mathematics

MDPI

Article

Parameter-Efficient Fine-Tuning Method for Task-Oriented Dialogue Systems

Yunho Mo, Joon Yoo * and Sangwoo Kang *

School of Computing, Gachon University, 1342, Seongnam-daero, Sujeong-gu, Seongnam-si 13120, Republic of Korea; ahdbsgh@gmail.com
* Correspondence: joon.yoo@gachon.ac.kr (J.Y.); swkang@gachon.ac.kr (S.K.)

Abstract: The use of Transformer-based pre-trained language models has become prevalent in enhancing the performance of task-oriented dialogue systems. These models, which are pre-trained on large text data to grasp the language syntax and semantics, fine-tune the entire parameter set according to a specific task. However, as the scale of the pre-trained language model increases, several challenges arise during the fine-tuning process. For example, the training time escalates as the model scale grows, since the complete parameter set needs to be trained. Furthermore, additional storage space is required to accommodate the larger model size. To address these challenges, we propose a new new task-oriented dialogue system called PEFTTOD. Our proposal leverages a method called the Parameter-Efficient Fine-Tuning method (PEFT), which incorporates an Adapter Layer and prefix tuning into the pre-trained language model. It significantly reduces the overall parameter count used during training and efficiently transfers the dialogue knowledge. We evaluated the performance of PEFTTOD on the Multi-WOZ 2.0 dataset, a benchmark dataset commonly used in task-oriented dialogue systems. Compared to the traditional method, PEFTTOD utilizes only about 4% of the parameters for training, resulting in a 4% improvement in the combined score compared to the existing T5-based baseline. Moreover, PEFTTOD achieved an efficiency gain by reducing the training time by 20% and saving up to 95% of the required storage space.

Keywords: natural language processing; task-oriented dialogue system; PEFT; fine-tuning; training efficiency

MSC: 68T50

Citation: Mo, Y.; Yoo, J.; Kang, S. Parameter-Efficient Fine-Tuning Method for Task-Oriented Dialogue Systems. *Mathematics* **2023**, *11*, 3048. https://doi.org/10.3390/math11143048

Academic Editor: Florentina Hristea

Received: 28 May 2023
Revised: 5 July 2023
Accepted: 7 July 2023
Published: 10 July 2023

1. Introduction

In task-oriented dialogue systems, the primary objective is to enable user–system communication to accomplish specific tasks, such as a restaurant search, hotel reservation, or schedule management. These systems generally focus on understanding the user input, tracking dialogue states, and generating appropriate responses.

The conventional task-oriented dialogue system follows a pipelined structure, consisting of several interconnected modules: the Natural Language Understanding (NLU) module, the Dialogue State Tracking (DST) module, the Dialogue Policy Learning (POL) module, and the Natural-Language-Generation (NLG) module, as shown in Figure 1 [1]. First, the NLU module is responsible for extracting semantic information from user inputs. The DST module utilizes the previous conversation history to update the belief state at the time of the current utterance. The belief state is a structured expression method that represents the user's conversational goals and information gathered thus far. The system then searches the database for relevant information based on the belief state. The POL module determines the system action based on the knowledge retrieved from the database and the current belief state. Finally, the NLG module generates a system response based on the decision made by the POL module. In general, this pipelined architecture facilitates the

flow of information in a task-oriented dialogue system, enabling efficient understanding of the user input, tracking dialogue states, determining system actions, and generating appropriate responses.

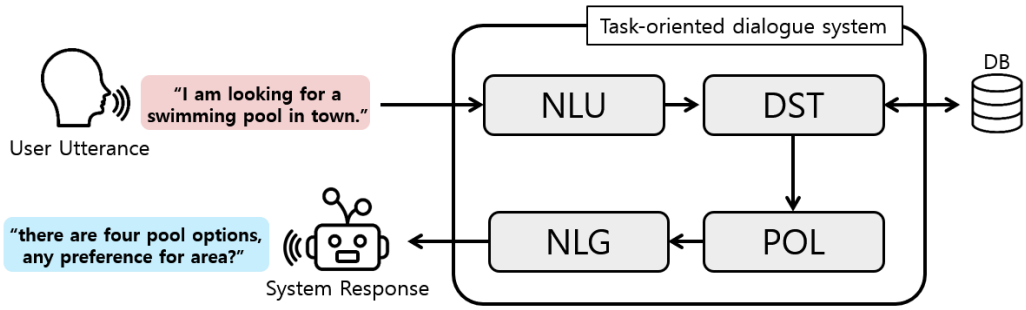

Figure 1. Conventional structure of the task-oriented dialogue system.

The conventional pipelined structure of task-oriented dialogue systems, however, suffers from error propagation between independent modules and limited adaptability to new domains. To address these problems, recent studies have proposed structures that integrate independent modules using pre-trained language models [2–4].

Recent language models have widely adopted the Transformer-based model [5] and have shown a dramatic increase in performance, in tasks such as object name recognition, natural language inference, and machine translation. Theses language models generally employ transfer learning [6], where knowledge is first learned from a source domain and then transferred to the target domain. The Transformer-based language model consists of a pre-training step, which first learns the syntax and semantics of the language from large text data, and a fine-tuning step, which adjusts the model's parameters for downstream tasks. Therefore, pre-trained language models that capture the syntax and semantics of a language render better performance compared to learning data from scratch. During fine-tuning, the entire parameter set of the pre-trained language model is adjusted to fit the downstream task.

A recent study [7] showed that larger pre-trained language models, with more parameters, tend to achieve better performance in downstream tasks. This trend also applies to task-oriented dialogue systems, where the parameter count of the pre-trained language models has reached tens of billions. However, fine-tuning such large-scale models poses challenges. First, the training time increases as the number of parameters grows since the entire parameter set is updated during fine-tuning. Second, fine-tuning a large-scale pre-trained language model requires significant storage space due to the increased model size.

In this paper, we propose PEFTTOD (the name PEFTTOD comes from integrating the PEFT method into TOD systems), a novel structure for solving task-oriented dialogue (TOD) systems using a large-scale pre-trained language model. PEFTTOD efficiently utilizes the parameters by employing the Adapter Layer [8] and prefix tuning [9] techniques from the Parameter-Efficient Fine-Tuning (PEFT) method [10]. The PEFT method incorporates a trainable layer into the pre-trained language model while freezing the parameters of the existing model and learning only the newly added layer. The PEFT method offers several advantages. First, although the PEFT method is trained with a much smaller parameter count than the pre-trained language model, it achieves performance comparable to fine-tuning. Second, by freezing the weight of the pre-trained language model and training only the added trainable layers, the original state of the pre-trained model is preserved. Third, whereas fine-tuning requires saving the entire model, the PEFT method only necessitates saving the parameters of the trainable layer, resulting in significantly reduced storage space. Lastly, since the parameters of the pre-trained language model remain frozen, the weight update process of the frozen layers is skipped, leading to faster training speeds.

