Kun Qian,[1] Wei Wei, [2] Zhou Yu [1]

A Student-Teacher Architecture for Dialog Domain Adaptation under the Meta-Learning Setting

[1] University of California, Davis
[2] Google Inc.
kunqian@ucdavis.edu, wewei@google.com, joyu@ucdavis.edu

Abstract

Numerous new dialog domains are being created every day while collecting data for these domains is extremely costly since it involves human interactions. Therefore, it is essential to develop algorithms that can adapt to different domains efficiently when building data-driven dialog models. Most recent research on domain adaption focuses on giving the model a better initialization, rather than optimizing the adaptation process. We propose an efficient domain adaptive task-oriented dialog system model, which incorporates a meta-teacher model to emphasize the different impacts between generated tokens with respect to the context. We first train our base dialog model and meta-teacher model adversarially in a meta-learning setting on rich-resource domains. The meta-teacher learns to quantify the importance of tokens under different contexts across different domains. During adaptation, the meta-teacher guides the dialog model to focus on important tokens in order to achieve better adaptation efficiency. We evaluate our model on two multi-domain datasets, MultiWOZ and Google Schema-Guided Dialogue, and achieve state-of-the-art performance.

Introduction

Intelligent personal assistants, such as Amazon's Alexa, have become popular for handling certain simple tasks, such as booking a restaurant, querying bus schedules, and ordering a taxi. Often each of these tasks requires a separate task-oriented dialog system to train. Various new domains, such as booking a haircut, a spa, etc., are added every day, so training a dialog system needs thousands of dialogs to leverage the power of deep learning. However, the data collection of these tasks is expensive, as humans must be involved. Therefore, building a task-oriented dialog model that can easily adapt to a new domain with limited data is essential for extending the usability of data-driven dialog systems.

Various methods have been proposed to tackle domain adaptation. One method is to learn a domain-agnostic embedding space, which represents the semantic meaning of dialog sentences that helps the dialog generation model generalize to new domains (**?**). Another approach is to take advantage of large pre-trained models, like BERT (**?**). Meta-learning methods can be applied for domain adaptation as well. **?** showed the model-agnostic meta-learning (MAML) (**?**) is promising for extracting domain-specific features during adaptation, which therefore helps the dialog model adapt to the new domain. However, all methods above focus on generating a decent initialized model before adaptation and ignore improving the model's adaptive efficiency in the adaptation process. To improve the adaptation process and utilize the adaptation data more efficiently, we propose a domain adaptive dialog model based on a meta-learning method and a teacher-student architecture.

During the adaptation step, when the dialog model generates a sequence of tokens as a response, we normally average the loss of all the tokens as the total loss of the entire response. However, different tokens have different importance with respect to different contexts in different domains. For example, in the sentence "what food type do you like?" from the restaurant domain, the tokens "food type" are more related to this domain compared with other tokens in this sentence. Therefore, we should pay more attention to this token during adaptation. If the model fails to generate this token, it should receive more penalty, i.e. larger loss. Therefore, the focus of our work is utilizing the meta-teacher model to learn each token's weight concerning different contexts for more effective adaptation.

In this paper, we present a **D**omain **A**daptive task-oriented dialog model with **S**tudent-**T**eacher architecture (DAST). We employ the state-of-the-art end-to-end dialog model DAMD (**?**) as the student model and adopt a transformer-based model as the teacher model. The final objective function combines the sequence of token losses generated from the student model and the corresponding weights from the meta-teacher model. We train these two models adversarially under a model-agnostic meta-learning setting, MAML (**?**). When we adapt the student model to a new domain, we update its parameters with the weighted loss, and fix the parameters of the meta-teacher model during this step. We evaluate the DAST on two multi-domain task-oriented dialog datasets, MultiWOZ (**?**) and Schema-Guided Dialog (**?**). Experimental results show that our model is effective in extracting domain-specific features and achieves a better domain adaptation performance. We will release the code base upon acceptance.

