Domain-Lifelong Learning for Dialogue State Tracking via Knowledge Preservation Networks

Qingbin Liu[1,2], **Pengfei Cao**[1,2], **Cao Liu**[3], **Jiansong Chen**[3], **Xunliang Cai**[3], **Fan Yang**[3], **Shizhu He**[1,2], **Kang Liu**[1,2], **Jun Zhao**[1,2]
[1] National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing, China
[3] Meituan, Beijing, China
{qingbin.liu, pengfei.cao}@nlpr.ia.ac.cn,
{liucao, chenjiansong, caixunliang, yangfan79}@meituan.com,
{shizhu.he, kliu, jzhao}@nlpr.ia.ac.cn

Abstract

Dialogue state tracking (DST), which estimates user goals given a dialogue context, is an essential component of task-oriented dialogue systems. Conventional DST models are usually trained offline, which requires a fixed dataset prepared in advance. This paradigm is often impractical in real-world applications since online dialogue systems usually involve continually emerging new data and domains. Therefore, this paper explores Domain-Lifelong Learning for Dialogue State Tracking (DLL-DST), which aims to continually train a DST model on new data to learn incessantly emerging new domains while avoiding catastrophically forgetting old learned domains. To this end, we propose a novel domain-lifelong learning method, called Knowledge Preservation Networks (KPN), which consists of multi-prototype enhanced retrospection and multi-strategy knowledge distillation, to solve the problems of expression diversity and combinatorial explosion in the DLL-DST task. Experimental results show that KPN effectively alleviates catastrophic forgetting and outperforms previous state-of-the-art lifelong learning methods by 4.25% and 8.27% of whole joint goal accuracy on the MultiWOZ benchmark and the SGD benchmark, respectively.

1 Introduction

Task-oriented dialogue systems aim at helping users to accomplish various tasks, such as reserving restaurants, booking flights, and checking weather (Young et al., 2013; Lei et al., 2018; Gao et al., 2020). Dialogue state tracking (DST) is an essential component of task-oriented dialogue systems, which estimates user goals for downstream modules (Bohus and Rudnicky, 2006; Williams et al., 2013; Henderson et al., 2014b; Mrkšić et al., 2017; Shan et al., 2020). Given a user utterance and its dialogue history, a DST model should be able to output an accurate dialogue state. In general, the dialogue state is represented as a set of slot-value pairs, such as ((*restaurant-area*, *north*), (*restaurant-price*, *expensive*)).

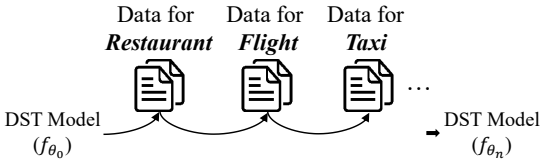


Figure 1: An example of domain-lifelong learning for DST. Italicized words denote domains. The DST model requires lifelong learning of new domains.

Previous DST models are usually trained offline, which requires a fixed dataset prepared in advance. These offline solutions are often impractical in real-world applications, as online dialogue systems usually involve continually emerging new data and domains, especially when new services are introduced. In addition, it is infeasible to retrain DST models from scratch every time new domain data arrives due to computational costs, storage budgets, and data privacy (McMahan et al., 2017). To tackle this realistic issue, we explore Domain-Lifelong Learning for Dialogue State Tracking (DLL-DST). As shown in Figure 1, the DLL-DST task aims to continually train a DST model on new data to learn incessantly emerging new domains. At each step, new data generally contains one or multiple new domains, and the updated model should be able to make accurate predictions for all the domains observed so far.

A plain approach to domain-lifelong learning is to simply fine-tune a pre-trained model on new data. However, this approach suffers from the problem of catastrophic forgetting (McCloskey and Cohen, 1989; French, 1999). To be more specific, fine-

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 2301–2311
November 7–11, 2021.

State: restaurant-price=expensive, restaurant-area=north

User 1: I want an upscale restaurant in the northern part.
User 2: Hello, I want an expensive restaurant in the north.
User 3: Is there a fine dining restaurant in the north?

