UniDU: Towards A Unified Generative Dialogue Understanding Framework

Anonymous ACL submission

Abstract

With the development of pre-trained language models, remarkable success has been witnessed in dialogue understanding (DU) direction. However, the current DU approaches just employ an individual model for each DU task, independently, without considering the shared knowledge across different DU tasks. In this paper, we investigate a unified generative dialogue understanding framework, namely UniDU, to achieve information exchange among DU tasks. Specifically, we reformulate the DU tasks into unified generative paradigm. In addition, to consider different training data for each task, we further introduce model-agnostic training strategy to optimize unified model in a balanced manner. We conduct the experiments on ten dialogue understanding datasets, which span five fundamental tasks: dialogue summary, dialogue completion, slot filling, intent detection and dialogue state tracking. The proposed UniDU framework outperforms task-specific well-designed methods on all 5 tasks. We further conduct comprehensive analysis experiments to study the effect factors. The experimental results also show that the proposed method obtains promising performance on unseen dialogue domain. Our code will be open-sourced, once the paper is accepted.

1 Introduction

The development of the conversational system plays an important role on the spread of the intelligence devices, such as intelligence assistant and car play. In recent years, there has been a growing interest in neural dialogue system (Li et al., 2017; Bao et al., 2020; Adiwardana et al., 2020; Ham et al., 2020; Peng et al., 2020). The dialogue understanding is a core technology and hot topic in the dialogue system, which aims to accurately analyze a dialogue from different fine-grained angles.

There are five classical dialogue understanding tasks: dialogue summary (DS) (Liu et al., 2019a), dialogue completion (DC) (Su et al., 2019; Quan et al., 2020), intent detection (ID) (Kim et al., 2016; Casanueva et al., 2020), slot filling (SF) (Zhang et al., 2017; Haihong et al., 2019) and dialogue state tracking (DST) (Kim et al., 2020; Liao et al., 2021). For dialogue summary, it is normally formulated as a sequence-to-sequence generation problem. Recently, the advance methods adopt two-step generation strategy (Wu et al., 2021). They first generate the dialogue keywords as the sketch and then generate the summary based on the predicted keywords. For dialogue completion, Chen et al. (2021b) regard the co-reference and information ellipsis as the noises and directly leverage BART (Lewis et al., 2020) as the rewrite model. The intent detection is formulated as a classification problem (Liu and Lane, 2016). The advance method uses the big pre-trained model as utterance encoder learned by the classification loss (Mehri et al., 2020). The excellent slot filling methods normally formulate the task as a sequence labeling task (Zhang et al., 2017; Coope et al., 2020). For dialogue state tracking task, the advance models are hybrid of classification (Mrkšić et al., 2017) and generation (Wu et al., 2019; Tian et al., 2021). The five different tasks aim to interpret a dialogue from five different perspectives. To date, these DU tasks are still learned independently due to different task formats. However, they are intuitively related, for example dialogue completion task should have positive effect on dialogue state tracking task (Han et al., 2020). On the other hand, the dialogue data is expensive to gather and its annotations also need to consume substantial human and financial resource, which constraints the scale of annotated dialogue corpora. It is important and imperative to study how to enhance the dialogue understanding capability with the existing different dialogue corpora.

There are two main challenges to share the knowledge across the dialogue understanding tasks. The first is how to construct a unified dialogue
understanding model, which can eliminate the impacts of DU models and focus on the effects of DU tasks. In this paper, we propose a Unified Dialogue Understanding (UniDU) framework to validate the effects between different DU tasks. We unify five fundamental DU tasks as a sequence-to-sequence generation task. The second challenge is that there are huge differences between DU tasks, especially on the output space of different DU tasks. For example, there are only a few classification names in the intent detection task, while the output vocabulary of the dialogue summary task may exceed 10K. It is a nontrivial problem to efficiently learn a unified model with different dialogue corpora. In this paper, we explore eight different training strategies under UniDU framework and deeply analyze the effected factors.

The main contributions of this paper are summarized as below:

- To the best of our knowledge, we are the first to formulate the different dialogue understanding tasks as the unified generation task spanned five DU tasks. The proposed UniDU outperforms well-designed models on five well-studied dialogue understanding benchmarks.

- We validate the effects of eight different training strategies under UniDU framework. We find that the intuitive multitask mixture training method makes the unified model to bias convergence to more complex tasks. The proposed model-agnostic training method can efficiently relieve this problem.

- The experimental results show that the proposed UniDU method has excellent generalization ability, which achieves advance performance both on few-shot and zero-shot setups.

