant (v7 / one || Quantity)
         :mod (v8 / 4 slices || Unit))))
```

T4 *Agent:* **Awesome! would you like to have some sandwich? I must say we have some sizzling sandwiches...**

T5 *Customer:* **Okey, one sandwich.**

```
(v1 / and
   :op1 (v2 / ConfirmationIntent)
   :op2 (v3 / OrderIntent
      :order-item (v4 / sandwich || Sandwich
         :quant (v5 / one || Quantity))))
```

T6 *Agent:* **I'll take that order. How about some toppings for sandwich?**

T7 *Customer:* **emmm, please add salted butter and onions on it.**

```
(v1 / OrderIntent
   :order-item (v2 / reference
      :refer T:5 N:v4
      :ingredient (v3 / and
         :op1 (v4 / salted butter || Ingredient)
         :op2 (v5 / onions || Veggies))))
```

T8 *Agent:* **Perfect! Is there anything else that you'd like to order?**

T9 *Customer:* **Oh, sorry, I want no toppings, instead order me a soda. that's all.**

```
(v1 / OrderIntent
   :order-item (v2 / and
      :op1 (v3 / reference
         :refer T:5 N:v4
         :ingredient (v4 / reference||toppings
            :refer (v5 / and
               :op1 T:7 N:v4
               :op2 T:7 N:v5)
            :polarity -))
      :op2 (v6 / soda || DrinkItem
         :quant (v7 / a || Quantity))))
```

Figure 3: A dialogue example taken from fast food domain. The customer turns are annotated with DMRs.

works that apply AMR to dialogue systems. Bai et al. (2021) model dialogue state with AMR for chit-chat. Bonial et al. (2019) extended AMR for human-robot dialogues, and further formalize it as Dialogue-AMR (Bonial et al., 2020, 2021). Dialogue-AMR represents both the illocutionary force and the propositional content of the utterance. Compared to these works, DMR focuses on TOD specifically with extended node types for TOD description. DMR is intent-centric, and only captures semantics defined in the ontology of the intents. Further, the design of inheritance hierarchy aims at a better domain generalization to support a broad range of applications. AMR and Dialogue-AMR are closely related to DMR, so we have more detailed comparisons in Appendix B to show the

differences between them.

Compositional Intent-Slot Some recent works focus on the compositional intent-slot framework, such as TOP (Gupta et al., 2018), SB-TOP (Aghajanyan et al., 2020) and TreeDST (Cheng et al., 2020). These formalizations are much more powerful than the flat intent-slot schema. Generally, they focus less on how to get representations for different domains and give fewer descriptions on how nodes/edges are connected. On the contrary, the design which contains both domain-agnostic and domain-specific parts allows DMR to be applied and extended to different domains while assuring the maintenance of the representation structure in the meantime. From this point of view, DMR is designed to provide services to different businesses. This different focus of the application scenarios marks the key difference between DMR and these compositional intent-slot representations.

Programs Many efforts have been devoted to explore the representation in programs (Price, 1990; Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Liang et al., 2011). Though powerful expressiveness, they are hard in annotation which makes it limited in proposing large-scale dialogue datasets. Recently, some works such as SMCalFlow (Semantic Machines et al., 2020) and TOC (Campagna et al., 2021) and ThingTalk (Lam et al., 2022; Campagna et al., 2022) proposes to use executable dialogue states for TOD. To this end, the representation itself is also a specially-designed programming language. While executing, both database operations and response generation are performed by the program at the same time. DMR keeps the dialogue state architecture, and leaves the implementation of business logic to the user applications.

4 Data

We use the fast food domain data from the MultiDoGO dataset proposed by Peskov et al. (2019). We annotate all the customers' utterances with the redefined ontology. We call the annotated dataset **DMR-FastFood**. This dataset contains 7k annotated dialogues, and each dialogue has 18.5 turns on average, which is much more than other datasets. Further, there are 7k references, 16k conjunctions and 557 negations annotated. The annotation process, statistics and comparison with related datasets are described in Appendix A.

5 NLU with DMR

As NLU tasks of the flat intent-slot representation including *Intent Classification* and *Slot Filling*, NLU tasks under the DMR framework are to extract DMR graphs from the customer's utterances. Given a customer's utterance x_i and the dialogue context $(x_0, \cdots, x_{i-1})$, the NLU tasks are to predict the DMR graph g_i. In this section, we introduce NLU tasks with DMR and proposed models.

5.1 Tasks

Though most of the DMRs can be predicted by a semantic parsing model, the turns that have cross-turn coreferences, namely *referring turns*, are not the case. The reference nodes in the referring turns need to be resolved to link to their referent nodes – nodes that are assigned with variables – in DMRs from the dialogue context. Parsing DMRs and resolving coreferences for the referring turns at the same time is not a trivial task. Thus, in this work we split NLU with DMR into two sub-tasks: DMR Parsing and Coreference Resolution.

DMR Parsing aims to parse a customer utterance into a DMR graph, *without* resolving the reference nodes. This semantic parsing task is similar to the NLU task in most related works, including TOP, SB-TOP, TreeDST and SMCalFlow.

Coreference Resolution resolves the reference nodes predicted by DMR parsing. Differing from traditional text-based coreference resolution, which links referring expressions to their antecedents in texts, this task is defined as follows: for each reference node n_r in a referring turn's DMR g_t, the task is to predict whether n_r and a given candidate node $n_c \in \{g_0, \cdots, g_{t-1}\}$ are coreferred.