Figure 2: An example of expression diversity. Different users have different expressions for a dialogue state.

tuning the model on new data usually results in a significant performance drop on old data. To address this problem, there are two mainstream lifelong learning methods: (1) regularization-based methods, which try to identify and preserve the parameters important to old data (Kirkpatrick et al., 2017; Aljundi et al., 2018); (2) replay-based methods, which reserve some representative old samples and combine them with new data to re-train the model (Rebuffi et al., 2017; Wang et al., 2019; Hou et al., 2019). Recently, replay-based methods have shown promising results in alleviating catastrophic forgetting of class-lifelong learning tasks in NLP scenarios (Han et al., 2020; Cao et al., 2020).

However, when deploying previous replay-based methods on the DLL-DST task, we find two main problems: expression diversity and combinatorial explosion. **Expression diversity:** In the DST task, dialogue texts usually contain a variety of expressions for each dialogue state, as shown in Figure 2. The expression diversity makes it difficult for previous replay-based methods to select the most representative old samples. The unrepresentative samples, such as the first utterance in Figure 2, do not contain typical expressions for any slot. Retraining models with these unrepresentative samples is not conducive to retaining the performance on old domains. **Combinatorial explosion:** Ideally, we should reserve at least one sample for each dialogue state in old domains. However, the classes of dialogue states explode rapidly as the number of slot-value pairs increases. For example, the MultiWOZ 2.1 dataset (Eric et al., 2019) has an average of 2732 classes of dialogue states per domain. In the DLL-DST task, it is infeasible for replay-based methods to reserve samples for each class of dialogue states in old domains due to limitations of memory capacity and training time. Since the reserved samples involve only a few types of dialogue states, previous methods may gradually forget previous knowledge, leading to catastrophic forgetting.

To address the above two problems, we propose Knowledge Preservation Networks (KPN), which contain two main components: (1) to handle expression diversity, we propose multi-prototype enhanced retrospection, which computes multiple slot prototypes for each domain and selects the most representative old samples based on these slot prototypes; (2) to cope with the combinatorial explosion problem, we propose multi-strategy knowledge distillation, which enables the model at the current step to preserve the knowledge of the model trained in the last step from multiple aspects, instead of just reserving some old samples. Experimental results demonstrate that KPN outperforms previous state-of-the-art lifelong learning methods by 4.25% and 8.27% of whole joint goal accuracy on the MultiWOZ benchmark and the SGD benchmark, respectively. The contributions of this paper are listed as follows:

- To the best of our knowledge, we are the first to formally introduce domain-lifelong learning into dialogue state tracking and we construct two benchmarks through two widely used DST datasets, MultiWOZ 2.1 and SGD.
- We propose Knowledge Preservation Networks, which handle expression diversity and combinatorial explosion in the DLL-DST task via multi-prototype enhanced retrospection and multi-strategy knowledge distillation.
- Experimental results show that our method outperforms previous lifelong learning methods and achieves state-of-the-art performance. We will release the source code and the benchmarks for further research (`https://github.com/liuqingbin/Knowledge-Preservation-Networks`).

2 Task Formulation

The DST task is usually formulated as a slot-filling task (Bohus and Rudnicky, 2006; Williams et al., 2013). At each dialogue turn, the DST model takes the user utterance and the dialogue history as input and predicts values for each slot. As shown in Figure 2, the DST model is expected to fill the slot "*restaurant-price*" with the value "*expensive*".

The DLL-DST task continually trains DST models on emerging data to learn new domains. New data arrives in a stream form ($\{\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_N\}$). At each step, the new data ($\mathcal{D}_i$) can contain one or multiple new domains. In addition, inspired by other lifelong learning work (Lopez-Paz and Ranzato, 2017; Zenke et al., 2017), we treat dialogues