2 Dialogue Understanding Tasks

We denote dialogue context as \( C = (H_n, U_n) \), where \( H_n = (U_1, U_2, \ldots, U_{n-1}) \) represents the dialogue history containing the first \( n-1 \) turns of utterances. \( U_n \) is \( n \)-th turn utterance, which may consist of multiple sentences stated by one speaker. For the task-oriented dialogue, the domain scope is restricted by the dialogue ontology, which is designed by the dialogue expert. The ontology \( O \) is composed of dialogue domains \( D = \{d\} \) (like hotel), domain slots (like price) \( S = \{s\} \) and user intent candidates \( I = \{i\} \) (like find_hotel). There are five fundamental tasks to interpret a dialogue from different perspectives.

**Dialogue Summary** (DS) aims to extract important information of the dialogue. It is a typical generation problem, which takes the whole dialogue context \( C \) as input and generates the summary description. DS requires the model to focus on the whole dialogue flow and the important concepts.

**Dialogue Completion** (DC) purposes to relieve the co-reference and information ellipsis problems, which occur frequently in the dialogue context. It is also a typical generation task, which inputs the dialogue history \( H_n \) and the current utterance \( U_n \) and then infers the semantic-completed statement of the current utterance \( U_n \). DC requires the model to focus on connection between current utterance and dialogue history.

**Slot Filling** (SF) is to extract the slot types \( S \) of the entities mentioned by the user. It is a slot tagging problem, where the utterance is labeled in the IOB (Inside, Outside and Beginning) format. The input is only the current utterance \( U_n \).

**Intent Detection** (ID) is to recognize the intent from predefined abstracted intent expresses \( I \). It is normally formulated as a classification problem. The input is the current utterance \( U_n \) and the output is the possibility distribution of all the intent candidates \( I \).

**Dialogue State Tracking** (DST) aims to record the user’s constraints, which consists of the triple set of domain-slot-value. For example, hotel-price-cheap means the user wants a cheap hotel. The input of DST at the \( n \)-th turn is the first \( n \) turns \((U_1, \ldots, U_n)\).

3 UniDU

In this section, we first introduce the unified sequence-to-sequence format for five different dialogue understanding tasks. Then we introduce the formulation of each task in detail, especially how to reformulate the intent detection, slot filling and dialogue state tracking as the generation task.

There are three components in the input of UniDU: task identification, dialogue content, task query. The task identification represents with a special token, e.g., dialogue summary identified by “[DS]”. The dialogue content means the task-dependent input, such as dialogue history for dialogue summary. The task query can be regarded as the task-specific prompt, which includes the task definition and domain-related information. There
are two elements in the output of UniDU: task identification and query answer. The query answer is the understanding result of task query given by the dialogue content. The unified input and output can be formalized as:

**INPUT:** [TI] dialogue content [C] task query

**OUTPUT:** [TI] query answer

where “[C]” is separate character and “[TI]” is task identification (replaced by “[DST]”, “[DC]”, “[SF]”, “[ID]” and “[DST]”), which correspond to dialogue summary, dialogue completion, slot filling, intent detection and dialogue state tracking respectively). At inference time, the UniDU model has to predict the task identification first.

Dialogue summary and dialogue completion are originally generative tasks. The dialogue contents in the input are the whole dialogue context C and multi-turn utterances \( H_n \) respectively. Since these two tasks are independent with dialogue domain, there is no domain information in task query. For dialogue summary, the task query is “what is the summary of this dialogue?”. For dialogue completion, the query is “what is the semantic completion statement of \( U_n \)?”, where \( U_n \) is the t-th utterance. In the output, their understanding answers are annotated dialogue summary and rewritten utterance respectively.

The original slot filling task demands the model to extract all the mentioned slot values and their slot types in an utterance \( U_n \). In this paper, the UniDU model predicts the value slot by slot, which is an iterated generation process on the slot candidate list. Two different slot filling formats are shown below:

![Diagram](image)

To be clear, we do not list all the candidate slots here. In general, for each sample, it can be formalized as:

**INPUT:** [SF] \( U_n \) [C] what is s of d?

**OUTPUT:** [SF] slot value

where s and d are predefined slot and domain. If s has no value in \( U_n \), slot value will be “not mentioned”. If s has multiple values, they will be separated by comma in slot value. When the value is “not mentioned”, we call it negative sample. Otherwise, it is positive sample. To balance the ratio of negative and positive samples at training process, we set the ratio below 2:1. If the number of negative samples exceeds the threshold, we randomly sample twice as many negative instances as positive ones.