PEFTTOD utilizes PPTOD [2] as its pre-trained language model, which integrates an extensive knowledge conversational domain based on T5 [11] and combines it with the

PEFT method [10]. The performance of PEFTTOD was evaluated using the Multi-WOZ 2.0 benchmark dataset [12]. Compared to the conventional fine-tuning method, PEFTTOD uses only 4% of the parameters of the existing model during the training process. This leads to improvement in training time by 20% and storage space savings by up to 95%. Moreover, PEFTTOD demonstrated 4% improvement in the combined score compared to the baseline, despite using only 4% from the parameters of the previous model.

The main contribution of this paper is three-fold. Firstly, existing pre-trained language models typically employ billions of parameters, which leads to longer training times, as well as significant storage space due to the larger model size. In our proposed approach, PEFTTOD, we adopted the Adapter Layer and PEFT-based prefix tuning to decrease the number of parameters. Secondly, PEFTTOD was trained with a substantially smaller parameter count and, thus, requires less storage space. Consequently, as the parameters of the pre-trained language model remain frozen, the training speed is accelerated. Thirdly, we conducted extensive experiments using the Multi-WOZ 2.0 benchmark dataset to prove our advantages.

The remainder of the paper is organized as follows. In Section 2, we provide an overview of the related work. Section 3 presents the details and design of our proposed approach, and Section 4 presents the evaluation results. Finally, we conclude our paper in Section 5.

2. Related Work

This section describes technologies related to PEFTTOD: the pre-trained language model, the task-oriented dialogue system, and various PEFT methods.

2.1. Pre-Trained Language Models

In the field of natural language processing, since the advent of Transformer technology, the grammar and vocabulary of a language are first learned from a large corpus in order to apply transfer learning. This method of fine-tuning the pre-trained language model shows good performance in all tasks of natural language processing. Transformer's encoder-based models (BERT [13], RoBERTa [14], DeBERTa [15]) perform fine-tuning for natural-language-understanding tasks and show high performance. The parameters of the model are increased in the order of BERT (110 M)-RoBERTa (125 M)-DeBERTa (1.5 B). Transformer's decoder-based models (GPT-1 [16], GPT-2 [17], GPT-3 [7], LaMDA [18], OPT [19]) are fine-tuned for natural language generation and show high performance. The number of parameters for GPT-1 (117 M)-GPT-2 (1.5 B)-GPT-3 (175 B) are increasing, and the recently published LaMDA (137 B) and OPT (175 B) also have a very large number of parameters. Transformer's encoder–decoder-based models (BART [20], T5 [11]) are used after fine-tuning for the translation and summary tasks, which require natural language understanding and natural language generation. BART (400 M) and T5 (11 B) also have the problem of increasing parameters.

2.2. Task-Oriented Dialogue System

In the task-oriented dialogue system, the structure typically consists of three main components: Dialogue State Tracking (DST), Dialogue Policy Learning (POL), and Natural Language Generation (NLG). These components work together to understand user utterances, determine the dialogue objectives, and generate appropriate responses [1].

Before the emergence of pre-trained language models, several approaches were used in task-oriented dialogue systems. Some of these approaches include the following. First, the LSTM+CNN structure [21] combines Long Short-Term Memory (LSTM) networks with Convolutional Neural Networks (CNNs) for dialogue understanding. Second, the Sequence-to-Sequence (Seq2Seq) model [22–24] is used for generating responses in dialogue systems. Sequicity [22] is an example of Seq2Seq-based models applied to dialogue systems; DAMD [23] extended a single-domain dialogue system to multiple domains; LABES-S2S [24] attempted semi-supervised learning. Third, several studies have explored

the application of reinforcement learning in dialogue systems, including models such as JOUST [25], LAVA [26], DORA [27], SUMBT+LaRL [28], and CASPI [29]. With the advent of pre-trained language models, models such as DoTS [30] used Bidirectional Encoder Representations from Transformers (BERT) and Gated Recurrent Unit (GRU) for dialogue state tracking. Regarding the decoder structure, some models introduced specific methods. SimpleTOD [31] used special tokens and delexicalization [21] for domain adaptation, while SOLOIST [32] employed contrastive learning [33] and negative data samples. UBAR [34] uses the entire conversation history to generate an answer, as opposed to the traditional single-answer methods. The combination of encoder and decoder structure has also been explored. Models such as MinTL [3], PPTOD [2], and MTTOD [4] use pre-trained models such as BART and T5. In MTTOD, span prediction was applied as an auxiliary loss. GALAXY [35] used UNILM and unified reconciliation for multiple datasets as ISO norms.

2.3. Parameter-Efficient Fine-Tuning Method

Transformer-based pre-trained language models have become the foundation for natural language processing by learning the syntax and semantics in advance. It has become a common approach to fine-tune the entire model for transfer learning. However, recent studies have proposed more-efficient methods for utilizing pre-trained models: learning without adding parameters or learning by adding more parameters.

2.3.1. PEFT Method without Adding Parameters

One approach is to fine-tune only the top layer or prediction head of the pre-trained language model while keeping the remaining layers frozen. This partial fine-tuning method, as described by Lee et al. (2019) [36], achieves lower performance compared to fine-tuning all parameters. Another method called BitFit [37] trains only the bias term of the pre-trained language model, which has shown on-par performance with fine-tuning on certain resource-constrained tasks.

2.3.2. PEFT Method with Added Parameters

The PEFT method involves adding learnable parameters inside the pre-trained language model. During the learning process, the parameters of the pre-trained language model are frozen, and only the added parameters are trained. This method achieves performance similar to conventional fine-tuning.

Adapters have been introduced as an efficient way to incorporate additional parameters into pre-trained language models. Houlsby Adapter [8] was the pioneering work to apply the Adapter concept, featuring a bottleneck structure that can be added to the pre-trained model. It adds two Adapter Layers within one layer of the Transformer, one after the Attention Layer and another after the Feed-Forward layer.

AdapterFusion [38] proposed a structure called Pfeiffer Adapters and using the Adapters in parallel before merging. It adds an Adapter in one layer of the Transformer after the last Feed-Forward Network after the Add and Norm. Zhu et al. [39] proposed a parallel Adapter structure that uses the value before passing the input to the Attention Layer as the input in the existing Adapter structure.

Additionally, studies have explored Adapters for specific purposes. In the work in [40], a domain Adapter for domain adaptation in machine translation was proposed. MAD-X [41] proposed a language Adapter, a task Adapter, and an invertible Adapter, which are effective for learning the multilingual language models. LoRA [42] proposed a method to decompose the attention weight update process during fine-tuning in the pre-trained language model and applying it to the Adapter. He et al. [43] experimented with multiple adaptors on various downstream tasks to propose an effective Adapter structure. UniPELT [44] proposed an integration framework that integrates the PELT method into submodules and enables utilizing the best method for the current data or task setup through a gating mechanism.