Related Work

End-to-End Task-Oriented Dialog Systems

Traditional task-oriented dialog systems consist of four modules: natural language understanding (**?**), dialog state tracker (**?**), dialog policy learning (**?**), and natural language generation (**?**). Along with the rise of neural networks, more works have explored combing these modules into a single end-to-end model (**?**). **?** proposes to construct the end-to-end dialog model based on a two-stage CopyNet (**?**), which generates belief spans and the delexicalized response (**?**) at the same time. Moreover, **?** incorporate a dialog act predictor and conducts multi-task training (**?**). The decoder takes in the dialog context and the predicted dialog action together to generate responses.

Domain Adaptation for Dialog System

Models that can utilize rich-resource domain data to adapt to new low-resource domains effectively have received much attention. **?** and **?** adopt transfer learning method to build a user adaptive dialog model. **?** introduces an end-to-end dialog model based on Hybrid Code Network (**?**) for sentiment adaptation. As for task-oriented dialog systems, **?** and **?** adopt the typical transfer learning method (**??**) and learn latent variables in a domain-agnostic embedding space to solve the problem. Pre-trained models like BERT (**?**) and GPT-2 (**?**) are also introduced to provide decent initialized model parameters for adaption (**??**). The pre-trained model improves the quality and diversity of generated sentences. Furthermore, **?** combines a pre-trained model and human correction mechanism, achieving impressive results.

Meta-Learning

Meta-learning has been shown to be promising in solving various tasks (**??**). **?** applies the MAML (**?**) algorithm to an end-to-end dialog system and demonstrates its decent performance for dialog domain adaptation. **?** modifies MAML for the dialog generation model and learns to customize the model structure for new domains. Rather than these previous models which focus on generating an initialized model for adaptation, we pay more attention to improving the process of adaptation. We adopt the MAML algorithm to train a meta-teacher model to generate weights to be multiplied with token losses. The meta-teacher learns to distinguish the tokens that the student model needs to focus on from source domains. During adaptation, the meta-teacher instructs the student on which tokens require more attention by assigning weights to tokens and re-weighting each token loss.

Teacher-Student Architecture

The teacher model has been introduced to improve the training process of the student model in various aspects. Much previous work on teacher model focuses on selecting training data (**??**). **?** and **?** distill external knowledge from teacher model to guide the training of student model, based on the idea of knowledge distillation (**?**), Moreover, **?** proposes a reinforcement learning-based teacher model, learning to teach (L2T), that can not only teach the model to select data but also select loss function. Inspired by L2T's work on selecting loss function, **?** constructs a teacher model that simulates different optimal loss functions according to different training stages. More specifically, it assigns weights to each token class for machine translation task. However, with different contexts, even the same token has varying impacts on model training, especially in different domains. Therefore we propose a meta-teacher model to generate different weights for tokens with respect to different specific domains and contexts, which instructs the student model where to pay attention to and enables the student model to adapt to a new domain more efficiently. Since our meta-teacher model focuses on the adaptation process, it is also compatible with other domain adaptation methods.

Problem Formulation

We formulate our problem as a domain adaptation problem. We have K source domains S_k $(k = 1, 2, \ldots, K)$ with the corresponding data:

$$D_{S_k} = \{(c_n^k, r_n^k, S_k), n = 1, 2, \ \ldots \ , N\}$$

and a target domain T with a limited amount of data for adaptation:

$$D_T = \{(c_n^T, r_n^T, T), n = 1, 2, \ \ldots \ , N'\}$$

where $N' \ll N$. Specifically, we set N to be around 50 times larger than N'. In the above equations, c represents the dialog context, which is the dialog model's input. And r refers to the system response, which is the dialog model's output.

There are two stages in the domain adaptation process: training and adaptation. In the training stage, we learn a general dialog model $\mathcal{M}_{source}$ using all the source domain data. This model is not an optimal model in each individual source domain. However, it can be easily adapt to a well-performed model $\mathcal{M}_{S_k}$ in specific domain through fine-tuning on this domain's data, D_{S_k}.

$$\mathcal{M}_{S_k} : C^{S_k} \times S_k \rightarrow R^{S_k}$$

where C^{S_k}, R^{S_k} are the sets of contexts and the system responses from domain S_k. Then, we adapt the model, $\mathcal{M}_{source}$ to the target domain by fine-tuning it with the target domain data, D_T. Finally, we obtain a new dialog model $\mathcal{M}_T$.

$$\mathcal{M}_T : C^T \times T \rightarrow R^T$$

Proposed Method

In this section, we first describe the detailed structure of DAST in Figure 1, including a student model and a meta-teacher model. We also introduce the forward-propagation process in the first two subsections. Then we describe student and meta-teacher's architecture and DAST model's training process.