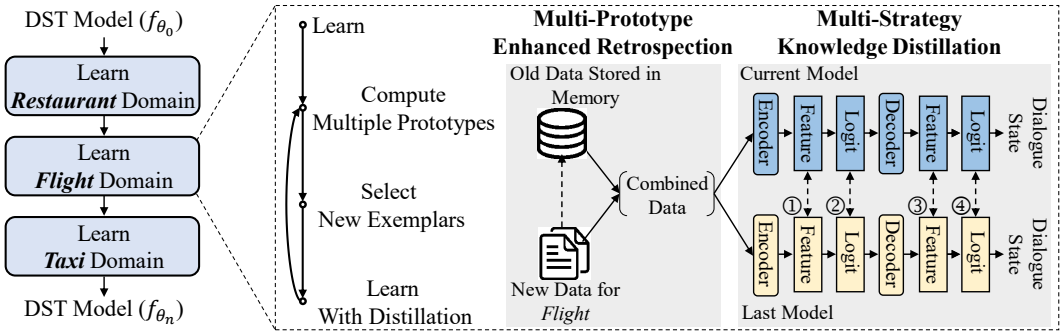


Figure 3: An example of continually learning three domains to demonstrate the framework of KPN. When learning the *Flight* domain, the model is updated with the combination of training data for the *Flight* domain and the old data stored in memory. KPN adopts multi-strategy knowledge distillation to retain previous knowledge. The numbers 1 to 4 represent four different knowledge distillation strategies. Then, the method selects new representative samples to reserve via multi-prototype enhanced retrospection.

across the same multiple domains as data of a special domain, since these cross-domain dialogues usually contain specific expressions that distinguish them from other dialogues, such as domain transformation and slot reference (Ouyang et al., 2020; Hu et al., 2020). Each new data has its own training/validation/test set ($\mathcal{D}_i^{\text{train}}, \mathcal{D}_i^{\text{dev}}, \mathcal{D}_i^{\text{test}}$). When new data ($\mathcal{D}_i$) arrives, the DST model is optimized using the new training data ($\mathcal{D}_i^{\text{train}}$). The updated model should still perform well on all previous domains. Therefore, in the testing stage of the i-th step, we evaluate the updated model on the test data of all observed domains (i.e., $\bigcup_{k=1}^{i} \mathcal{D}_k^{\text{test}}$). The evaluation protocol indicates that it will be more and more difficult to achieve high performance for DST models as more and more domains arrive.

3 Method

In this work, we propose Knowledge Preservation Networks (KPN) to handle the DLL-DST task. KPN consists of two core components, i.e., multi-prototype enhanced retrospection and multi-strategy knowledge distillation, for dealing with the two main challenges, i.e., expression diversity and combinatorial explosion. The framework of KPN is shown in Figure 3.

3.1 Background

Our method, KPN, is a lifelong learning framework, which is model-agnostic. The DST model is only a basic component, not our research focus. DST models, such as TRADE (Wu et al., 2019), SAS (Hu et al., 2020), and SOM-DST (Kim et al., 2020), can all be used as this basic component. We adopt the previous best model, SOM-DST, in this work.

In each dialogue turn, SOM-DST simplifies the dialogue history to the last system response and the last dialogue state, and then combines them with the current user utterance into an input sequence for the BERT encoder (Devlin et al., 2019). BERT is a multi-layer Transformer (Vaswani et al., 2017), pre-trained on large-scale unlabeled corpora. To fit the input form of BERT, the tokens [CLS] and [SEP] are placed in the input sequence. In addition, a special token [SLOT] is placed at the beginning of each slot in the last dialogue state. The BERT encoder obtains the contextual representation for the input sequence. The encoded hidden state of [SLOT] is used as the feature vector of each slot.

For each slot, SOM-DST first classifies it into four categories, including "dontcare", "carryover", "update", and "delete". "dontcare" means that the user does not care about this slot. "carryover" indicates that the slot inherits the value of the same slot from the last dialogue state. "update" means that the model needs to generate a value for the slot. "delete" means that this slot does not contain any value. A softmax classifier is added to the feature vector of each slot to predict its category. The cross-entropy loss is used to train the classifier:

$$\mathcal{L}_c = -\frac{1}{|\mathcal{N}|} \sum_{x \in \mathcal{N}} \sum_{s \in \mathcal{C}} \boldsymbol{y}^s \log(\boldsymbol{p}^s) \qquad (1)$$

where $\boldsymbol{y}^s$ is the one-hot label for the slot s and $\boldsymbol{p}^s$ is the predicted probability. $\mathcal{N}$ is the training samples and $\mathcal{C}$ is the slots of all observed domains.