Prefix tuning [9], inspired by the prompt methodology, aims to improve the performance of pre-trained language models. It involves modifying the input data format according to the learning method of the pre-trained language model. Prefix tuning adds a prefix vector, which can be trained within the pre-trained language model, allowing the treatment of prompts as if they were combined with a virtual token created by the learnable prefix vector, without directly modifying the input data.

3. Design

This paper proposes PEFTTOD, a Transformer-based task-oriented dialogue system that leverages a parameter-efficient language-model-tuning method. This system combines a Transformer-based language model with an efficient learning structure for conversational knowledge. PEFTTOD's pre-trained language model uses PPTOD [2], which is trained on a large amount of conversational domain knowledge, based on T5 [11]. In PPTOD, a prompt corresponding to the downstream task of the task-oriented dialogue system is combined with the input data. For example, prompts such as "translate dialogue to belief state:", "translate dialogue to dialogue action:", and "translate dialogue to system response:" are used. However, a prompt attached to the data may not be optimized for the model's performance [9]. To address this issue, the proposed PEFTTOD system incorporates a structure that enables the model to learn the prompt directly through prefix tuning.

3.1. End-to-End Dialogue Modeling

PEFTTOD incorporates a structured framework that effectively learns conversational knowledge by leveraging PPTOD [2], a T5-based language model trained on a substantial amount of information specific to the conversational domain.

The system architecture of PEFTTOD is based on a sequence-to-sequence architecture model, as shown in Figure 2. At each dialog turn, the encoder takes input consisting of the dialogue history and the user's utterance. On the basis of the encoded conversation information, the decoder generates a belief state, which represents the system's understanding of the user's intentions and requirements.

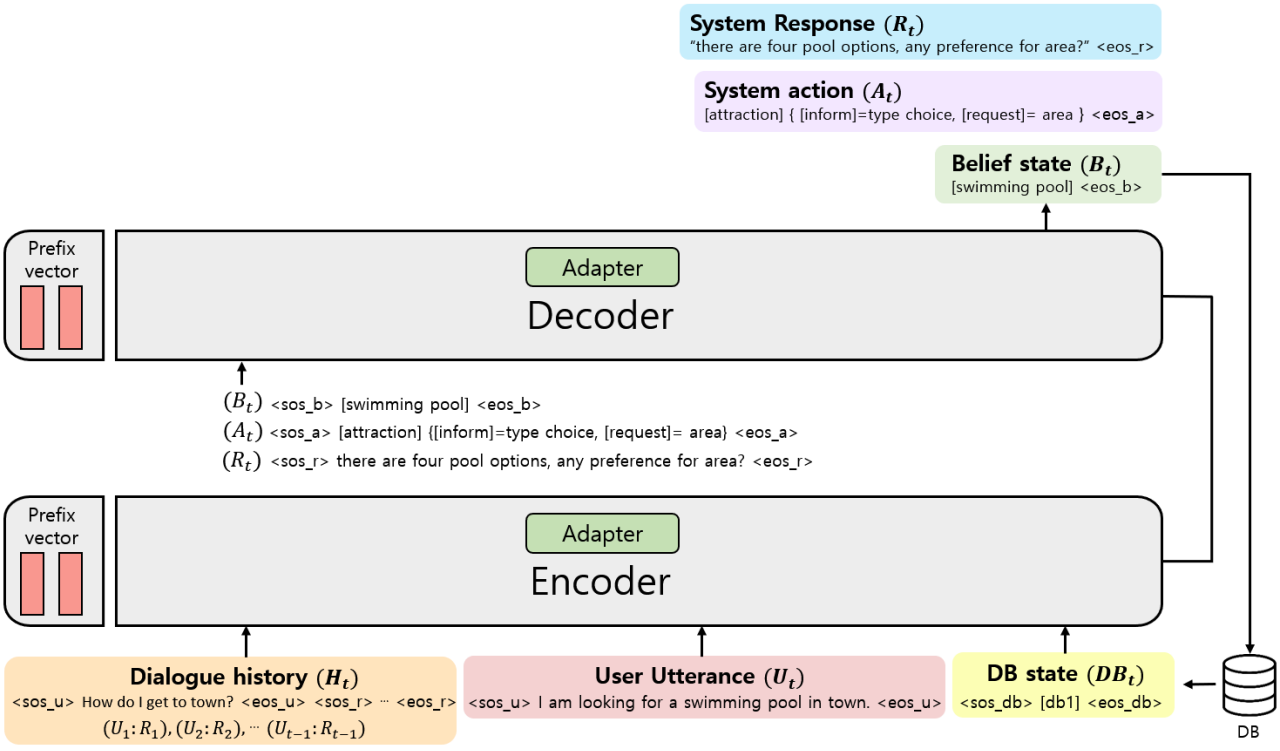

Figure 2. Structure of PEFTTOD.

The generated belief state is used for database search, enabling the system to obtain the corresponding DB state from the database. Additionally, based on the encoded dialog information and DB state, the decoder generates a system action and a system response. The system action determines the decision or action the dialogue system should take, while the system response represents the system's generated reply to the user.

PEFTTOD was trained on the Multi-WOZ 2.0 dataset, specifically on the task of the end-to-end dialogue modeling [45]. The proposed system was trained using the maximum likelihood method, a common approach in machine learning, which aims to optimize the model's parameters by maximizing the likelihood of generating the correct outputs given the inputs.

Say that $D = (x, y)$ (here, D is the data and $x = \{H_t, U_t\}, \{H_t, U_t, DB_t\}, y = B_t, A_t, R_t$), then the loss ($L$) becomes:

$$L = -logP(y_t|x_t) \tag{1}$$

3.2. The Proposed Model

Figure 3 shows the encoder and decoder parts of Figure 2 in detail. PEFTTOD incorporates a PEFT method within a pre-trained language model. The left part of Figure 3 shows the structure of the existing system, while the right part represents the structure of PEFTTOD. PEFTTOD effectively compresses the hidden state information as it passes through the Attention Layer and Feed-Forward Layer and then transfers it to the subsequent layers. It then adds an Adapter, i.e., a trainable bottleneck layer, to each layer. In addition, within the attention mechanism, prefix tuning is performed to learn P_K and P_V. This allows the model to directly learn the prompt information within the language model itself, making the structure task-independent. Unlike the existing system, which combines prompts with input data on a task-specific basis, PEFTTOD learns and utilizes prompt information within the language model itself. In the following subsection, we describe the parallel Adapter and prefix tuning in more detail.