5.2 Models

Our overall framework is composed of two stages. One is to parse graphs from the utterance by a Seq2Seq model, then is to resolve coreferences based on a GNN model.

5.2.1 DMR Parsing Model

The conventional approach to semantic parsing task is the Seq2Seq architecture which inputs the utterances and outputs the linearized tree or graph. This approach is also applied by SMCalFlow, SB-TOP, and Rongali et al. (2020). We utilize Seq2Seq architecture for DMR parsing as well, and restrict the decoder vocabulary to get more reasonable results.

The details of our Seq2Seq model are described as follows:

Input We concatenate the utterances to be parsed and the dialogue context – which are the customer's and agent's utterances in previous turns – to form the model input according to their order. Specifically, the model takes the input sequence $r_{i-c}||x_{i-c}, \cdots, r_j||x_j, \cdots, r_{i-1}||x_{i-1}, r_i||x_i$, where x_i is the customer's utterance to be parsed; x_j is the customer's or agent's utterance in the dialogue context; c is the context size. r_j is the role tag which can be `= customer:` or `agent:`, for the customer's and the agent's turn, respectively.

Output The output sequence of turn i is the linearized form of g_i. To linearize g_i takes three steps: 1) remove the `refer` edges of each reference node since the coreference resolution model will resolve them (see Section 5.2.2), 2) remove the variables of nodes and assign them back in the post-processing step for resolving references, and 3) convert the graph to a bracket expression. For example, the following DMR:

```
(v1 / OrderIntent
  :order-item
      (v2 / reference
          :refer (T:3 N:v1)
          :mod (v3 / large || Size)))
```

is converted to

```
( OrderIntent ( :order-item ( reference ( :mod ( large ||
      Size ) ) ) ) )
```

Since the `<lexical_value>` units are from the utterance contents, we constrain the decoding step to only generate tokens from either the schema or the utterance x_i. In our model, we mask the probabilities of non-relevant tokens to zeros in the output distribution at each decoding step. The output sequence is then parsed to DMRs with a shift-reduce parser, and the nodes are assigned with variables with Depth-First-Search.

Post-processing Though we restrict the decoder vocabulary, it does not guarantee the predicted sequence could be parsed to a valid graph because the sequence could be an invalid bracket expression. To tackle this issue, a rather flexible shift-reduce parser is applied to get a valid bracket expression by adding missing brackets or removing redundant brackets. If this rescue fails, the prediction is set to `OutOfDomainIntent`. And then we assign variables to the nodes in the recovered DMR graphs.

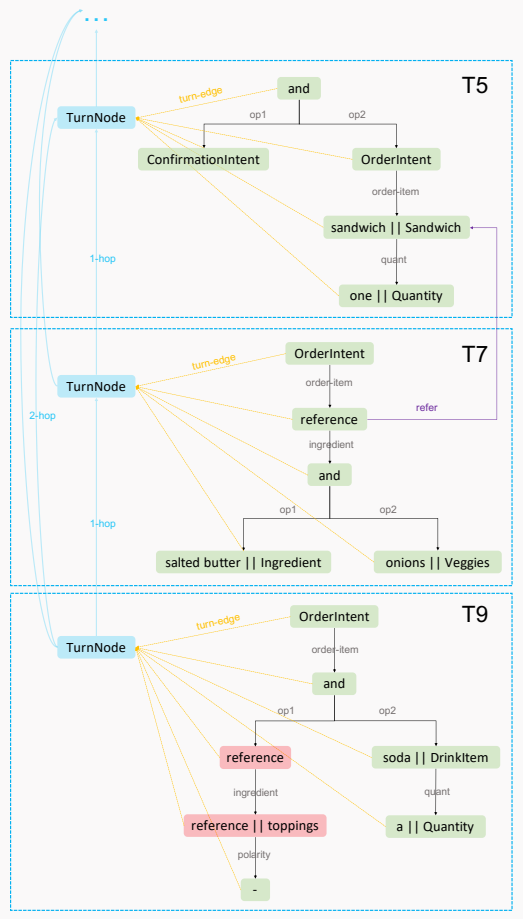


Figure 4: Dialogue Graph for GNNCoref model. The example used here is for resolving reference nodes (pink-colored nodes) in T9. The black arrows are edges within DMRs; the orange arrows are edges from DMR nodes to the turn nodes; the blue arrows are inter-turn edges that link DMRs through turn nodes; and the purple arrow is the edge for linking resolved coreferences in the context.

5.2.2 Coreference Resolution Model

For the graph-based coreference resolution task, we propose a GNN-based model **GNNCoref**. The following equations show how the model works:

$$G_t = \text{BuildGraph}(g_0, \cdots, g_t) \quad (1)$$

$$\mathbf{G}_t = \text{GNNEncoder}(G_t) \quad (2)$$

$$p(\text{corefer}|n_r, n_c) = \text{Classifier}(\mathbf{n}_r, \mathbf{n}_c) \quad (3)$$

First, for each referring turn t, a *Dialogue Graph* G_t is built; then the dialogue graph is encoded by a GNN encoder to encode the node features, and the encoded graph is denoted as $\mathbf{G}_t$; $\mathbf{n}_r$ and $\mathbf{n}_c$ in $\mathbf{G}_t$ are the encoded node features of the reference node and candidate node respectively, they are input into a binary classifier to predict whether they are core-

ferred. Next, we describe the details of these three modules.