For dialogue state tracking tasks, the classification methods always achieve better performance than generative methods. However, under UniDU framework, we also formulate DST as slot-wise value generation task similar to slot filling task. The DST task formats are shown as below:
where the output of the original DST model is the distribution of all the candidate values of the slot. The input and output of DST task under UniDU can be formalized as:

**INPUT:** \([\text{DST}] (H_n, U_n) [C]\) what is the user’s constraint about \(s\) of \(d\)?

**OUTPUT:** \([\text{DST}]\) slot value

where \((H_n, U_n)\) is dialogue context. If slot \(s\) of domain \(d\) is not in the dialogue state, its value is “not mentioned”, which is negative sample. Note that different utterances are separated by special token “[T]” in the input. At training process, the ratio of negative and positive samples is also set below 2:1.

For intent detection task, the original methods always formulate it as the intent classification problem and output the distribution of all the candidate intents. The UniDU model directly generates the intent name of the current utterance, which can be formalized as:

**INPUT:** \([\text{ID}] U_n [C]\) what is the user’s intent on domain \(d\)?

**OUTPUT:** \([\text{ID}]\) intent name

where domain \(d\) is normally known in advance. The specific examples of original and UniDU formats are shown as below:

where we do not list all the intents. To integrate the generalization capability into the UniDU model, we also construct negative samples for intent detection task. The intent name of negative sample is “not defined”, where the input utterances \(U_n\) are sampled from out-of-domain dialogues. The ratio of negative and positive samples is set to 2:1. Until now, all the five dialogue understanding tasks are formulated as the unified sequence-to-sequence generation task. The specific examples are shown in Figure 1.

4 Multitask Training Strategies

Under UniDU framework, five dialogue understanding tasks have been formulated as a unified generative task. Due to large gap of the output space across five DU tasks, it becomes an important topic about how to efficiently train five different tasks together. In this section, we mainly introduce the multitasking training strategies.

4.1 Multitask Learning Classification

The existing multitask training strategies can be classified into three categories: average sum method, manual scheduled method and learnable weight method.

**Average Sum** method distributes all the samples with the same weight. In other words, the losses from different samples are directly averaged, formulated as \(L = \frac{1}{T} \sum_{t=1}^{T} L_t\), where \(T\) is number of the tasks and \(L_t\) is the loss of the \(t\)-th task.

**Manual Schedule** method designs a heuristic training schedule to plan the learning process of different tasks. For example, the curriculum learning (Bengio et al., 2009) is a kind of typical manual scheduled method, which first trains the easier samples and then adds the more complicated cases. The manual scheduled method can be formulated as \(L = \frac{1}{T} \sum_{t=1}^{T} \tilde{I}(t) \cdot L_t\), where \(\tilde{I}(t)\) is indicator function, whose value is 0 or 1.

**Learnable Weight** method aims to parameterize the loss weights of different tasks. The target of the parameterized weights is to balance the effects of task instances, which avoids the model to slant to one or several tasks and achieves the global optimization. There are two classical learnable weight algorithms: homoscedastic uncertainty weighting (HUW) (Kendall et al., 2018) and gradient normalization (GradNorm) (Chen et al., 2018). For the tasks, the loss function is formulated as \(L = \sum_{t=1}^{T} W_t \cdot L_t\), where \(W_t\) is learnable weights and greater than 0. In the HUW algorithm, the weights update as following loss function:

\[
L_{\text{HUW}} = \sum_{t=1}^{T} \left( L_t \cdot W_t - \log(W_t) \right), \tag{1}
\]

where \(\log(W_t)\) is to regularize weights, which is adaptive to regression tasks and classification tasks. The motivation of GradNorm method is to slow down the learning scale of task that has the larger gradient magnitude and faster convergence rate.
4.2 Model-Agnostic Training Strategy

In Equation 1, the learnable weight $W_t$ is only dependent on the corresponding task. Thus, we can regard the weight as the function of task $W_\phi(t)$, where $\phi$ are parameters shared among five tasks. Under UniDU framework, five tasks share the same encoder-decoder model, which is a constant in weight function $W_\phi(t)$. The task format depends on task attributes, such as input, output and data scale. To extract the characters of five tasks, we manually design a vector as the task feature to represent a task. Each dimension in the task feature has its physical meaning related to model-agnostic setting. In this paper, we design 14 dimensional vector $f_t$ for each task detailly introduced in Appendix B. Since the model-agnostic training strategy (MATS) formulates the weight as the task-related function and may share the function parameters among different tasks, the weights are not longer independent to each other as in original learnable weight method. The MATS improved from Equation 1 is formalized as:

$$L_{\text{MATS}} = \sum_{t=1}^{T} (\mathcal{L}_t \cdot W_\phi(f_t) - \log(W_\phi(f_t))).$$ (2)

5 Experiments

We conduct the experiments on ten dialogue understanding corpora. Each task has two corpora. We evaluate UniDU framework with eight different training strategies. Compared with well-designed models, our proposed UniDU can get better performance in five benchmarks. Then we deeply analyze different factors to affect the performance of UniDU model including DU tasks, unified format and pre-trained language models. Last but not least, we conduct few-shot experiments to validate the generalization ability of UniDU.