Student Model

We show the student model in Figure 1(a). It is constructed based on DAMD dialog model (**?**), which consists of two encoders and three decoders.

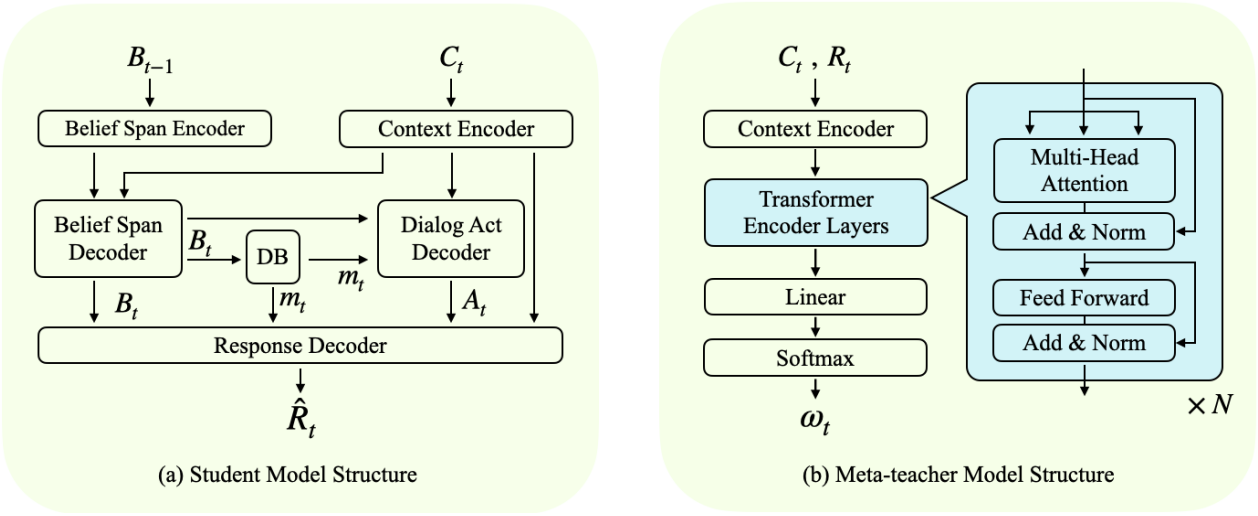

Figure 1: DAST's model architecture. (a) shows the student model's architecture. The input B_{t-1} and C_t represents the previous belief span and dialog context. The student model outputs the current belief span B_t, dialog act A_t and system response $\hat{R}_t$. m_t means the number of available choices found in the database (DB) constrained by B_t. (b) illustrates the meta-teacher model's architecture. The meta-teacher takes the context C_t and ground truth system response R_t as the input and produces the weights ω_t as the output.

Because natural sentences and belief span (**?**) (records all the task-related information provided by user, e.g "[restaurant] food Chinese price moderate") have different structures, we build separate encoders to encode them. $Encoder_B$ is used to encode belief span B_t and $Encoder_C$ is used to encode dialog context C_t, which includes the previous responses R_{t-1} and the current user utterance U_t:

$$h_B = Encoder_B(B_{t-1})$$
$$h_C = Encoder_C(C_t)$$

We input the hidden state of the previous belief span h_B and context h_C into the belief span decoder to update the belief span:

$$B_t = Decoder_B(h_B, h_C)$$

Searching the database with the current belief span, the model generates a one-hot vector m_t to suggest the number of matched entities. The dialog act decoder then takes the result obtained from database search, m_t, the belief span B_t, and the context C_t as input to predict the next system dialog act A_t:

$$A_t = Decoder_A(B_t, m_t, h_C)$$

Since the model is trained with the data from multiple domains and each domain involves a different number of dialog acts, we use a decoder rather than a classifier to obtain dialog act results.