Build Dialogue Graph The dialogue graph G_t is built by connecting the DMRs $(g_0, \cdots, g_j, \cdots, g_t)$ according to their order in the dialogue. Figure 4 shows the dialogue graph for resolving references in T9 in Figure 3. Specifically, we first build a *Turn Graph* for each turn with its DMR graph structure, and link each node to a turn-level global node which we call *turn node* with edge labeled `turn-edge`. The turn graphs are included in the dashed boxes in the figure. Then each turn node (e.g. for g_j) points to its k-hop ancestor (g_{j-k}) with the edge labeled k`-hop`. These inter-turn edges connect the DMRs to form one connected dialogue graph. Finally, if there are reference nodes already resolved in the dialogue context, e.g. the reference node in T7 in the figure, they are connected to their referent nodes with edge `refer`. In this way, the coreference resolution for the current referring turn depends on the previously resolved references, which brings more information for the task. Additionally, we add inverse edges for each edge to allow the message to pass bidirectionally. In dialogue graphs, every two turn graphs are linked through the turn nodes. Since the turn node links to all the nodes in that turn, every node in the dialogue graph can connect to each other after a three-step message passing so that all the context information can be encoded to the nodes.

GNN Encoder The graph is encoded with a 3-layer Relational Graph Convolutional Network (R-GCN) (Schlichtkrull et al., 2018) to encode the edge information to the nodes. The GNN encoder is designed to enhance the message passing among nodes and edges so that global context information can be captured in this process.

Classifier The binary classifier is a Multilayer Perceptron (MLP) with a Sigmoid activation for output. In the inference stage, we set a threshold β to determine the predictions. In our experiments, the value of β is tuned on the development set (see Appendix D.1 for details).

For a given reference node n_r, treating all the nodes in the context as its candidates is unwise, because n_r has the same entity type as its referents. However, the reference nodes are not labeled with types. According to our annotation guideline for the DMR-FastFood dataset described in Appendix A.2, all the reference nodes have the same incoming edge as the referents, thus, we choose the nodes in the context with the same incoming edge (or have the same incoming edge if there are more than one incoming edges) as n_r to be its candidates.

6 Experiments

We report experiments for DMR Parsing and Coreference Resolution separately, and the combined results on the complete DMR graphs. Further, we analyze the key factors that affect the model performance for the two tasks. The hyperparameters used and training details are listed in Appendix D.

6.1 DMR Parsing

The details of the DMR parsing models are described as follows:

BiLSTM+GloVe The encoder is a two-layer bi-directional LSTM (BiLSTM) and the decoder is a two-layer uni-directional LSTM. The word embeddings are initialized with GloVe840B-300d. This model contains 23M parameters.

RoBERTa-base The encoder is RoBERTa-base (Liu et al., 2019), and the decoder is a two-layer randomly initialized transformer with four attention heads and the same hidden size as the encoder. This model contains 183M parameters.

BART-base BART (Lewis et al., 2020) is a powerful pretrained encoder-decoder model for Seq2Seq tasks. We finetune the BART-base, directly for DMR Parsing. This model contains 140M parameters.

We use **Exact Match** accuracy to measure between predicted DMRs and the ground truths to evaluate the DMR Parsing results. To match the graphs semantically, we utilize the Smatch metric (Cai and Knight, 2013) designed for evaluating AMRs.[3] Two DMRs are exactly matched if their Smatch score equals 1. Setting context size $c = 1$, the performances of the DMR Parsing models are shown in Table 1a. The best results are achieved by BART-base model which are more than 10 points over the other two models, showing a well pretrained Seq2Seq model is essential for this task.

6.2 Coreference Resolution

To show the effectiveness of the proposed Coreference Resolution model, we compare the results with a heuristic **rule-based** method and a **MLP**

[3]The Smatch code we use is adapted from https://github.com/snowblink14/smatch

Model	Dev set	Test set
BiLSTM+GloVe	65.57	65.75
RoBERTa-base	69.24	68.23
BART-base	**82.56**	**83.39**

(a) DMR Parsing exact match accuracy ($c = 1$).

Method	Train set	Dev set	Test set
Rule	21.03	22.77	21.19
MLP	-	69.92	70.42
GNNCoref	-	**78.23**	**79.01**

(b) Coreference Resolution accuracy.

Model	Dev set	Test set
BiLSTM+GloVe	64.09	64.30
RoBERTa-base	67.57	66.69
BART-base	**80.73**	**81.47**

(c) The overall NLU results. The results are Exact Match of the completed DMR graphs whose reference nodes have been resolved by GNNCoref.

Table 1: Coreference resolution and the overall NLU performances on the DMR-FastFood dataset.

baseline model. The rule-based method selects the last DMR graph in the context and selects the candidate nodes in this DMR as the predicted referents. This distance-based heuristic is commonly used as an important feature in coreference resolution (Bengtson and Roth, 2008). In the MLP model, the features of the reference node and candidate nodes are the average of the word embeddings of their one-hop neighbor in the DMR graph and their own, and the features of the reference node and candidate node are concatenated to input into a 2-layer MLP classifier. For GNNCoref model, the initial node features for entity nodes and reference nodes are the average of GloVe6B-100d embeddings (Pennington et al., 2014) of all tokens (except for the variable) in the node. Other nodes are symbols defined in the DMR-FastFood ontology and their embeddings are randomly initialized. In our experiments, we use the DGL (Wang et al., 2019) implementation of R-GCN. All the methods are trained and evaluated on ground truth DMRs.