5.1 Corpora&Metrics

There are ten dialogue understanding corpora in total spanned five tasks: dialogue summary (DS), dialogue completion (DC), slot filling (SF), intent detection (ID) and dialogue state tracking (DST). We choose two well-studied corpora for each task: one is evaluation corpus and the other is auxiliary corpus. The dataset statistics is shown in Appendix A. Dialogue Summary: We choose SAMSUM (Gliwa et al., 2019) and DIALOGSUM (Chen et al., 2021a) datasets. The common metrics for summary task are ROUGE scores, which measure the overlap of $n$-grams in the generated summary against reference summary.

Dialogue Completion: Task (Quan et al., 2019) and CANARD (Elgohary et al., 2019) are used. The metrics are BLEU score and exact match (EM) accuracy. BLEU measures how similar the rewritten sentences are to golden ones. Exact match means the rate of the generated totally equaled to the golden.

Intent Detection: We conduct the experiments on BANKING77 (Casanueva et al., 2020) and HWU64 (Liu et al., 2019c), where 77 and 64 means the number of predefined intents. The evaluation metric is detection accuracy (ACC.).

Slot Filling: We choose to conduct the experiments on RESTAURANTS8K (Coope et al., 2020) and SNIPS (Coucke et al., 2018). We report $F_1$ scores for extracting the correct span per user utterance. Note that the correct prediction on negative samples are not calculated in $F_1$ score, which is comparable with traditional methods.

Dialogue State Tracking: WOZ2.0 (Wen et al., 2017) and MULTIWOZ2.2 (Zang et al., 2020) are used. The metric is joint goal accuracy (JGA), which measures the percentage of success in all dialogue turns, where a turn is considered as success if and only if all the slot values are correctly predicted. Note that we only use “hotel” domain data of MULTIWOZ2.2 in the training phase.

5.2 Eight Training Strategies

As introduced in Section 4, the multitask training strategies can be divided into three categories: average sum, manual schedule and learnable weight. Before introducing MTL training methods, there is an intuitive baseline trained on its own data named single training (ST). In ST, the sequence-to-sequence models are only trained on five evaluated datasets respectively. In average sum method, there are two types of training strategies: task transfer learning (TT) (Torrey and Shavlik, 2010; Ruder et al., 2019) and mixture learning (MIX) (Wei et al., 2021). The task transfer learning aims to enhance the performance using external data from auxiliary corpus that has the same task setup. This is the main reason that we select two corpora for each task. The mixture learning directly mixes up all the training samples from ten corpora together. In this two methods, the learning weight for each sample is equally distributed. In manual schedule method, we test two training routes according to curriculum learning method. From the input perspective, five tasks can be divided into three classes:
We evaluate three methods introduced in Section 4: (updated per 8 steps). The learning rates of the
we run their released code with BART-base instead of BART-large to fairly compare with our model.
∗ (according to underlined metric values) of each task continues to be fine-tuned on separate task corpus.

### Table 2: The best overall performance of MIX, HWU

<table>
<thead>
<tr>
<th>Methods</th>
<th>DS (SAMSUM)</th>
<th>DC (TASK)</th>
<th>ID (BANKING77)</th>
<th>SF (RESTAURANTS8K)</th>
<th>DST (WOZ2.0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST</td>
<td>49.74</td>
<td>47.10</td>
<td>76.4</td>
<td>89.0</td>
<td>91.4</td>
</tr>
<tr>
<td>TT</td>
<td>51.24</td>
<td>48.59</td>
<td>76.1</td>
<td>89.2</td>
<td>91.94</td>
</tr>
<tr>
<td>MIX</td>
<td>50.98</td>
<td>48.13</td>
<td>76.2</td>
<td>90.8</td>
<td>91.91</td>
</tr>
<tr>
<td>G2S</td>
<td>51.13</td>
<td>48.75</td>
<td>76.3</td>
<td>90.1</td>
<td>90.12</td>
</tr>
<tr>
<td>CL</td>
<td>51.04</td>
<td>48.36</td>
<td>77.2</td>
<td>89.8</td>
<td>92.17</td>
</tr>
<tr>
<td>GradNorm</td>
<td>51.33</td>
<td>48.69</td>
<td>77.4</td>
<td>90.4</td>
<td>92.07</td>
</tr>
<tr>
<td>HWU</td>
<td>50.31</td>
<td>47.69</td>
<td>76.2</td>
<td>90.4</td>
<td>93.14</td>
</tr>
<tr>
<td>MATS</td>
<td>50.53</td>
<td>47.97</td>
<td>76.6</td>
<td>90.6</td>
<td>93.60</td>
</tr>
<tr>
<td>Finetune</td>
<td>51.93</td>
<td><strong>49.01</strong></td>
<td>76.1</td>
<td><strong>91.0</strong></td>
<td>93.54</td>
</tr>
</tbody>
</table>

Table 1: The results on five DU tasks trained with eight learning strategies. Finetune means that the best model (according to underlined metric values) of each task continues to be fine-tuned on separate task corpus. ∗ means that we run their released code with BART-base instead of BART-large to fairly compare with our model.