In the end, the response decoder generates system response $\hat{R}_t$ based on all the internal variables (B_t, m_t, A_t) and the context C_t:

$$\hat{R}_t = Decoder_R(B_t, m_t, A_t, h_C)$$

We adopt cross-entropy as the basic loss function. For simplicity, we use GRU (**?**) with attention layer (**?**) and copy mechanism (**?**) as encoders and decoders.

Meta-Teacher Model

Figure 1(b) describes the structure of our meta-teacher model. We named the teacher model, meta-teacher as it has no knowledge of the target domain, but has access to samples from all the source domains. The meta-teacher learns to recognize domain-specific features in a new domain by comparing target domain data with source domain data. Then the meta-teacher model guides the student to focus on unique features in a target domain.

As illustrated in Figure 1(b), we first encode the dialog context C_t and the ground truth system response R_t with the same context encoder $Encoder_C$ as used in the student model.

$$h_1 = Encoder_C([C_t, R_t])$$

Then we input the hidden state h_1 into multiple transformer (**?**) encoder layers. In our experiment, we set the layer number as two.

$$h_2 = TransformerEncoderLayers(h_1)$$

We do not use a complete transformer here because we find that the previous decoded weight does not affect decoding the current weight, which is different from decoding tokens. The encoder layer is enough to generate well-performed weights.

At the end, we map the hidden state h_2 to weights with a linear layer and normalize the sequence of weights with a softmax layer:

$$\omega_t = softmax(W \cdot h_2)$$

Training Process

Figure 2 displays the simultaneous training process of the student and the meta-teacher model under the MAML (**?**)

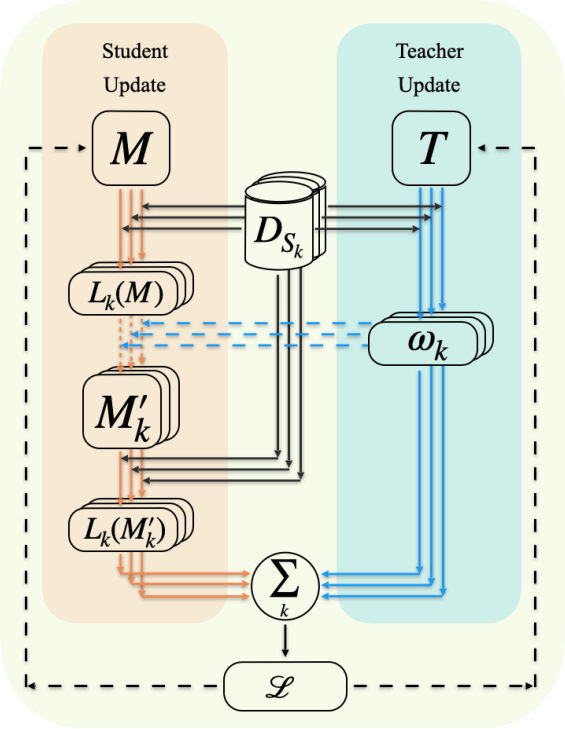

Figure 2: DAST's training process. Teacher and student model update simultaneously. The solid lines represent the forward propagation steps and the dashed lines represent back-propagation steps

Algorithm DAST

Input: Source domain data $\{D^{S_k}\}$; α; β; γ
Output: Optimal student and meta-teacher model
  Randomly initialize student model $\mathcal{M}$ and meta-teacher model $\mathcal{T}$
  while *not converged* **do**
    for $S_k \in Source\ \ Domains$ **do**
      Sample data (c^k, r^k) from D^{S_k}
      $\omega_k = \mathcal{T}(c^k, r^k)$
      $Loss_k(\mathcal{M}) = L(\mathcal{M}'(c^k), r^k)^T \cdot \omega_k$
      $\mathcal{M}' = \mathcal{M} - \alpha \nabla_{\mathcal{M}} Loss_k(\mathcal{M})$
      $Loss_k(\mathcal{M}') = L(\mathcal{M}'(c^k), r^k)^T \cdot \omega_k$
    end for
    $\mathcal{M} \leftarrow \mathcal{M} - \beta \nabla_{\mathcal{M}} \sum_k Loss_k(\mathcal{M}')$
    $\mathcal{T} \leftarrow \mathcal{T} + \gamma \nabla_{\mathcal{T}} \sum_k Loss_k(\mathcal{M}')$
  end while