We measure coreference resolution with accuracy, i.e., a reference node is resolved correctly if the predicted turns and nodes are the same as the ground truth. Note that about 31.2% of the reference nodes in DMR-FastFood dataset have only one candidate which is directly their referent, we ignore these cases during training and evaluation. The results are listed in Table 1b. We can see that a simple heuristic rule can't handle this task well. Also, GNNCoref outperforms MLP well indicating the global dialogue context information captured with the graph structure is very useful compared to the local one-hop features.

6.3 The Overall NLU Performance

Combining the predictions of the DMR Parsing and Coreference Resolution model, we get the complete DMR graphs. The exact match of the complete DMRs are shown in Table 1c, the coreference resolution predictions used here are by GNNCoref reported in Table 1b. Comparing to the parsing results in Table 1a, the performance drops less than two points which proves the effectiveness of the two-step approach to this NLU task.

6.4 Error Analysis of DMR Parsing

We conduct error analysis to explore the difficulties and room for improvement of the DMR Parsing task. First, we analyze four types of errors:

- **Invalid Graph**, denotes that the predicted sequence cannot be parsed into a DMR graph with the shift-reduce parser. It is similar to Tree Validity used in Gupta et al. (2018).
- **Ontology Mismatch**, denotes that parts of the graph structure are not aligned with the definition in the ontology, e.g. edge `:order-item` points to a quantity, a `Pizza` entity with argument `:address`, etc.
- **Wrong Intent**, denotes wrong intents prediction. As intents are the first-class citizen which would directly affect the behavior of the chatbot agent. We consider a predicted DMR graph with wrong intents when the set of intents in it are not exactly matched with the golden DMR.
- **Compositional Errors**, denotes wrong or missing compositions in complex structures. In DMR-FastFood, we only care about the errors for `OrderIntent`. We extract the `OrderIntent` sub-graph from the golden DMR and the predicted DMR, and consider it as a compositional error if their Smatch score is not 1.

We take the error cases from the parsing results of BART-base model shown in Table 1a to conduct

Error Type	Dev set	Test set
Invalid Graph	3.7%	2.9%
Ontology Mismatch	2.3%	0.8%
Wrong Intent	31.1%	39.7%
Compositional Error	54.1%	46.0%

Table 2: Portions of errors in the error cases of BART-base model with $c = 1$.

	Dev set	Test set
GNNCoref	**78.23**	**79.01**
- *Turn Node*	68.45	70.56
- *Depend on Resolved Coref*	76.93	78.02
2-layer R-GCN	76.75	76.19
1-layer R-GCN	50.73	47.73

Table 3: Ablation study of GNNCoref.

the analysis. Table 2 shows the portions of the errors in the development and test sets respectively. We can see that the model can mostly generate valid graphs which also match the ontology. The main errors are compositional errors and wrong intent, and we observe that 79% of the wrong intent cases are the utterances with multiple intents. These results indicate that compositional generalization is the main bottleneck of Seq2Seq parsers.

Second, we analyze which factors affects the performance of the DMR parsing models. The details are listed in Appendix E. The main conclusions are intuitive: 1) DMR parsing is dependent on the dialogue context, and 2) longer utterances, deeper and larger DMR graphs make the parsing task harder.

6.5 Ablation Study of GNNCoref model

We conduct ablation studies to investigate the effectiveness of the two key designs in the dialogue graph for GNNCoref: 1) the global nodes *Turn Node* connecting DMRs through turn-level, and 2) depending on resolved coreferences in the context by adding `refer` edges as described in Section 5.2.2. We remove the turn nodes by connecting the DMRs through their root nodes instead, and remove the dependence on the resolved coreferences by removing `refer` edges. As shown in Table 3, both of the results are declined compared to original GNNCoref. However, without the help of turn node as a global node, performance drops sharply (From 79.01 to 70.56 on Test set), which indicates that the turn nodes are more critical.

In order to more rigorously prove the importance of the dialogue context information, we further conduct experiments of fewer R-GCN layers for GNNCoref. The results are shown in the last two rows of the table. Theoretically, With a 2-layer R-GCN, the nodes can only see information within their turns, thus, no dialogue context information is captured; the 1-layer R-GCN only captures one-hop information for each node. We can see that the performances declined which indicates that less captured context leads to lower performance.

7 Conclusion

In this paper, we focus on the representation with the expression ability of both complex compositional semantics and task-oriented semantics and propose DMR which is capable of representing complex linguistic constructions with a high transferability across domains. Moreover, we design the inheritance hierarchy which allows reusing, extending and inheriting node types to enable DMR scale to different domains easily. We annotated a large dataset on fast-food ordering domain, named DMR-FastFood, to incentive research on semantics parsing, which contains more than 70k utterances annotated with rich linguistic semantics. We conduct experiments on DMR parsing and coreference resolution tasks. Experimental results show that pre-trained Seq2Seq models could improve the DMR parsing results. We also propose a graph-based model, GNNCoref, for the coreference resolution on DMRs.