For each slot belonging to the "update" category, SOM-DST generates a value for this slot via the GRU decoder (Cho et al., 2014). The decoder is equipped with the ability to copy words from the input sequence (Kim et al., 2020). The cross-entropy

loss is used to train the generation probability:

$$\mathcal{L}_g = -\frac{1}{|\mathcal{N}|}\sum_{x\in\mathcal{N}}\sum_{s\in\mathcal{U}}\sum_{i\in d}\boldsymbol{y}_i^v \log(\boldsymbol{p}^v(v_i|v_{<i})) \quad (2)$$

where $\boldsymbol{p}^v(v_i|v_{<i})$ is the predicted probability of the i-th word of the value v. $\boldsymbol{y}_i^v$ is the one-hot label. d is the length of the value. $\mathcal{U}$ is the slots that are predicted to be the "update" category.

3.2 Multi-Prototype Enhanced Retrospection

In this paper, we focus on the domain-lifelong learning scenario for DST. Given a model trained on old data, we aim to continually learn a unified DST model for all observed domains so far based on a new combined dataset $\mathcal{N} = \mathcal{D}_i^{\text{train}} \bigcup \mathcal{P}$. $\mathcal{D}_i^{\text{train}}$ is the training data of the new domains at step i. $\mathcal{P}$ is a bounded memory that stores representative old samples, denoted as $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_m\}$. $\mathcal{P}_i$ is the set of stored samples for the i-th domain. m is the number of old domains.

Since the DLL-DST task suffers from expression diversity, we propose multi-prototype enhanced retrospection to reserve the most representative samples of old domains. In this way, important information about the data distribution of all previous domains enters the subsequent training process. This approach is inspired by prototype learning (Snell et al., 2017; Yang et al., 2018), which uses prototypes as representative points.

Specifically, after learning on the new domains, we store $|\mathcal{P}_i| = B/m$ samples for each new domain. m is the number of all observed domains and B is the total number of samples that can be reserved. We encode all samples of the i-th domain into the hidden representation and compute a prototype $\boldsymbol{\mu}^s$ for each slot s in this domain:

$$\boldsymbol{\mu}^s = \frac{1}{|N|}\sum_{x\in N} f^s(x) \quad (3)$$

where N is the training samples of the i-th domain. $f^s(x)$ is the hidden state of [SLOT] in front of the slot s, which is the slot representation.

Then, we compute the distance between the slot representation of each training sample and the corresponding slot prototype. Based on the average distance of all slots, we produce a sorted list of new training samples. Intuitively, the closer the samples to these slot prototypes, the more representative the samples will be for these slots. Based on the sorted list of samples, the top B/m samples are selected as exemplars to be stored in the memory. Since the storage size of memory is constant, when new domains arrive, the memory needs to remove some reserved exemplars of old domains to allocate space for the exemplars of new domains. Suppose the number of new domains is t. The memory needs to remove $B/(m-t) - B/m$ stored samples of each old domain. For each old domain, we remove the samples that are far from these prototypes according to the sorted list.

3.3 Multi-Strategy Knowledge Distillation

As mentioned above, just reserving some old samples makes previous lifelong learning methods still suffer from combinatorial explosion. Since the reserved samples involve only a few types of dialogue states, these methods may gradually forget the previous knowledge. To handle this problem, we propose multi-strategy knowledge distillation, which preserves the knowledge of the model trained in the last step through multiple distillation strategies. In this way, the current model can perform well on the old domains. Knowledge distillation is an effective way to transfer knowledge from one network to another (Hinton et al., 2015).

3.3.1 Encoder Feature Distillation

For each slot, we denote its feature vector extracted by the BERT encoder of the current model and the BERT encoder of the last model as $f^s(x)$ and $f^{s,*}(x)$, respectively. To preserve the feature distribution in the original encoder, we adopt an encoder feature distillation loss:

$$\mathcal{L}_{ef} = \frac{1}{|\mathcal{N}|}\sum_{x\in\mathcal{N}}\sum_{s\in\mathcal{C}} 1 - \langle f^s(x), f^{s,*}(x)\rangle \quad (4)$$

where $\langle f^s(x), f^{s,*}(x)\rangle$ measures the cosine similarity between the two features. This loss is computed for all samples from the new domains and the reserved exemplars. If the features of the current encoder don't greatly deviate from those of the last encoder, the current model can effectively preserve the knowledge of the model trained in the last step.