<table>
<thead>
<tr>
<th>Methods</th>
<th>DS (R-L)</th>
<th>DC (R-L)</th>
<th>ID (EM)</th>
<th>SF (BLEU)</th>
<th>DST (ACC.)</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIX</td>
<td><strong>48.04</strong></td>
<td><strong>90.40</strong></td>
<td>91.9</td>
<td>96.43</td>
<td>90.1</td>
<td>83.23</td>
</tr>
<tr>
<td>HWU</td>
<td>47.63</td>
<td>89.95</td>
<td>93.0</td>
<td>97.43</td>
<td>91.8</td>
<td>83.97</td>
</tr>
<tr>
<td>MATS</td>
<td>47.57</td>
<td><strong>90.43</strong></td>
<td><strong>93.5</strong></td>
<td><strong>97.46</strong></td>
<td>91.9</td>
<td><strong>84.16</strong></td>
</tr>
</tbody>
</table>

Table 2: The best overall performance of MIX, HWU and MATS methods.

utterance-level input on intent detection and slot filling, turn-level input on dialogue completion and dialogue state tracking and dialogue-level input on dialogue summary. The inputs gradually become more complex in the order: utterance-level, turn-level and dialogue-level. Thus, the intuitive method (named CL) trains five tasks in this order. Note that the previous data are kept in the next training phase. From the task setup perspective, dialogue summary and dialogue completion belong to domain-independent tasks. The other three tasks are domain-dependent tasks. There is another training route (G2S): from general tasks to domain-specific tasks. In learnable weight method, we evaluate three methods introduced in Section 4: GradNorm, HWU and our proposed MATS.

### 5.3 Experimental Setup

In this paper, we set BART-base as the backbone of unified encoder-decoder model. The BART model is implemented with HuggingFace library (Wolf et al., 2019). We conduct all the experiments on the 2080TI GPU with 11G memory. We run every experiment for 60 epochs spent 72 hours. The batch size is 32 with gradient accumulation strategy (updated per 8 steps). The learning rates of the unified model and learnable weights are 1e-5 and 1e-4 respectively. In MATS method, the weight function consists of two linear layers with ReLU activation function, whose hidden sizes are 64.

### 5.4 Results

In Table 1, we report the best evaluation performance on five tasks with eight training strategies. The well-designed models as baselines are introduced in Section 1. The experimental results show that different training strategies greatly affect the performance on five tasks under UniDU framework. Our proposed MATS achieves the best or near best performance except on dialogue summary. On the atypical generation tasks (intent detection, slot filling and dialogue state tracking), the UniDU with MATS methods can achieve promising improvement compared to well-designed models. The simple task transfer learning method (TT) can not largely increase the performance compared with single training. The mixture operation leads consistent performance improvement on five tasks. However, compared with TT, the improvement is still limited except on dialogue completion. Compared with our proposed MATS, MIX biases convergence to more complex DU tasks (dialogue summary and dialogue completion). Two manual schedule methods (G2S and CL) do not have any distinct advantage. In learnable weight methods, GradNorm only achieves excellent performance on dialogue summary. HWU achieves performance gain on intent detection, slot filling and dialogue state tracking. We continue fine-tuning the best UniDU models (signed with underline) on the corresponding corpus. We find that only the dialogue summary and
We guess that the general dialogue summary task to validate the effects of the dialogue understanding task. For example, when pre-training a scaling dialogue model, the pre-trained tasks should be close to dialogue completion. Table 3 shows that dialogue completion have obvious performance gain, which also reflects the necessity of the UniDU framework for simpler generative tasks.

In Table 1, we report the task-specific performance of the UniDU model, whose checkpoints are selected by the task-specific metric. Table 2 shows unified performance on five tasks with MIX, HWU and MATS methods. We evaluate the single checkpoint of UniDU model, which has the highest evaluated overall score, on the five tasks. The overall score is the average value of five main metrics shown on Table 2. We can see that our proposed MATS gets the highest overall performance and also get the best performance on four DU tasks.

5.5 Analysis

In this subsection, we analyze factors to affect the performance of UniDU model including DU tasks, unified format and pre-trained language models.