Limitations

In this work, we propose DMR and a dialogue dataset on the fast-food domain annotated with it. There are some limitations of this work we describe as follows. Firstly, DMR is designed to support a broad range of domains and applications for task-oriented dialogue. However, because of the human resources, and the observation that enough compositional semantics to begin with, such as conjunction, modification, and negation are contained in fast-food domain data, our dataset is only annotated on the fast-food domain for now. Secondly, we use DMR for natural language understanding task in TOD. In the real-world TOD systems, the complete pipeline also includes dialogue state tracking, dialogue policy and response generation that we do not deal with. Thirdly, annotating conversations with DMR is more expensive than annotating with

intents and slots. Few-shot learning and transfer learning for DMR parsing and coreference resolution could address this issue, and we leave them to future work. And Lastly, compared with the transition-based parser, the Seq2Seq-based semantic parser is not guaranteed to generate well-formed DMRs as it's not introduced with the inductive bias of the ontology, and it needs more data to train.

References

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

Xuefeng Bai, Yulong Chen, Linfeng Song, and Yue Zhang. 2021. Semantic representation for dialogue modeling. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4430–4445, Online. Association for Computational Linguistics.

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

Eric Bengtson and Dan Roth. 2008. Understanding the value of features for coreference resolution. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 294–303, Honolulu, Hawaii. Association for Computational Linguistics.

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

Claire Bonial, Mitchell Abrams, David Traum, and Clare Voss. 2021. Builder, we have done it: Evaluating & extending dialogue-AMR NLU pipeline for two collaborative domains. In *Proceedings of the 14th International Conference on Computational Semantics (IWCS)*, pages 173–183, Groningen, The Netherlands (online). Association for Computational Linguistics.

Claire Bonial, Lucia Donatelli, Mitchell Abrams, Stephanie M. Lukin, Stephen Tratz, Matthew Marge, Ron Artstein, David Traum, and Clare Voss. 2020. Dialogue-AMR: Abstract Meaning Representation for dialogue. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 684–695, Marseille, France. European Language Resources Association.

Claire Bonial, Lucia Donatelli, Stephanie M. Lukin, Stephen Tratz, Ron Artstein, David Traum, and Clare Voss. 2019. Augmenting Abstract Meaning Representation for human-robot dialogue. In *Proceedings of the First International Workshop on Designing Meaning Representations*, pages 199–210, Florence, Italy. Association for Computational Linguistics.

Shu Cai and Kevin Knight. 2013. Smatch: an evaluation metric for semantic feature structures. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 748–752, Sofia, Bulgaria. Association for Computational Linguistics.

Giovanni Campagna, Sina Semnani, Ryan Kearns, Lucas Jun Koba Sato, Silei Xu, and Monica Lam. 2022. A few-shot semantic parser for Wizard-of-Oz dialogues with the precise ThingTalk representation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 4021–4034, Dublin, Ireland. Association for Computational Linguistics.

Giovanni Campagna, Sina J. Semnani, Ryan Kearns, Lucas Jun Koba Sato, Silei Xu, and Monica S. Lam. 2021. Skim : Few-shot conversational semantic parsers with formal dialogue contexts.

Jianpeng Cheng, Devang Agrawal, Héctor Martínez Alonso, Shruti Bhargava, Joris Driesen, Federico Flego, Dain Kaplan, Dimitri Kartsaklis, Lin Li, Dhivya Piraviperumal, Jason D. Williams, Hong Yu, Diarmuid Ó Séaghdha, and Anders Johannsen. 2020. Conversational semantic parsing for dialog state tracking. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8107–8117, Online. Association for Computational Linguistics.

David Dowty. 1982. Grammatical relations and montague grammar. In *The nature of syntactic representation*, pages 79–130. Springer.

David Dowty. 2003. The dual analysis of adjuncts/complements in categorial grammar. *Modifying adjuncts*, 33.

David R Dowty. 1989. On the semantic content of the notion of 'thematic role'. In *Properties, types and meaning*, pages 69–129. Springer.

Charles J. Fillmore. 1968. The case for the case. In E. Bach and R. Harms, editors, *Universals in Linguistic Theory*. Rinehart and Winston, New York.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

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

Monica S. Lam, Giovanni Campagna, Mehrad Moradshahi, Sina J. Semnani, and Silei Xu. 2022. Thingtalk: An extensible, executable representation language for task-oriented dialogues. *CoRR*, abs/2203.12751.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Percy Liang, Michael Jordan, and Dan Klein. 2011. Learning dependency-based compositional semantics. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 590–599, Portland, Oregon, USA. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Denis Peskov, Nancy Clarke, Jason Krone, Brigi Fodor, Yi Zhang, Adel Youssef, and Mona Diab. 2019. Multi-domain goal-oriented dialogues (MultiDoGO): Strategies toward curating and annotating large scale dialogue data. 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)*, pages 4526–4536, Hong Kong, China. Association for Computational Linguistics.

Patti J. Price. 1990. Evaluation of spoken language systems: the ATIS domain. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27,1990*.

Subendhu Rongali, Luca Soldaini, Emilio Monti, and Wael Hamza. 2020. Don't parse, generate! A sequence to sequence architecture for task-oriented semantic parsing. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 2962–2968. ACM / IW3C2.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer.