3.3.2 Classifier Prediction Distillation

In addition, we also adopt a classifier prediction distillation, which preserves the previous knowledge by encouraging the predictions of the current classifier to match the predictions of the last classifier. For each slot, the classification logits (i.e., the results before the softmax layer) of the current model and the last model are $\boldsymbol{o} = [\boldsymbol{o}_1, \boldsymbol{o}_2, ..., \boldsymbol{o}_\eta]$ and $\boldsymbol{o}^* = [\boldsymbol{o}_1^*, \boldsymbol{o}_2^*, ..., \boldsymbol{o}_\eta^*]$, respectively. $\eta = 4$ in

this classifier. The classifier prediction distillation loss is formulated as:

$$\mathcal{L}_{cp} = -\frac{1}{|\mathcal{N}|}\sum_{x\in\mathcal{N}}\sum_{s\in\mathcal{C}}\sum_{i=1}^{\eta}\gamma_i^*\log(\gamma_i)$$
$$\gamma_i^* = \frac{e^{\boldsymbol{o}_i^*/T}}{\sum_{j=1}^{\eta} e^{\boldsymbol{o}_j^*/T}},\ \gamma_i = \frac{e^{\boldsymbol{o}_i/T}}{\sum_{j=1}^{\eta} e^{\boldsymbol{o}_j/T}} \quad (5)$$

where T is the temperature scalar. T is usually greater than 1 to increase the weights of small probability values. The classifier prediction distillation loss is also computed for the training samples of the new domains and the reserved exemplars of the old domains.

3.3.3 Decoder Feature Distillation

To retain the previous knowledge of the last decoder, we adopt a decoder feature distillation loss to learn the feature distribution of the last decoder.

$$\mathcal{L}_{df} = \frac{1}{|\mathcal{N}|}\sum_{x\in\mathcal{N}}\sum_{s\in\mathcal{U}}\sum_{i\in d} 1 - \langle g_i(x), g_i^*(x)\rangle \quad (6)$$

where $g_i(x)$ and $g_i^*(x)$ are the i-th hidden state decoded by the current decoder and the last decoder for the slot s.

3.3.4 Generation Prediction Distillation

We adopt another prediction distillation loss $\mathcal{L}_{gp}$ to mimic the generation probability predicted by the last decoder. Because the sigmoid function in the decoder (Kim et al., 2020) makes it impossible to adopt the above prediction distillation loss, we use the KL-divergence as the generation prediction distillation loss as follows:

$$\mathcal{L}_{gp} = \frac{1}{|\mathcal{N}|}\sum_{x\in\mathcal{N}}\sum_{s\in\mathcal{U}}\sum_{i\in d}\boldsymbol{q}_i\log(\frac{\boldsymbol{q}_i}{\boldsymbol{p}_i}) \quad (7)$$

where $\boldsymbol{p}_i$ and $\boldsymbol{q}_i$ are the i-th probability predicted by the current and last decoder for the slot s.

3.4 Training

During each step of the domain-lifelong learning process, we combine the above losses to train the DST model:

$$\mathcal{L} = \mathcal{L}_c + \mathcal{L}_g + \alpha(\mathcal{L}_{ef} + \mathcal{L}_{df}) + \beta(\mathcal{L}_{cp} + \mathcal{L}_{gp}) \quad (8)$$

where α and β are two adjustment coefficients. The coefficients are used to balance the performance of the old domains and the new domains. If α and β are very small, the model will pay more attention to the new domains, thus hurting the performance of the old domains. At each step, we combine the training set $(\mathcal{D}_i^{\text{train}} \bigcup \mathcal{P})$ to train the model with the loss $\mathcal{L}$, and then select the most representative samples to update the memory. Therefore, our method can continually learn new domains while avoiding catastrophically forgetting old domains.