Student Model $\mathcal{M}(c = \{B_{t-1}, C_t\})$:
  $h_B, h_C = Encoder_B(B_{t-1}), Encoder_C(C_t)$
  $B_t = Decoder_B(h_B, h_C)$
  $m_t = SeachDatabase(B_t)$
  $A_t = Decoder_A(h_C, B_t, m_t)$
  $R_t = Decoder_R(h_C, B_t, m_t, A_t)$
  return R_t

setting. MAML adopts a two-step updating strategy so that the learned model parameter is easy to adapt to a new domain. We apply MAML on both student and meta-teacher models.

We first initialize the student and the meta-teacher model's parameters M and T separately. For each source domain S_k, a data pair (c^k, r^k) is sampled from dataset D_{S_k}. Taking in the data pair, the meta-teacher model generates a vector of weights ω_k, corresponding to tokens in the true response r^k. Meanwhile, the student model computes the cross entropy loss $L(\mathcal{M}(c^k)_i, r_i^k)$ of each generated token, where $i \in \{1, 2, \ldots, |r^k|\}$. After weighting each token's loss, we obtain the final loss:

$$Loss(\mathcal{M}(c^k), r^k, \omega_k) = L(\mathcal{M}(c^k), r^k)^T \cdot \omega_k$$

According to MAML, we then temporarily update the student model with gradient descent:

$$\mathcal{M}' = \mathcal{M} - \alpha \nabla_{\mathcal{M}} Loss(\mathcal{M}(c^k), r^k, \omega_k)$$

and compute the loss with the updated model similarly:

$$Loss(\mathcal{M}'(c^k), r^k, \omega_k) = L(\mathcal{M}'(c^k), r^k)^T \cdot \omega_k$$

Here, for simplicity, we use the same data pair for both updates. However, we update weights when a new data pair is sampled to compute the second loss. We combine all loss values for all the sampled source domains to obtain the final loss function:

$$\mathscr{L}(\mathcal{M}, \mathcal{T}, D_S) = \sum_k Loss(\mathcal{M}'(c^k), r^k, \omega_k) \qquad (1)$$

Finally, we optimize the student model by minimizing the final loss:

$$\mathcal{M} \leftarrow \mathcal{M} - \beta \nabla_{\mathcal{M}} \mathscr{L}(\mathcal{M}, \mathcal{T}, D_S)$$

The teacher model's responsibility is to guide the student model to extract domain-specific features and quickly adapt to a target domain. This means the meta-teacher should give large weights to domain-related tokens. Since such tokens do not show up frequently in source domains, the student model does not perform well when predicting these tokens, which produces a large token loss. So meta-teacher's weight should maximize the final loss in Eq. (1). Consequently, we train the meta-teacher model and student model adversarially by optimizing:

$$\min_{\mathcal{M}} \max_{\mathcal{T}} \mathscr{L}(\mathcal{M}, \mathcal{T}, D_S)$$

Optionally, a regularization term can be added to the loss function $\mathscr{L}(\mathcal{M}, \mathcal{T}, D_S)$ to avoid situations where all weights are the same. In our experiments, we adopt the L2 regularization term.

Besides, we update the meta-teacher model during both training and validation stages. Since the student model cannot observe validation data, this does not affect the student model to learn when to stop training. In addition, training with more data enables the meta-teacher to learn the domains comprehensively and distinguish domain-specific features better.

During the adaptation stage, we still compute the weights with the meta-teacher model and multiply the weights with token losses for more efficient adaptation.

Experiments

We first introduce the datasets and the metrics used to evaluate DAST. Then, we describe the baseline models used to compare against our model. Finally, we describe the implementation details, including hyper-parameters, of our model.

Datasets

MultiWOZ (**?**) is a human-human task-oriented dialog dataset, covering seven domains. Since certain dialog covers multiple domains, we extract only single-domain dialogs and each domain averagely contains 487 dialogs. Once we select one domain as the target domain, we adopt all the data from the other six domains as training data. For the adaptation, we randomly choose nine dialogs (2% of source domain) in the target domain as adaptation data and leave the rest for testing. For each experiment, to reduce randomness in the few-shot learning setting, we repeat the adaptation process for 10 times and report the average result.