Semantic Machines, Jacob Andreas, John Bufe, David Burkett, Charles Chen, Josh Clausman, Jean Crawford, Kate Crim, Jordan DeLoach, Leah Dorner, Jason Eisner, Hao Fang, Alan Guo, David Hall, Kristin Hayes, Kellie Hill, Diana Ho, Wendy Iwaszuk, Smriti Jha, Dan Klein, Jayant Krishnamurthy, Theo Lanman, Percy Liang, Christopher H. Lin, Ilya Lintsbakh, Andy McGovern, Aleksandr Nisnevich, Adam Pauls, Dmitrij Petters, Brent Read, Dan Roth, Subhro Roy, Jesse Rusak, Beth Short, Div Slomin, Ben Snyder, Stephon Striplin, Yu Su, Zachary Tellman, Sam Thomson, Andrei Vorobev, Izabela Witoszko, Jason Wolfe, Abby Wray, Yuchen Zhang, and Alexander Zotov. 2020. Task-oriented dialogue as dataflow synthesis. *Transactions of the Association for Computational Linguistics*, 8:556–571.

Minjie Wang, Da Zheng, Zihao Ye, Quan Gan, Mufei Li, Xiang Song, Jinjing Zhou, Chao Ma, Lingfan Yu, Yu Gai, Tianjun Xiao, Tong He, George Karypis, Jinyang Li, and Zheng Zhang. 2019. Deep graph library: A graph-centric, highly-performant package for graph neural networks. *arXiv preprint arXiv:1909.01315*.

John M. Zelle and Raymond J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence and Eighth Innovative Applications of Artificial Intelligence Conference, AAAI 96, IAAI 96, Portland, Oregon, USA, August 4-8, 1996, Volume 2*, pages 1050–1055. AAAI Press / The MIT Press.

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *UAI '05, Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence, Edinburgh, Scotland, July 26-29, 2005*, pages 658–666. AUAI Press.

A Data Annotation and Statistics

The data annotation process has two stages: 1) DMR graph annotation and 2) reference annotation.

A.1 DMR Annotation

In this stage, the annotators draw DMR graphs for each customer turn. They also annotate the referred turn number for the reference nodes. We developed an annotation tool based on GoJS[4] for quickly drawing DMR graphs. As shown in Figure 5a, the right part of the tool shows the utterances of dialogue up to the current turn, and the left part is the area for drawing graphs.

Before the annotation, the annotators followed a detailed guideline and took a training process. They draw the DMR graph in the diagram by adding nodes and linking them together. We enable the graph drawing process to follow the schema, which guarantees the validity of the resulting DMR.

To ensure the annotators do not hallucinate node values, the annotators must either select nodes in the bottom, or copy tokens from the current utterance (tokenized by Spacy[5]) to fill the node. Also, there are sanity checks before saving the annotations to the database. After the graph annotation, we assign variables to the nodes in each DMR.

A.2 Reference Annotation

In this stage, the annotators are given the current turn's DMR and the referred turn's DMR, and they need to annotate the referents for each reference node. The tool is modified as Figure 5b, and the left part has two diagrams. The below diagram shows the DMR graph for the current turn, and there are reference nodes there. When clicking the reference node, the referred DMR graph will appear in the above diagram, and the annotator can select the referents in it. Further, we constrain the annotator to only select referents with the same incoming edges as the reference nodes.

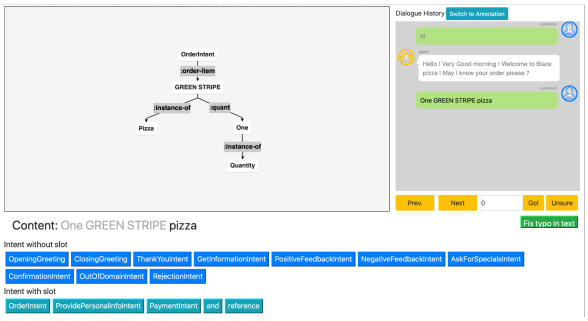
A.3 Quality Control

We make some efforts to ensure the high quality of the dataset.

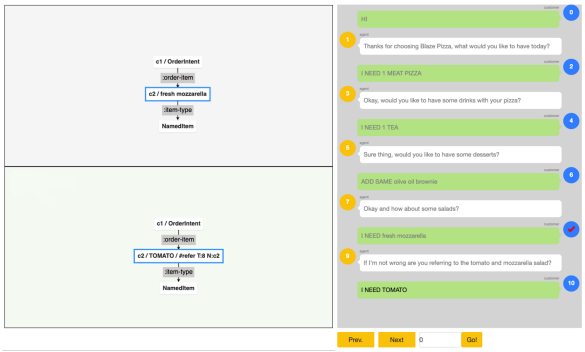
First, we ask the annotators to fix the typos in the utterances during the annotation process.

Second, in some dialogues, reference nodes appear in the first customer turn, mainly due to the customer ordering toppings but no food items. We remove these dialogues from the data.

The third is double annotation. Though the annotators are well-trained experts, we have 10% of the dialogues double annotated. We compute Fleiss' kappa (Fleiss, 1971) for measuring the Inter-Annotator Agreement (IAA). After cleaning the annotation, 5,159 utterances have valid double annotations, and the IAA is 0.748, which is a substantial agreement.



(a) DMR graph annotation tool.



(b) Reference annotation tool.

Figure 5: The interface of the annotation tool.