5.5.1 Effects of DU Tasks

To validate the effects of the dialogue understanding tasks, we directly remove one of five DU corpora and train UniDU model with MATS method shown in Table 3. In general, the five DU tasks benefit each other, except that dialogue summary has negative effects on dialogue state tracking task.

We guess that the general dialogue summary task just summarizes a dialogue into a sentence, which ignores the domain-specific information. On the other hand, we find that the dialogue completion task has the biggest effects on the other four DU tasks. It indicates that the co-reference and information ellipsis are still main factors to impact the dialogue understanding ability. The phenomenon can facilitate the dialogue understanding community to pay more attention to dialogue completion. For example, when pre-training a scaling dialogue model, the pre-trained tasks should be close to dialogue completion task.

5.5.2 Effects of Unified Format

As introduce in Section 3, we formulate dialogue understanding tasks as QA format. There is an intuitive alternative: prefix format, where the task query is concatenated on the decoder side. At inference time, the decoder is directly fed with task query and then generates the answer. As shown in Figure 2, the QA format achieves performance boost on four of five DU tasks (except for dialogue summary) compared to prefix format.

5.5.3 Effects of PLMs

To validate the effects of the different pre-trained language models with encoder-decoder architecture. 100M and 400M are parameter sizes.

<table>
<thead>
<tr>
<th>Method</th>
<th>DS (R-L)</th>
<th>DC (BLEU)</th>
<th>ID (ACC)</th>
<th>SF</th>
<th>DST (P1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MATS</td>
<td>47.97</td>
<td>90.60</td>
<td>93.60</td>
<td>97.61</td>
<td>92.3</td>
</tr>
<tr>
<td>- DS</td>
<td>90.2 ± 0.14</td>
<td>93.20 ± 0.4</td>
<td>97.35 ± 0.37</td>
<td>92.8 ± 0.5</td>
<td></td>
</tr>
<tr>
<td>- DC</td>
<td>47.77 ± 0.20</td>
<td>93.41 ± 0.15</td>
<td>97.39 ± 0.22</td>
<td>91.8 ± 0.5</td>
<td></td>
</tr>
<tr>
<td>- ID</td>
<td>47.81 ± 0.15</td>
<td>90.5 ± 0.1</td>
<td>97.45 ± 0.16</td>
<td>92.3 ± 0.0</td>
<td></td>
</tr>
<tr>
<td>- SF</td>
<td>47.77 ± 0.20</td>
<td>90.5 ± 0.1</td>
<td>93.60 ± 0.0</td>
<td>92.0 ± 0.3</td>
<td></td>
</tr>
<tr>
<td>- DST</td>
<td>47.85 ± 0.12</td>
<td>90.6 ± 0.0</td>
<td>93.47 ± 0.1</td>
<td>97.58 ± 0.03</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 3: Ablation study on the effects of each task corpora.

<table>
<thead>
<tr>
<th>Backbone</th>
<th>DS (R-L)</th>
<th>DC (BLEU)</th>
<th>ID (ACC)</th>
<th>SF</th>
<th>DST (P1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>around 100M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trans.-B</td>
<td>34.84</td>
<td>74.2</td>
<td>86.36</td>
<td>83.01</td>
<td>72.5</td>
</tr>
<tr>
<td>BART-B</td>
<td>47.97</td>
<td>90.6</td>
<td>93.60</td>
<td>97.61</td>
<td>92.3</td>
</tr>
<tr>
<td>T5-S</td>
<td>41.63</td>
<td>85.9</td>
<td>87.04</td>
<td>96.94</td>
<td>89.9</td>
</tr>
<tr>
<td>around 400M</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trans.-L</td>
<td>34.10</td>
<td>67.4</td>
<td>86.46</td>
<td>71.65</td>
<td>71.0</td>
</tr>
<tr>
<td>BART-L</td>
<td>48.89</td>
<td>88.6</td>
<td>93.44</td>
<td>97.12</td>
<td>92.6</td>
</tr>
<tr>
<td>T5-B</td>
<td>48.89</td>
<td>90.7</td>
<td>93.90</td>
<td>98.14</td>
<td>92.6</td>
</tr>
</tbody>
</table>

Table 4: Ablation study on the effects of different pre-trained language models with encoder-decoder architecture. 100M and 400M are parameter sizes.

Figure 2: Ablation study of different unified understanding format.
5.6 Generalization Ability

To further evaluate the generalization ability of UniDU model, we first conduct few-shot learning experiments on the domain-dependent slot filling task. We test the zero-shot capability of UniDU on unseen dialogue data.