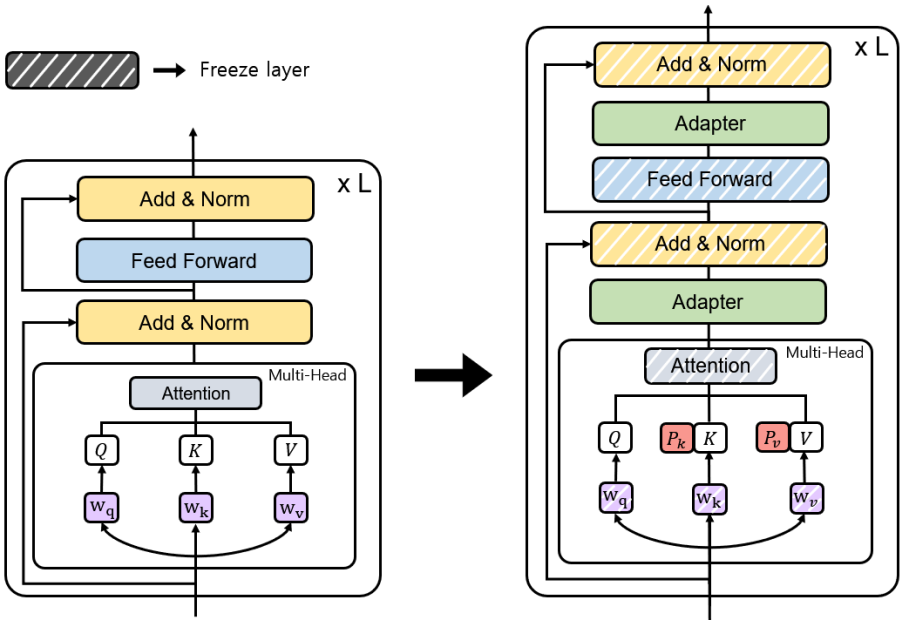

Figure 3. Combining the Transformer structure and PEFT method.

Figure 4 shows the structure with the parallel Adapter [39] applied in PEFTTOD. PEFTTOD is a Transformer-based architecture that incorporates two Adapter Layers within a single layer and input value x, replacing the input of the hidden state. The value x represents the value before passing through the Attention Layer.

$$h \leftarrow W_{up} \cdot f(W_{down} \cdot x) + h \tag{2}$$

In Equation (2), W_{down} down-projects the incoming hidden state h, f is a non-linear activation function, W_{up} up-projects the hidden state, and r is the residual network. Here, $W_{down} \in \mathbb{R}^{D_{hidden} \times D_{bottle}}$ and $W_{up} \in \mathbb{R}^{D_{bottle} \times D_{hidden}}$, where D_{hidden} is the hidden size and D_{bottle} is the bottleneck size. During training, the pre-trained language model combined with these Adapters freezes the parameters corresponding to the pre-trained language model, and only the Adapter is fine-tuned. Thus, the conversational knowledge can be efficiently forwarded within the pre-trained language model.

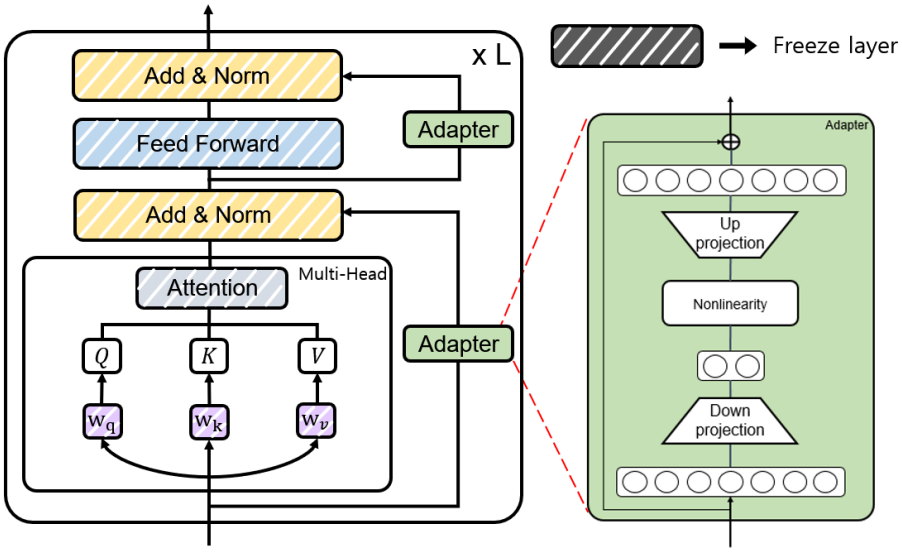

Figure 4. Structure of the parallel Adapter.

Figure 5 illustrates the structure of prefix tuning [9] in PEFTTOD. We combined the key (K) and value (V) of the Transformer's multi-head attention block with the prefix vectors P'_k and P'_v each of length l. P'_k and P'_v are defined as $P'_k, P'_v \in R^{l \times hidden}$. However, if we use the combined prefix vector as a direct parameter, then the performance will degrade. To solve this problem, we stabilized P by reparameterizing P' through a neural network identical to the structure of the Adapter, as shown in Equation (3) [46].

$$P = W_{up} \cdot f(W_{down} \cdot P') \tag{3}$$

where $W_{down} \in \mathbb{R}^{D_{hidden} \times D_{bottle}}$, $W_{up} \in \mathbb{R}^{D_{bottle} \times D_{hidden}}$, f denotes the non-linear activation function, D_{hidden} is the hidden size, and D_{bottle} is the bottleneck size. This neural network only maintains the matrix corresponding to the reparameterized P and can be removed after training. In the training step, the query of the Transformer's attention block is defined as $Q \in R^{M \times hidden}$, the key is $K \in R^{M \times hidden}$, and the value is $V \in R^{M \times hidden}$. Here, M is the max sequence length. During training, as shown in Equation (4), we concatenate the prefix vectors P'_k and P'_v in K and V, respectively, where $P_k + K \in R^{(l+M) \times hidden}$ and $P_v + VR^{(l+M) \times hidden}$.

$$head_i = Attention(QW_q^i, concat(P_k^i, KW_k^i), concat(P_v^i, VW_v^i)) \tag{4}$$

In PEFTTOD, the prefix tuning is trained by inserting a prefix vector into the attention mechanism of the pre-trained language model. This differs from the existing model where the prompt is combined with the input data in an arbitrary manner [2]. In contrast, the prefix vectors P_k and P_v inserted inside the model allow for the learning of the prompt that is optimized specifically for the entire conversation system.