A.4 Data Statistics

The statistics of DMR-FastFood are listed in Table 4, we split the dataset as the original setting.

We add marks and exclude the following utterances: first, we omit utterance-annotation pairs in the train set that also occur in the dev and test set, for including them will cause information leakage; and second, a portion of the data annotated with a single intent are excluded from the dev and test set, since they are more like text classification and trivial to get right. The left data is used for NLU. In Table 4, "Utterance for NLU", "NLU DMR Depth" and "NLU DMR Nodes" are statistics based on the utterances for NLU.

We also compare DMR-FastFood with related open-source datasets in Table 7. Though DMR-FastFood is not the biggest dataset, it has more

[4] https://github.com/NorthwoodsSoftware/GoJS

[5] https://spacy.io/

	Train	Dev	Test
Dialogue	5,585	710	899
Utterance	102,843	13,111	16,889
Utterance/Dialogue	18.41	18.47	18.79
Customer Utterance	54,465	6,911	8,952
Utterance for NLU	23,633	4,256	5,581
Utterance Length	10.24	10.28	10.25
Reference	6,007	802	1,039
Negation	430	62	65
Conjunction	11,770	1,499	1,989
NLU DMR Depth	2.43	2.66	2.64
NLU DMR Nodes	3.18	3.46	3.43

Table 4: Detailed statistics of DMR-FastFood dataset.

turns per dialogue, longer utterance content, and more explicit annotations of negation and conjunction.

B Comparison of DMR with AMR and Dialogue-AMR

In this section, we make detailed comparisons of DMR with AMR and Dialogue-AMR.

B.1 DMR vs AMR

The key differences between DMR and AMR are as follows:

- AMR is a sentence-level meaning representation, and it is designed for general purpose. DMR is proposed for task-oriented dialogue (TOD), so there are special contents in DMR designed for TOD, such as Intent, Entity, and cross-turn coreference. These contents are not included in AMR.
- AMR has more than 8k pre-defined predicates and is intended to capture as much semantics as possible. While DMR is task-driven, it only captures the contents that are related to the dialogue task.
- AMR abstracts surface forms to concepts, while DMR keeps the surface forms (`<lexical_value>`) for entities, because the abstraction would depend on applications. For example, 'big mac' in the fast food domain is a type of burger, DMR abstracts it to 'Burger' entity, and keeps the surface form which could be important for the downstream module, e.g. the entity linking module, to find the exact burger item.
- DMR ontology is featured with inheritance hierarchy which serves for reusability across domains.

B.2 DMR vs Dialogue-AMR

Dialogue-AMR is an extension of AMR, so the differences described above also apply in the case of Dialogue-AMR. Further, we highlight two more differences between DMR and Dialogue-AMR as follows.

First, they serve for different applications. Dialogue-AMR is designed for human-robot interaction, and focuses on the mapping of natural language instructions to specific robotic control commands (e.g movement), whereas DMR focuses on the human interaction with software systems and their underlying business logic. The intents in DMR map to API calls, and entities map to predefined (compositional) data structures.

Second, there are multiple structural differences between them. Dialogue-AMR is an extension of AMR, which captures the illocutionary force (speech act), tense and aspect, spatial information in addition to AMR. In contrast, DMR is motivated by compositional intent-slot structures, and borrows key ideas of AMR to make it generalizable across applications in a task-oriented dialogue system. The main structural differences are:

- Dialogue-AMR captures information about speech acts, aspects, spatial information for human-robot interaction, as well as other concepts (defined in AMR) for general purpose. In contrast, DMR captures task-related intents, entities, relations defined in the ontology of a TOD application and ignores irrelevant contents.
- DMR represents coreferences across turns, which are common in TOD, while Dialogue-AMR does not.

C Details about the Inheritance Hierarchy in DMR

Inheritance Hierarchy is designed for better domain extensibility by reusing and expanding existing ontology. Extensibility and reusability of domain ontology are essential for chatbot development, given the fact that a task-oriented dialogue system generally supports large amounts of applications, and each application often serves multiple domains.

The effectiveness and efficiency of the inheritance hierarchy are rooted in the observation that despite different domains, some intents and entities can be abstracted into one intent or entity. For instance, the intent of ordering is a general intent not only for fast food ordering, but also for flight tickets, taxis, and meeting room ordering. Similarly, the 'PaymentIntent', and 'ConfirmationIntent', can also be reused across domains. On the other hand, different domains imply different requirements. Items to order in fast-food domain (food, drink) are different from those in the flight domain (tickets), which requires necessary adaptation on ontology.

To this end, the inheritance hierarchy allows inheriting from existing intents and entities, which allows preserving the arguments of their parent and modifying new arguments as needed at the same time. It is noted that the constraints about the arguments can be also derived through inheritance hierarchy. In this way, DMR arguments would help reduce much less effort in designing ontology for new domains while maintaining high extensibility. For instance, given pre-defined 'OrderIntent' and 'Item', ordering intent in the flight domain can reuse the properties of 'OrderIntent' as 'OrderFligthtTicketIntent', e.g. they can order multiple items; they can only order orderable things. And 'FlightTicket' can inherit from 'Item' like 'FoodItem', because they are all orderable and countable, etc.

D Experiment Settings

D.1 Hyperparameters

The hyperparameters for the reported results are as follows.