**Few-shot Learning:** We select UniDU model that gets the best evaluation overall performance on five tasks learned with MATS method. For slot filling task, we extend another dialogue corpus DSTC8 (Rastogi et al., 2020). We choose the “Bus” domain data in DSTC8, which is unseen in the training process of UniDU. Compared with vanilla BART, UniDU has obvious advantages, especially on extremely resource-limited situation. When there is only 1% training data, the vanilla BART is disable to learn as shown in Figure 3. The few-shot experiment on DST task is shown in Appendix C.

**Zero-shot Performance:** We validate UniDU model trained with MATS method on unseen “Taxi” domain dialogue data collected from MULTiWOz2.2 corpus. UniDU model can get 18.24% accuracy on ID, 39.69% F1 score on SF and 1.6% JGA on DST. The case study as shown in Appendix D indicates that the UniDU can generate reasonable results for five DU tasks on unseen domain.

6 Related Work

Our work relates to several broad research areas including prompting, dialogue modelling and multitask learning. Due to the content limitation, here we describe one subarea: multitask learning in NLP applications, that relates most closely to our work. Luong et al. (2016) apply a sequence-to-sequence model on three general NLP tasks and study different parameter-sharing strategies. Kumar et al. (2016); McCann et al. (2018) try to cast NLP tasks as QA over a context. The main topics in these work are how to design efficient model to integrate the knowledge between question and context. Liu et al. (2019b) combine four natural language understanding tasks, which utilizes BERT as the shared representation model. The model corresponding to each task still has the well-designed part to solve the intrinsic problem. It hampers the analysis of the interaction among the different tasks.

Recently, Wei et al. (2021) formulate the NLP tasks as the generation task by directly mixing scaling annotated data up. They only focus on zero-shot and few-shot ability on the NLP tasks and ignore the impacts of the different multitask training strategies, which can not achieve better performance on general NLP tasks compared to supervised learning methods on well-designed models. In task-oriented dialogue (TOD) modelling, Peng et al. (2020); Su et al. (2021) reformulate the pipeline TOD model as the sequential end-to-end generation problem. The end-to-end model needs to generate dialogue state, dialogue action and response at the same time, which is not scalable when the number of tasks increases. The sequential format needs all the annotations of the same context, which is unavailable in DU area. Most recently, PPTOD (Su et al., 2021) unifies the TOD task as multiple generation tasks including intent detection, DST and response generation. However, they focus on the response generation ability and ignore the effects of different tasks. In this paper, we deeply dive into analyzing the effects of five DU tasks.

7 Conclusion & Future Work

In this paper, we propose a unified generative dialogue understanding framework (UniDU) to share the knowledge across five dialogue understanding tasks. To alleviate the biased generation problem, we improve the existing learnable weight method, which can achieve the best overall performance. Our proposed UniDU method achieves better performance compared to well-designed models on total five DU tasks. We further deeply dive into studying the effected factors. Finally, experimental results indicate that our proposed UniDU model can also get excellent performance under few-shot and zero-shot settings. In the future, we will increase the scale of the DU corpora and integrate the unsupervised dialogue pre-training tasks. We will further examine the task-level transferability of the UniDU model.
References


Appendix

A Dialogue Understanding Corpora

<table>
<thead>
<tr>
<th>Corpora</th>
<th>#Sample</th>
<th>I(Token)</th>
<th>I(Turn)</th>
<th>O(Token)</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAMSUM</td>
<td>14732</td>
<td>104.95</td>
<td>11.16</td>
<td>20.31</td>
<td>DS</td>
</tr>
<tr>
<td>DIALOGSUM</td>
<td>12460</td>
<td>140.48</td>
<td>9.49</td>
<td>22.86</td>
<td>DS</td>
</tr>
<tr>
<td>TASK</td>
<td>2205</td>
<td>34.92</td>
<td>2.75</td>
<td>10.84</td>
<td>DC</td>
</tr>
<tr>
<td>CANARD</td>
<td>31526</td>
<td>102.67</td>
<td>9.80</td>
<td>11.55</td>
<td>DC</td>
</tr>
<tr>
<td>BANKING77</td>
<td>12081</td>
<td>21.64</td>
<td>1</td>
<td>3.14</td>
<td>ID</td>
</tr>
<tr>
<td>HWU64</td>
<td>25715</td>
<td>17.69</td>
<td>1</td>
<td>2.05</td>
<td>ID</td>
</tr>
<tr>
<td>RESTAURANTSK</td>
<td>15270</td>
<td>14.44</td>
<td>1</td>
<td>3.38</td>
<td>SF</td>
</tr>
<tr>
<td>SNIPS</td>
<td>35748</td>
<td>15.31</td>
<td>1</td>
<td>1.77</td>
<td>SF</td>
</tr>
<tr>
<td>W022.0</td>
<td>7608</td>
<td>78.96</td>
<td>4.63</td>
<td>1.30</td>
<td>DST</td>
</tr>
<tr>
<td>MULTIW022.2</td>
<td>35119</td>
<td>115.80</td>
<td>5.99</td>
<td>1.45</td>
<td>DST</td>
</tr>
</tbody>
</table>

Table 5: The ten DU corpora trained on UniDU model. I(Token) and I(Turn) mean the average length of the split tokens and the average turns of the input dialogue content. O(Token) means the average length of the split tokens of the task-specific output.