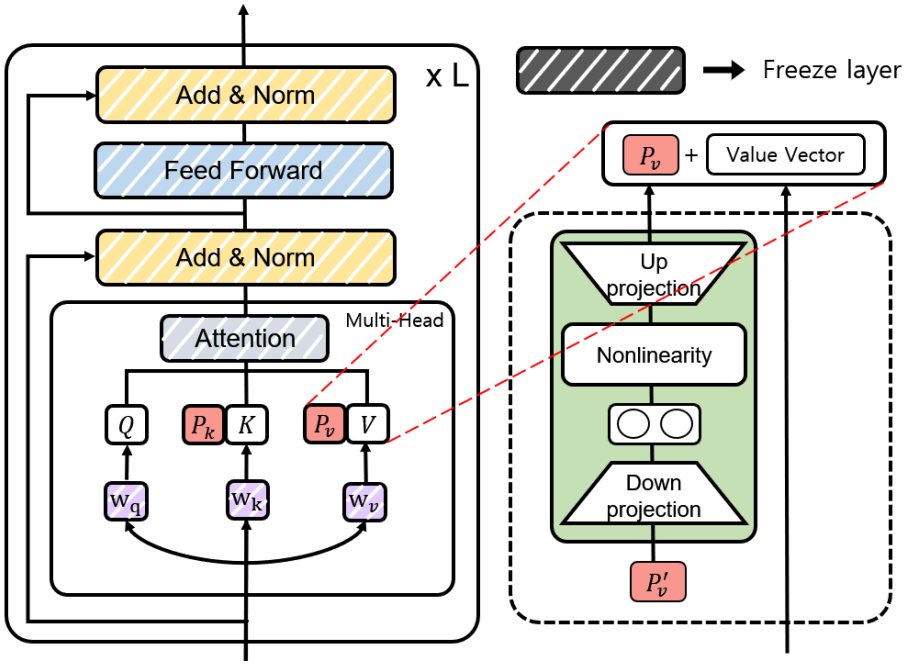

Figure 5. Structure of prefix tuning. (The parameters of the layer inside the dotted-line box can be removed after reparametrization).

3.3. Domain Adaptation

The proposed system uses two methods for domain adaptation. The first way is to use a special token. Special tokens are specifically designed to identify different components of the inputs corresponding to different subtasks. As demonstrated by SimpleTOD [31], the absence of special tokens can lead to the generation of much longer belief states, system actions, and system responses. Therefore, it is important to clearly distinguish between the user and the text of the system within the dialogue history of the system. To identify the user's utterance, the system's utterance, the dialogue state (belief state), the DB state, and the system action, the proposed system uses <sos_u>, <eos_u>, <sos_r>, <eos_r>, <sos_b>, <eos_b>, <sos_db>, <eos_db>, and <sos_a>, <eos_a>, respectively.

The second method employs delexicalization. The delexicalization method is a preprocessing method that groups specific slot values into categories [21]. For example, if there is a slot called "Food" with various food options, the corresponding slots related to food are pre-processed and categorized as "Food". During the generation process, the actual values are retrieved from the database and filled accordingly.

4. Evaluation

We evaluated the performance of PEFTTOD in the context of task-oriented dialogue systems for end-to-end dialogue modeling [45]. The evaluation was conducted using the benchmark dataset Multi-WOZ 2.0 [12]. The baseline model, PPTOD, which is described as a language model based on T5 [11], was trained to acquire a significant amount of knowledge about the conversation domain. We conducted a comparative experiment according to the structure of the system combined with the PEFT method.

PPTOD uses a smaller model, and it was trained directly to replicate the same experimental setup as the proposed system. In Tables 1–3, the baseline performance is indicated as "Fine-tuning", while the performance of direct training is indicated as "Fine-tuning (our run)". Additionally, "params" represents the trainable parameters of the language model with the PEFT method applied.

4.1. Dataset and Evaluation Metrics

The experiments used the Multi-Woz 2.0 dataset, which is widely used as a benchmark dataset for the task-oriented dialogue system. The dataset is a multi-domain dataset,

which consists of 8438 conversations for seven domains: tourism, attractions, hospitals, police stations, hotels, restaurants, taxis, and trains. The experiment focused on five of these domains, excluding hospitals and police stations, due to the absence of dev and test data for these domains. Note that a single conversation can involve conversations from multiple domains and databases associated with the belief state are organized based on their respective domains. Therefore, the database state uses the dialogue state (belief state) generated through dialogue state tracking as a query to search from a predefined database and obtain the search result. The proposed system first predicts the dialogue state (belief state) through DST and searches the DB at the time of inference. Next, based on the DB state and dialogue history obtained as a search result, the system action and system response results are generated sequentially. To evaluate the performance of the model, an end-to-end dialogue modeling evaluation was conducted, which measured the quality of the generated belief state, system action, and system response when a user utterance is input. The model's evaluation metrics followed the automatic evaluation metrics [12]. The automatic evaluation metrics are widely used in dialogue system research utilizing the MultiWOZ 2.0 dataset. Inform measures whether the system has provided the correct entity, and success measures whether it has responded to all the requested information. Additionally, BLEU [47] was used to assess the quality of the generated response. The combined score was the performance evaluation index proposed in [48] and is shown as Equation (5).

$$Combined\ Score = (Inform + Success) \times 0.5 + BLEU \quad (5)$$

4.2. Adapter Types

This experiment evaluated the performance of the Adapters with different structures, namely the Houlsby Adapter and Parallel Adapter. These Adapters were compared with PPTOD, a model that was pre-trained on the conversation knowledge. The results are presented in Table 1, indicating that the Parallel Adapter structure demonstrated the best performance among the evaluated options. Therefore, the paper leveraged this parallel Adapter structure for further experiments and analysis. Furthermore, we also explored the usage of prefix tuning on the dialogue system. When only prefix tuning was used, it resulted in a lack of communication knowledge within the language model. To address this limitation, the experiments in Section 4.5 combined the use of prefix tuning with the Adapter structure.

Table 1. Experimental results for Adapter types. In this and the following tables, the bold numbers indicate the highest performance for each criteria.

Method	Inform	Success	BLEU	Comb.	Params
Fine-tuning	87.8	75.3	19.89	101.44	100%
Fine-tuning (our run)	83.7	75.4	19.07	98.62	100%
Prefix tuning	58.5	42.7	12.28	62.88	0.30%
Houlsby Adapter	82.0	71.8	17.50	94.40	1.32%
Parallel Adapter	**83.4**	**74.0**	**19.14**	**97.84**	1.32%

4.3. Performance Comparison for the Number of Adapters

Generally, in a pre-trained language model, as more parameters are trained, the performance tends to improve [7]. Therefore, this experiment investigated the impact of increasing the number of Adapter Layers. Table 2 presents the results of this comparison for both the Houlsby Adapter and the Parallel Adapter. The numbers in the parentheses denote the number of Adapters connected in series. It was observed that, as the number of Adapter Layers increased, the performance of both Adapter structures improved. This suggested that incorporating more Adapter Layers enhanced the overall performance of the model. Notably, even when the parameters corresponding to the pre-trained language model were not trained, but the parameters related to the PEFT method increased, there

was still a performance improvement. This indicated that the Adapter Layer played a crucial role. However, note that, when the Adapter number reached seven, we observed a performance degradation; thus, it is important to find the optimal number of Adapters to achieve the best performance.