Schema-Guided Dataset (**?**) is another newly-released human-human dataset of task-oriented dialogs. It was collected in the Wizard-Of-Oz style (**?**). The dataset consists of 20 topics, each of which contains multiple tasks (e.g. both task "looking for a dentist" and task "looking for a salon" belongs to "Services" topic) and we consider each task as a domain. Then, we filter out the domains that do not require the dialog system to provide an entity to complete the tasks and the domains that contain less than 100 dialogs. There are 21 domains from 12 topics left. We randomly select seven domains from different topics as source domains for training and randomly select one domain from each of the other topics as target domains. Similar to the setting on MultiWOZ dataset, we randomly choose nine dialogs for adaptation in each domain and test our model on the other dialogs. We report the average result of each metric over 10 runs.

Evaluation Metrics

Following **?**, we adopt Inform rate, Success rate, and BLEU (**?**) score as our main metrics.

Inform rate represents the accuracy of successfully providing the correct entity (e.g. the name of a restaurant that satisfies all user's constraints in the restaurant domain).

Success rate measures how much the system answers all the requested information.

BLEU score is adopted to evaluate the quality of the generated response, compared with the oracle human response.

In addition, we also report the Slot F1 score and the Act F1 score. The Slot F1 checks if the belief span decoder correctly extracts and generates belief states, while the Act F1 measures the matching of predicted dialog act, generated by the dialog act decoder.

Baselines

To evaluate the effectiveness of our model, we compare DAST with the following two baselines:

MAML (**?**) is simple but powerful for domain adaptation. We apply MAML to train the student model and conduct adaptation with gradient descent. This baseline is similar to DAML proposed by **?**, except that the dialog model in the baseline adds the belief span encoder and the dialog act decoder.

SOLOIST (**?**) is a newly-released model that is based on pre-trained GPT-2 model. This model achieves state-of-the-art performance for domain adaptation on the MultiWOZ dataset. However, its code has not been published, we only compare its results with the same settings our model used.

Implementation Details

We adopt GloVe (**?**) as the initialized value for word embeddings, with an embedding size of 50. For the student model, each GRU from encoders and decoders contains one layer and the hidden size is set as 100. Furthermore, the GRU models of two encoders are bi-directional. As for the teacher model, it contains 2 self-attention layers with 5 heads for each. We use Adam (**?**) for optimization and set an initialized learning rate as 0.005 for both student and teacher model, as well as the meta optimizer. The learning rate decays by half if no improvement is observed on validation data for 3 successive epochs and the training process would stop early when no improvement is observed on validation data for 5 successive epochs. We adopt the batch normalization (**?**) and use a batch size of 32.

Results and Analysis

Table 1 lists the model's results on the metrics of Inform rate, Success rate, and BLEU score in all seven domains from

	Attraction			Restaurant			Train			Hotel		
Model	Inform	Success	BLEU	Inform	Success	BLEU	Inform	Success	BLEU	Inform	Success	BLEU
MAML	45.5	22.5	9.5	46.0	10.8	7.0	76.3	49.0	**5.7**	48.6	17.7	6.6
DAST	**54.7**	**32.5**	**10.2**	**51.2**	**17.5**	**8.0**	**76.9**	**50.0**	5.5	**49.1**	**25.1**	**7.6**
	Taxi			Police			Hospital			Average		
Model	Inform	Success	BLEU	Inform	Success	BLEU	Inform	Success	BLEU	Inform	Success	BLEU
MAML	-	59.8	7.7	-	41.2	7.0	-	47.6	9.8	54.1	35.5	7.6
DAST	-	**61.8**	7.7	-	**47.2**	**8.2**	-	**48.2**	**10.4**	**58.0**	**40.3**	**8.2**

Table 1: The performance in the metrics of Inform rate, Success rate, and BLEU score on all seven domains from MultiWOZ, as well as the average values over domains. We do not report the Inform rate in domain "Taxi", "Police" and "Hospital" because each of these three domains contains a default task entity. DAST outperforms the MAML baseline in terms of Inform rate and Success rate in every domain and achieves better average BLEU score.

the MultiWOZ dataset. We do not report the Inform rate of "Taxi", "Police" and "Hospital" domain because these three domains have default informable entities, which means the Inform rate is always 100%.