DMR Parsing The DMR parsing models share the following hyperparameters: Adam optimizer, batch size 10, greedy search, and 10 training epochs. Others are listed in Table 5.

Coreference Resolution The hyperparameters for the GNN-based coreference resolution model (GNNCoref) are listed in Table 6. Note that the value of threshold β is tuned on the development set during training. Specifically, during each validation step, pick values in list $[0.01, 0.02, \cdots, 0.09]$ as the thresholds and calculate the accuracies, choose the threshold with the highest accuracy as β. GNNCoref model has 2.5M parameters.

Model	Hyper-parameter	Value
BiLSTM +GloVe	Embedding size	300
	Encoder layers	2
	Decoder layers	2
	Hidden size	512
	Dropout	0.1
	Learning rate	1e-3
RoBERTa -base	Decoder layers	2
	Decoder attention heads	4
	Decoder hidden size	768
	Learning rate	3e-5
BART-base	Learning rate	1e-5

Table 5: Hyperparameters for DMR parsing models.

Hyper-parameter	Value
R-GCN Layers	3
Dropout	0.2
Epoch	20
Batch size	10
Learning rate	1e-3
Hidden size	100
Activation	LeakyReLU
β	0.89

Table 6: Hyper parameters for GNNCoref.

D.2 Training

All the models are trained on one NVIDIA Tesla T4 16G GPU. The training time of the DMR parsing models is 2.3 hours, 2.4 hours and 33 minutes for BiLSTM+GloVe, RoBERTa-base and BART-base respectively. And the GNNCoref model can be trained within 9 minutes.

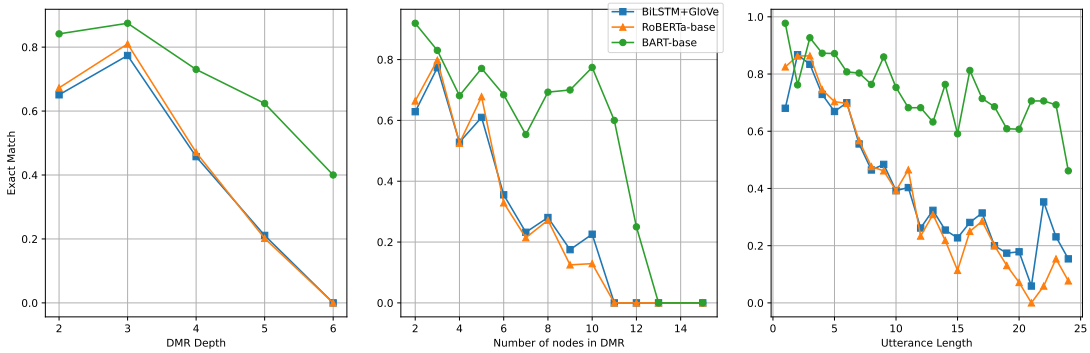
E More Analysis of the DMR Parsing model

To investigate factors that affect the performances, we analyze the DMR Parsing model from four aspects: 1) the size of the dialogue context, 2) the depth of the target DMR, 3) the number of nodes in the target DMR, and 4) the content length of the utterance.

Context Size For each DMR Parsing model, we vary the context size c from 0 to 3. The comparison of the results are listed in Table 8. The models get the best results with one or two context utterances, indicating the DMR Parsing is highly context-dependent.

	TOP	SB-TOP	TreeDST	SMCalFlow	DMR-FastFood
Dialogue	-	-	27,280	**41,517**	7,194
Utterance/Dialogue	-	-	7.1	4.1	**18.5**
Annotated Turns	44,783	64,815	**167,507**	155,923	70,328
Utterance Length	-	8.1	7.6	10.1	**10.2**
Reference	-	3,154	9,609	**45,520**	7,846
Negation	-	-	-	-	**557**
Conjunction	-	-	-	9,885	**16,087**

Table 7: Comparison of DMR-FastFood with related datasets that have tree/graph-structured representation for task-oriented dialogue systems.



(a) (b) (c)

Figure 6: DMR Parsing model performance analysis by a) DMR Depth b) Node Numbers, and c) Utterance Length.

Model	c	Dev set	Test set
BiLSTM+GloVe	0	65.41	64.63
	1	65.57	65.75
	2	**66.54**	**65.93**
	3	66.37	65.79
RoBERTa-base	0	65.93	66.29
	1	**69.24**	**68.23**
	2	67.31	67.49
	3	63.20	62.55
BART-base	0	80.89	81.99
	1	**82.56**	**83.39**
	2	82.18	83.26
	3	82.14	82.79

Table 8: DMR Parsing results with different context size c.

The following analysis are based on the test set results reported in Table 1a.

DMR Depth We compare the performance of different DMR parsing models at different DMR depths in Figure 6a. The performance drops for all models as the depth gets larger. Thus, the DMR depth is a good indicator of task complexity.

Node Number The more nodes in a DMR, the longer sequence to predict. Results in Figure 6b in line with our intuition. Moreover, we see the BART-base model performs much better than the other two on large DMR targets, indicating that a well-pretrained decoder is critical for long sequence generation.

Utterance Length We plot the DMR parsing results for different utterance lengths in Figure 6c. As expected, the models perform worse on longer utterances, and the BART-based model outperforms others substantially on these challenging test cases. This may be due to the correlation of the length of utterance and target sequence: in general, the more people say, the more information to be delivered. Thus, this result is consistent with Figure 6b.