Table 2. Experimental results for the number of Adapters.

Method	Inform	Success	BLEU	Comb.	Params
Fine-tuning	87.8	75.3	19.89	101.44	100%
Fine-tuning (our run)	83.7	75.4	19.07	98.62	100%
Houlsby Adapter	82.0	71.8	17.50	94.40	1.32%
Houlsby Adapter (3)	87.8	77.3	**17.73**	100.28	3.96%
Houlsby Adapter (5)	**89.4**	76.9	17.58	**100.73**	6.60%
Houlsby Adapter (7)	85.6	**77.7**	17.62	99.27	9.24%
Parallel Adapter	83.4	74.0	19.14	97.84	1.32%
Parallel Adapter (3)	**87.4**	76.1	17.58	99.33	3.96%
Parallel Adapter (5)	86.7	**76.9**	19.15	**100.95**	6.60%
Parallel Adapter (7)	87.0	75.4	**19.61**	100.81	9.24%

4.4. Prefix-Tuning Performance Comparison

In this experiment, we used the T5-based PPTOD-Small, which was trained to acquire conversation knowledge, in order to evaluate the performance of prefix tuning. PPTOD [2] is a trained model that incorporates a prompt with the input data. Therefore, for the models that use prefix tuning, we excluded the combination of prompts with the input data during training. Table 2 shows that the model with a combination of the Houlsby Adapters and Parallel Adapters in series for three and five times, respectively, achieved the highest performance. Hence, we incorporated prefix tuning into these Adapters in the experiments. In Table 3, we observe that the model combining prefix tuning after connecting the Parallel Adapter three times in series yielded the best performance. Consequently, we named this proposed model PEFTTOD. The inclusion of prefix tuning in the model's structure enhanced the performance by allowing the model to learn information related to specialized prompts within the conversation system, without explicitly combining prompts in the input data.

Table 3. Experimental results for prefix tuning.

Method	Inform	Success	BLEU	Comb.	Params
Fine-tuning	87.8	75.3	19.89	101.44	100%
Fine-tuning (our run)	83.7	75.4	19.07	98.62	100%
Houlsby Adapter (3)	87.8	77.3	17.73	100.28	3.96%
Houlsby Adapter (3) + prefix tuning	84.5	74.1	**18.38**	97.68	4.27%
Houlsby Adapter (5)	**89.4**	76.9	17.58	100.73	6.60%
Houlsby Adapter (5) + prefix tuning	88.3	**77.4**	18.01	**100.86**	6.90%
Parallel Adapter (3)	87.4	76.1	17.58	99.33	3.96%
Parallel Adapter (3) + prefix tuning	**88.3**	**78.4**	**19.38**	**102.73**	4.27%
Parallel Adapter (5)	86.7	76.9	19.15	100.95	6.60%
Parallel Adapter (5) + prefix tuning	86.5	75.2	18.92	99.77	6.90%

4.5. Low-Resource Conditions

This experiment examined how effectively PEFTTOD can transfer conversational knowledge under low-resource conditions. The MultiWOZ 2.0 dataset was used, with training conducted using 1%, 5%, 10%, and 20% of the available training data. As presented in the results in Table 4, when utilizing PEFTTOD with only 4.27% of the parameters compared to the baseline, the performance decreased at low-resource levels of 1% and 5%,

but improved at higher-resource levels of 10% and 20%. This indicated that, even when PEFTTOD learns from a small number of parameters, if it exceeds the threshold of 10% on MultiWOZ 2.0, the performance begins to show improvement.

Table 4. Experimental results for low-resource conditions. MultiWOZ 2.0 was tested on 1%, 5%, 10%, and 20% of the training data (PEFTTOD is a proposed model that uses prefix tuning after connecting a parallel Adapter three times in series).

Model	Inform	Success	BLEU	Comb.
1% of training data				
Baseline	**66.5**	**51.1**	**12.05**	**70.85**
PEFTTOD	51.3	34.7	9.64	52.64
5% of training data				
Baseline	**80.0**	**63.1**	14.82	**86.37**
PEFTTOD	76.6	54.3	**17.03**	82.48
10% of training data				
Baseline	79.5	65.6	**16.73**	89.28
PEFTTOD	**84.5**	**69.7**	15.98	**93.08**
20% of training data				
Baseline	**85.4**	69.0	15.77	92.97
PEFTTOD	82.9	**70.9**	**17.17**	**94.07**

4.6. Prefix Length

In this experiment, we investigated the optimal length of the learnable vectors P_k and P_v, in the prefix tuning, as illustrated in Figure 5. We explored the range of lengths for P_k and P_v from 3 to 15 to determine the optimal value. The results revealed that the optimal prefix length for PEFTTOD was 10. The results indicated that the optimal prefix length for PEFTTOD was 10. Therefore, finding the optimal prefix length was crucial to achieving the best performance (Figure 6).

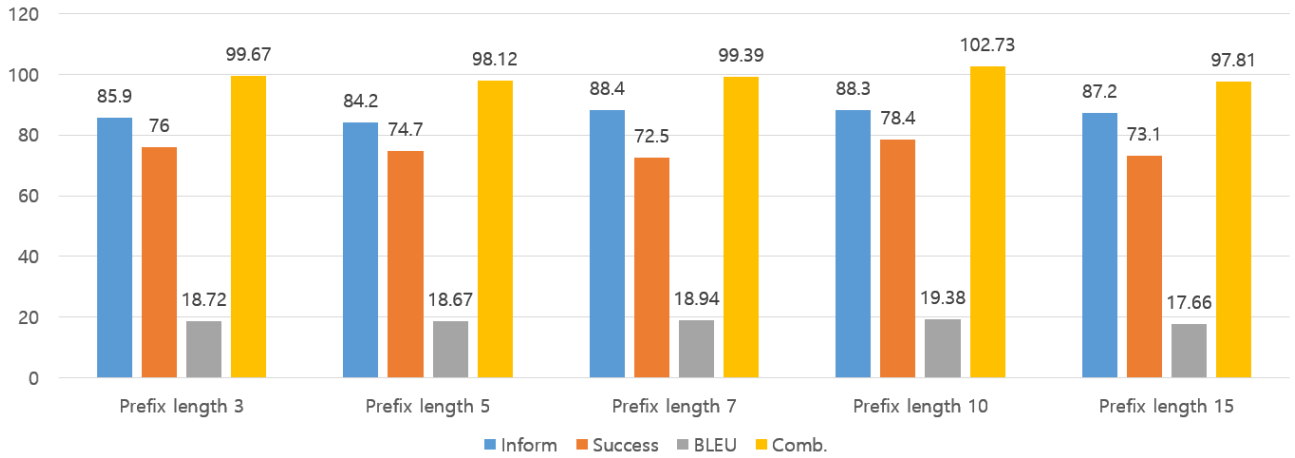

Figure 6. Experimental results for prefix length.