	Slot F1		Act F1	
Domains	MAML	DAST	MAML	DAST
Attraction	31.5	**38.4**	40.0	**41.4**
Restaurant	36.1	**42.5**	32.7	**36.9**
Train	37.5	**41.8**	29.0	**31.0**
Hotel	22.4	**26.0**	27.2	**29.1**
Taxi	-	-	44.0	**45.2**
Police	-	-	51.5	**53.9**
Hospital	-	-	**52.1**	51.1
Average	31.9	**37.2**	39.5	**41.2**

Table 2: The evaluation results on the metrics of Slot F1 and Act F1 in all domains. The DAST outperforms the MAML baseline in terms of Slot F1 in every domain and predicts better dialog acts on average.

The results in the Table 1 show that, for each domain, our model outperforms the baselines in terms of both Inform rate and Success rate. This suggests that the weights generated by the meta-teacher model are beneficial for the student model to optimize adaptation process and achieve better performance in dialog task completion. For the other metric, the BLEU score, our model does not consistently outperform the baselines. This is because the original unweighted loss treats every token in the same way in order to instruct the student model to learn the probabilistic language model. Therefore, the weights from meta-teacher slightly disturb this learning process and consequently reduce the BLEU score. However, our model still achieves better BLEU score than MAML baseline on average, indicating that although slightly affecting learning the language model, the meta-teacher helps the student model to learn new features of the new domain in general.

The performance on the other two metrics is shown in Table 2. Since each of "Taxi", "Police" and "Hospital" domain has a default task entity, which means the explicit state tracking is not required to accomplish the task, we do not report the results on Slot F1 in these three domains. On average, our model outperforms the MAML baseline in both Slot F1 and Act F1, suggesting that the meta-teacher also encourages the dialog model to generate the correct dialog states, which is necessary for database search. And only after searching the database with correct constraints, the dialog system can provide user with a correct task entity.

Model	Slot F1	Act F1	Inform	Success	BLEU
MAML	22.5	50.3	69.2	47.9	12.0
DAST	**23.2**	**59.5**	**89.6**	**61.6**	**12.1**

Table 3: The average performance of all target domains in Schema-Guided Dataset. The DAST outperforms the MAML baseline in terms of all reported metrics

We also explore the relationship between model performance and the amount of adaptation data. To keep the same setting as SOLOIST (**?**), we choose only four domains, "Attraction", "Restaurant", "Train" and "Hotel" for evaluation. We randomly sample either 80, 400, 800, or 1600 dialogs in total over four domains. We show all model's results in Table 4. We follow SOLOIST and only report Inform rate, Success rate, and BLEU score. We can find that our method consistently outperforms the baselines, both SOLOIST and MAML, in terms of Inform rate and Success rate. Since these two metrics are closely related to task completion, we believe meta-teacher guides the student model to adapt to new domains more efficiently. On the other hand, the SOLOIST model performs the best in the BLEU score. One main reason is that SOLOIST is fine-tuned based on a pre-trained language model, GPT-2. The pre-trained model makes SOLOIST generates more fluent sentences. However, our teacher-student architecture is generalizable to different student model structures, and can be built based on pre-trained student model as well. In addition, we find that the gap between SOLOIST and our methods reduces with an increasing amount of adaptation data. This is because, with

	80			400			800			1600		
Model	Inform	Success	BLEU	Inform	Success	BLEU	Inform	Success	BLEU	Inform	Success	BLEU
SOLOIST	58.4	35.3	**10.58**	69.3	52.3	**11.8**	69.9	51.9	**14.6**	74	60.1	**15.24**
MAML	62.09	38.36	9.96	72.31	52.91	10.87	74.78	57.71	11.29	76.73	60.61	11.99
DAST	**62.70**	**38.68**	9.49	**74.52**	**54.45**	11.08	**75.92**	**57.72**	11.52	**77.95**	**60.87**	11.97

Table 4: The average performance over four domains, “Attraction”, “Restaurant”, “Train” and“Hotel”, with 80/400/800/1600 dialogs for adaptation. DAST consistently achieves the best performance in the metrics of Inform rate and Success rate, with increasing the amount of adaptation data

enough data, the student model can learn the new domain well without the meta-teacher’s guidance. Therefore, the influence of the meta-teacher declines as the number of adaptation dialogs increases.