In this paper, we train our proposed unified generative model on ten dialogue understanding corpora, as shown in Table 5. For each DU task, we select two well-studied datasets. The first one is used to evaluate and the second one is an auxiliary corpus. The main reason to select two datasets for each task is to compare the multitask learning with the task transfer learning. We aim to know whether the knowledge sharing between different dialogue understanding data is only happening in the same DU task rather than all the DU tasks. The experimental results show that the annotated data from the other DU tasks are also important to enhance the performance, which indicates that it is an efficient way to transfer the knowledge among all the DU tasks. Note that the selected DU data are from different corpora, which means that the distribution of the input dialogue content is totally different. As shown in Table 5, the inputs and the outputs of the five DU tasks are greatly different from each other. The longest average input reaches to 140.48 and the shortest is only 14.44. The longest output is 22.86 from dialogue summary and the shortest is 1.30 from dialogue state tracking. These characters lead a big challenge to train all the dialogue understanding data in multitask learning way. The experimental results show that the intuitive mixture learning method makes UniDU model bias convergence to the more complex tasks like dialogue summary and dialogue completion. In this paper, we compare eight multitask training strategies. Our proposed MATS method can achieve the best overall performance on the five tasks under UniDU framework.

B Model-Agnostic Training Strategy

In traditional HWU algorithm, the learnable weight \( W_t \) is only dependent on the corresponding task. Thus, we can regard the weight function of task \( W_\phi (t) \), where \( \phi \) are parameters shared among five tasks. Generally, the task is associated with two factors: its corresponding model and task format. Under UniDU framework, five tasks share the same encoder-decoder model, which can be regarded as a constant in weight function \( W_\phi (t) \). The task format dependents on model-agnostic task setting, such as input, output and data scale. To distinguish the five tasks under UniDU framework, we manually design a vector as the task feature to represent a task. Each dimension in the task feature has its physical meaning related to model-agnostic setting. In this paper, we design 14 dimensional vector \( f_t \), as shown in Figure 4. For input and output, we add the average length of token, the average sentence number, the n-grams and the perplexity (PPL) as the attributes of the DU tasks. Especially for input, the average turn number is also an important character. The last attribute is training scale for each task. The language model calculated the PPL is LSTM-based\(^1\). Since the model-agnostic training strategy (MATS) formulates the weight as the task-related function and may share the function parameters among different tasks, the weights are not longer independent to each other as in original learnable weight method.

\(^{1}\)We run the released code at [https://github.com/pytorch/examples/tree/master/word_language_model](https://github.com/pytorch/examples/tree/master/word_language_model) for language model.
Table 6: Case study of the zero-shot performance of the best unified model trained with MATS method. The input dialogue contents are sampled from unseen “Taxi” domain.

C Few-shot Learning

We select UniDU model that gets the best evaluation overall performance on five tasks learned with MATS method. For dialogue state tracking, we utilize the “Train” domain data in MULTIWOZ2.2, which is unseen in MTL training phase. Compared with vanilla BART, UniDU has obvious advantages, especially on extremely resource-limited situation. When there is only 1% and 2% training data, the vanilla BART is disable to learn. UniDU model warmed up by MATS method can quickly adapt the model on the unseen domain.

D Case Study

We directly validate UniDU model trained with MATS method on unseen “Taxi” domain dialogue data collected from MULTIWOZ2.2 corpus. As shown in Table 6, we find that UniDU model can generate reasonable dialogue summary and completion. Note that UniDU model did not seen any task-oriented dialogue in these two tasks. For the domain-specific tasks, UniDU model can still generate accurate query answers in some cases. It indicates that our proposed generative UniDU model has excellent generalization ability, which not only can adapt to unseen dialogue and also directly generate reasonable answers on five DU tasks in zero-shot setting.

To further explore the relations among five tasks, we plot the reduced-dimension map of the task embeddings shown in Figure 6. The task embeddings are the final decoder layer representation of task identification token. We find that the embeddings of dialogue summary, dialogue completion and intent detection cluster together. These three tasks under UniDU framework are more general than slot filling and dialogue state tracking, whose task queries are slot-wise. The task formats between slot filling and dialogue state tracking are close. However, UniDU model can still have good performance to distinct between these two tasks as shown in Figure 6.