4.7. Efficiency

In order to evaluate the efficiency of PEFTTOD, we conducted experiments focusing on the training time and storage space. PEFTTOD takes advantage of the PEFT method by training only the Adapter Layers, without updating the baseline parameters. As a result, the training process is faster compared to traditional methods. Additionally, since only the parameters corresponding to the trained Adapter Layers are stored, significant storage space is saved.

The evaluation results in Table 5 show that PEFTTOD improved the training time by over 20%, while utilizing only 4% of the parameters compared to the baseline model. Additionally, it achieved a remarkable 96% savings in the storage space requirement. These findings highlight the efficiency gains achieved by adopting PEFTTOD in task-oriented dialogue systems.

Table 5. Experimental results for Efficiency (PEFTTOD uses prefix tuning after connecting a parallel Adapter three times in series).

Model	Training Time	Storage Space	Trainable Parameter
Baseline	1109 s (100%)	240 M (100%)	60.5 M (100%)
PEFTTOD	**882 s (79.5%)**	**10 M (4.27%)**	**2.5 M (4.27%)**

5. Conclusions and Future Work

This paper proposed a novel task-oriented dialogue system, called PEFTTOD, which incorporates the parameter-efficient language-model-tuning method. PEFTTOD leverages parallel Adapters and prefix tuning to efficiently train the conversation knowledge within a task-oriented dialogue system. Through experiments, we obtained the optimal Adapter structure and the number of stacks, and the effectiveness of combining the prefix tuning was demonstrated. The evaluation results revealed an improvement in the combined score, an evaluation metric of the Multi-Woz dataset, by 4% compared to the existing T5-based baseline model. Furthermore, despite utilizing only around 4% of the parameters compared to the baseline model, notable efficiency gains were achieved, including a 20% improvement in training speed and an approximately 96% reduction in storage space requirements.

As future work, we intend to extend our proposal to the open-domain dialogue systems rather than being limited to the task-oriented dialogue systems. Additionally, we plan to explore Adapters suitable for the ever-increasing large-scale pre-trained languages, in order to validate their effectiveness.

Author Contributions: Conceptualization, Y.M. and S.K.; methodology, Y.M.; validation, Y.M.; investigation, Y.M. and S.K.; resources, Y.M. and S.K.; data curation, Y.M.; writing—original draft preparation, Y.M. and J.Y.; writing—review and editing, Y.M. and J.Y.; visualization, Y.M. and J.Y.; supervision, S.K. and J.Y.; project administration, S.K.; funding acquisition, S.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was in part supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (2022R1A2C1005316 and 2021R1F1A1063640) and in part by the Gachon University research fund of 2021 (GCU-202106470001).

Data Availability Statement: Data sharing not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

NLU	Natural Language Understanding
DST	Dialogue State Tracking
POL	Dialogue Policy Learning
NLG	Natural Language Generation
PEFT	Parameter-Efficient Fine-Tuning method
TOD	Task-Oriented Dialogue system

References

1. Young, S.J. Probabilistic methods in spoken–dialogue systems. *Philos. Trans. R. Soc. Lond. Ser. A Math. Phys. Eng. Sci.* **2000**, *358*, 1389–1402. [CrossRef]
2. Su, Y.; Shu, L.; Mansimov, E.; Gupta, A.; Cai, D.; Lai, Y.A.; Zhang, Y. Multi-task pre-training for plug-and-play task-oriented dialogue system. *arXiv* **2021**, arXiv:2109.14739.
3. Lin, Z.; Madotto, A.; Winata, G.I.; Fung, P. Mintl: Minimalist transfer learning for task-oriented dialogue systems. *arXiv* **2020**, arXiv:2009.12005.
4. Lee, Y. Improving end-to-end task-oriented dialog system with a simple auxiliary task. Findings of the Association for Computational Linguistics. In Proceedings of the EMNLP 2021, Punta Cana, Dominican Republic, 7 November 2021; pp. 1296–1303.
5. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems 30, Proceedings of the NIPS, Long Beach, CA, USA, 4–9 December 2007*; MIT Press: Cambridge, MA, USA, 2007; pp. 5998–6008.
6. Pan, S.J.; Yang, Q. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.* **2010**, *22*, 1345–1359. [CrossRef]
7. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33, Proceedings of the NIPS 2020, Vancouver, BC, Canada, 6–12 December 2020*; MIT Press: Cambridge, MA, USA, 2020; pp. 1877–1901.
8. Houlsby, N.; Giurgiu, A.; Jastrzebski, S.; Morrone, B.; De Laroussilhe, Q.; Gesmundo, A.; Attariyan, M.; Gelly, S. Parameter-efficient transfer learning for NLP. In Proceedings of the International Conference on Machine Learning, PMLR, Long Beach, CA, USA, 9–15 June 2019; pp. 2790–2799.
9. Li, X.L.; Liang, P. Prefix tuning: Optimizing continuous prompts for generation. *arXiv* **2021**, arXiv:2101.00190.
10. Mangrulkar, S.; Gugger, S.; Debut, L.; Belkada, Y.; Paul, S. PEFT: State-of-the-Art Parameter-Efficient Fine-Tuning Methods. 2022 Available online: https://github.com/huggingface/peft (accessed on 6 July 2023).
11. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text Transformer. *J. Mach. Learn. Res.* **2020**, *21*, 5485–5551.
12. Budzianowski, P.; Wen, T.H.; Tseng, B.H.; Casanueva, I.; Ultes, S.; Ramadan, O.; Gašić, M. MultiWOZ—A large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modeling. *arXiv* **2018**, arXiv:1810.00278.
13. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional Transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
14. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
15. He, P.; Liu, X.; Gao, J.; Chen, W. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv* **2020**, arXiv:2006.03654.
16. Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving Language Understanding by Generative Pre-Training. Technical Report. OpenAI. 2018. Available online: https://www.mikecaptain.com/resources/pdf/GPT-1.pdf (accessed on 6 July 2023).
17. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language models are unsupervised multitask learners. *OpenAI Blog* **2019**, *1*, 9.
18. Thoppilan, R.; De Freitas, D.; Hall, J.; Shazeer, N.; Kulshreshtha, A.; Cheng, H.T.; Jin, A.; Bos, T.; Baker, L.; Du, Y.; et al. Lamda: Language models for dialog applications. *arXiv* **2022**, arXiv:2201.08239.
19. Zhang, S.; Roller, S.; Goyal, N.; Artetxe, M.; Chen, M.; Chen, S.; Dewan, C.; Diab, M.; Li, X.; Lin, X.V.; et al. Opt: Open pre-trained Transformer language models. *arXiv* **2022**, arXiv:2205.01068.
20. Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; Zettlemoyer, L. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv* **2019** arXiv:1910.13461.
21. Wen, T.H.; Vandyke, D.; Mrksic, N.; Gasic, M.; Rojas-Barahona, L.M.; Su, P.H.; Ultes, S.; Young, S. A network-based end-to-end trainable task-oriented dialogue system. *arXiv* **2016**, arXiv:1604.04562.
22. Lei, W.; Jin, X.; Kan, M.Y.; Ren, Z.; He, X.; Yin, D. Sequicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Melbourne, Australia, 15–20 July 2018; pp. 1437–1447.
23. Zhang, Y.; Ou, Z.; Yu, Z. Task-oriented dialog systems that consider multiple appropriate responses under the same context. In Proceedings of the AAAI Conference on Artificial Intelligence, Hilton, NY, USA, 7–12 February 2020; Volume 34, pp. 9604–9611.
24. Zhang, Y.; Ou, Z.; Wang, H.; Feng, J. A probabilistic end-to-end task-oriented dialog model with latent belief states towards semi-supervised learning. *arXiv* **2020**, arXiv:2009.08115.
25. Tseng, B.H.; Dai, Y.; Kreyssig, F.; Byrne, B. Transferable dialogue systems and user simulators. *arXiv* **2021**, arXiv:2107.11904.
26. Lubis, N.; Geishauser, C.; Heck, M.; Lin, H.c.; Moresi, M.; van Niekerk, C.; Gašić, M. LAVA: Latent action spaces via variational auto-encoding for dialogue policy optimization. *arXiv* **2020**, arXiv:2011.09378.
27. Jeon, H.; Lee, G.G. DORA: Towards policy optimization for task-oriented dialogue system with efficient context. *Comput. Speech Lang.* **2022**, *72*, 101310. [CrossRef]
28. Lee, H.; Jo, S.; Kim, H.; Jung, S.; Kim, T.Y. Sumbt+ larl: Effective multi-domain end-to-end neural task-oriented dialog system. *IEEE Access* **2021**, *9*, 116133–116146. [CrossRef]