Table 3 describes the average performance in all the target domains from the Schema-Guided dataset. Our method achieves better performance compared to the MAML baseline in all metrics, suggesting that our method can generalize to different multi-domain dialog datasets.

Case Study and Visualization

Figure 3 lists four example sentences in the restaurant domain from the MultiWOZ dataset, along with their weights assigned by the meta-teacher model. To visualize the weight of each token, we color each token according to its corresponding weight. The larger the weight is, the darker the color is. Since we multiply the weights with token losses to update the student model, the absolute value of the weight can be considered as part of the learning rate. Therefore, we mainly focus on the relative values of weights within the same sentence. And the color intensity only suggests the relative value of weights in the same sentence.

The first two sentences show that our meta-teacher model focuses more on the domain-related tokens like “area” and delexicalized slots such as “[value_area]”. One possible reason is that general tokens (like “there are” in the second sentence) have already been learned by the student model in source domains while tokens like “[value_area]” appear less frequently in the source domains. Since large weight amplifies the feedback of the token loss during back-propagation, larger weights for the domain-related tokens encourage the student model to focus on those tokens and quickly learn features of the new domain, leading to more efficient adaptation. In the third case, we find that the delexicalized slots like “[value_range]” and “[value_address]” attracts more attention than domain-related token “address”. This is because the domain-related tokens are still possible to be found in other domains. For example, “address” exists in five domains in the MultiWOZ dataset. The last sentence does not contain any domain-specific tokens. Hence, the token weights are close to each other. In this case, the weights do not make much impacts on updating model parameters.

Conclusion and Future work

We propose a domain adaptation method for low-resource task-oriented dialog systems, which incorporates a student-

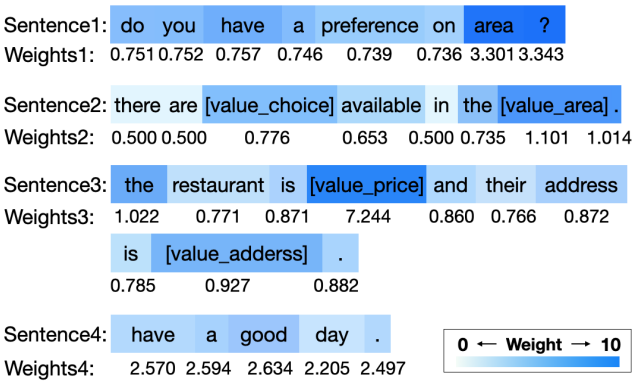

Figure 3: Visualizing weights corresponding to different tokens with different color intensities. The darker the color is, the larger the corresponding token’s weight is.

teacher architecture under the meta-learning setting. We present a transformer-based meta-teacher model, which learns to distinguish important tokens under different contexts across source domains during training. As for adaptation, the meta-teacher instructs the student dialog model to pay more attention to influential tokens by assigning weights to token losses, which improves the student model’s adaptation performance. We evaluate our method on two popular human-human multi-domain datasets. The results demonstrate that our method reaches state-of-the-art performance in most task-related metrics, compared with MAML and SOLOIST. Since the meta-teacher is built to assign weights to a sequence of generated tokens, our method can be applied to other NLP tasks, such as machine translation and summarization. Furthermore, our meta-teacher model is compatible with other domain adaptation methods, such as MAML and pre-trained models.

In the future, we aim to extend our method in several directions. First, we plan to include the Success rate and Inform rate into the loss function of the meta-teacher model in a reinforcement learning setting. We believe directly optimize task success metric may lead to better performance. Another direction is to combine the meta-teacher model and pre-trained models to explore the compatibility, as well as replacing GRU-based student model with pre-trained

Cupiditate placeat eveniet deleniti est suscipit vel sequi officiis molestiae a, ipsa consectetur