29. Ramachandran, G.S.; Hashimoto, K.; Xiong, C. Causal-aware safe policy improvement for task-oriented dialogue. *arXiv* **2021**, arXiv:2103.06370.
30. Jeon, H.; Lee, G.G. Domain state tracking for a simplified dialogue system. *arXiv* **2021**, arXiv:2103.06648.
31. Hosseini-Asl, E.; McCann, B.; Wu, C.S.; Yavuz, S.; Socher, R. A simple language model for task-oriented dialogue. In *Advances in Neural Information Processing Systems 33, Proceedings of the Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, Virtual, 6–12 December 2020*; MIT Press: Cambridge, MA, USA, 2020; pp. 20179–20191.
32. Peng, B.; Li, C.; Li, J.; Shayandeh, S.; Liden, L.; Gao, J. Soloist: Building task bots at scale with transfer learning and machine teaching. *Trans. Assoc. Comput. Linguist.* **2021**, *9*, 807–824. [CrossRef]
33. Chopra, S.; Hadsell, R.; LeCun, Y. Learning a similarity metric discriminatively, with application to face verification. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; Volume 1, pp. 539–546.
34. Yang, Y.; Li, Y.; Quan, X. Ubar: Towards fully end-to-end task-oriented dialog system with gpt-2. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 2–9 February 2021; Volume 35, pp. 14230–14238.
35. He, W.; Dai, Y.; Zheng, Y.; Wu, Y.; Cao, Z.; Liu, D.; Jiang, P.; Yang, M.; Huang, F.; Si, L.; et al. Galaxy: A generative pre-trained model for task-oriented dialog with semi-supervised learning and explicit policy injection. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual Event, 22 February–1 March 2022; Volume 36, pp. 10749–10757.
36. Lee, J.; Tang, R.; Lin, J. What would elsa do? freezing layers during Transformer fine-tuning. *arXiv* **2019**, arXiv:1911.03090.
37. Ravfogel, S.; Ben-Zaken, E.; Goldberg, Y. Bitfit: Simple parameter-efficient fine-tuning for Transformer-based masked language-models. *arXiv* **2021**, arXiv:2106.10199.
38. Pfeiffer, J.; Kamath, A.; Rücklé, A.; Cho, K.; Gurevych, I. AdapterFusion: Non-destructive task composition for transfer learning. *arXiv* **2020**, arXiv:2005.00247.
39. Zhu, Y.; Feng, J.; Zhao, C.; Wang, M.; Li, L. Counter-interference Adapter for multilingual machine translation. *arXiv* **2021**, arXiv:2104.08154.
40. Bapna, A.; Arivazhagan, N.; Firat, O. Simple, scalable adaptation for neural machine translation. *arXiv* **2019**, arXiv:1909.08478.
41. Pfeiffer, J.; Vulić, I.; Gurevych, I.; Ruder, S. Mad-x: An Adapter-based framework for multi-task cross-lingual transfer. *arXiv* **2020**, arXiv:2005.00052.
42. Hu, E.J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; Chen, W. Lora: Low-rank adaptation of large language models. *arXiv* **2021**, arXiv:2106.09685.
43. He, J.; Zhou, C.; Ma, X.; Berg-Kirkpatrick, T.; Neubig, G. Towards a unified view of parameter-efficient transfer learning. *arXiv* **2021**, arXiv:2110.04366.
44. Mao, Y.; Mathias, L.; Hou, R.; Almahairi, A.; Ma, H.; Han, J.; Yih, W.t.; Khabsa, M. Unipelt: A unified framework for parameter-efficient language model tuning. *arXiv* **2021**, arXiv:2110.07577.
45. Nekvinda, T.; Dušek, O. Shades of BLEU, flavours of success: The case of MultiWOZ. *arXiv* **2021**, arXiv:2106.05555.
46. Aghajanyan, A.; Zettlemoyer, L.; Gupta, S. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. *arXiv* **2020**, arXiv:2012.13255.
47. Papineni, K.; Roukos, S.; Ward, T.; Zhu, W.J. Bleu: A method for automatic evaluation of machine translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Philadelphia, PA, USA, 6–12 July 2002; pp. 311–318.
48. Mehri, S.; Srinivasan, T.; Eskenazi, M. Structured fusion networks for dialog. *arXiv* **2019**, arXiv:1907.10016.