Benchmark	Training	Validation	Test	Slot-Values	Steps
MultiWOZ	6343	775	787	1075	10
SGD	3217	1073	1080	5235	15

Table 1: Statistics of the MultiWOZ and SGD benchmarks. "Training" is the number of training dialogues.

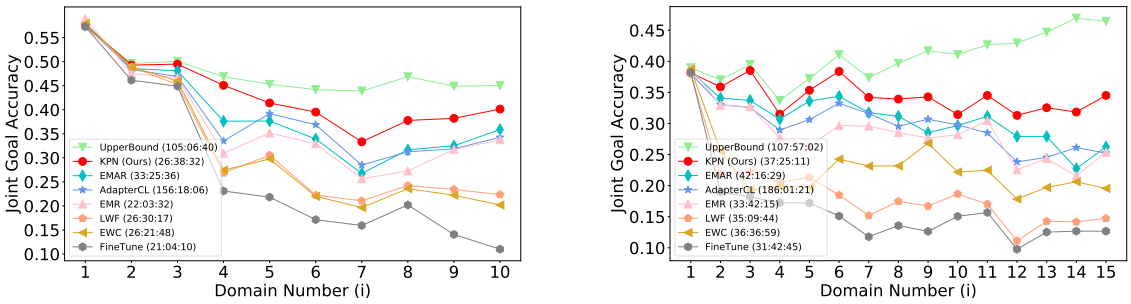
4 Experiments

4.1 DLL-DST Benchmarks

To the best of our knowledge, we are the first to formally introduce the DLL-DST task. Therefore, we construct two benchmarks based on the following method: for a given DST dataset, we arrange its domains in a fixed random order. Each domain has its own data and ontology (i.e., slot-value pairs). In a domain incremental manner, the lifelong learning methods continually train a DST model on one or multiple new domains. Following other tasks (Li and Hoiem, 2017; Cao et al., 2020), we adopt one new domain at each step. As described in Section 2, inspired by other tasks, we treat dialogues across the same multiple domains as data of a special new domain, since they usually contain many specific expressions. Based on two widely used datasets, MultiWOZ 2.1 (Eric et al., 2019) and SGD (Rastogi et al., 2019), we propose two instantiations of the above construction method. MultiWOZ benchmark: We use the data splitting of the official MultiWOZ 2.1 dataset. Since the domains in MultiWOZ 2.1 has a long-tail frequency distribution, we use the data of the top 10 most frequent domains (including the combined domains). SGD benchmark: Same as the MultiWOZ benchmark, we use the data of the top 15 most frequent domains. Table 1 shows the statistics of the two benchmarks.

4.2 Experimental Settings

For the DST task, joint goal accuracy (JGA) is used as the evaluation metric (Zhong et al., 2018). For the DLL-DST task, every time the model finishes training on new domains, we report JGA on the test data of all observed domains. For example, after the i-th step, the result is denoted as JGA_i. In addition, after the last step, we report **Aver-**



(a) MultiWOZ Benchmark (b) SGD benchmark

Figure 4: Performance (JGA_i) changes with increasing domains on the MultiWOZ benchmark (a) and the SGD benchmark (b). The **training time**, measured on GeForce RTX 2080Ti, is shown in the brackets.

age JGA which is the average score of all steps ($\frac{1}{k}\sum_{i=1}^{k}\text{JGA}_i$), and **Whole JGA** which is the JGA score on the whole testing data of all domains.

Our method uses the HuggingFace's Transformers library[1] to implement the BERT-based DST model. The learning rate is set to $5e-5$. The batch size is 4. The hyper-parameters α and β are 0.2 and 0.1, respectively. $T = 2$ in our experiments. The capacity of memory is 50. All hyper-parameters are obtained by a grid search on the validation set.

4.3 Baselines

In this work, we propose a model-agnostic lifelong learning method to handle the DLL-DST task. Therefore, we adopt other model-agnostic lifelong learning methods that achieve state-of-the-art performance on other tasks as our baselines:

EWC (Kirkpatrick et al., 2017), which slows down the update of important parameters by adding a L_2 regularization of parameter changes.

LwF (Li and Hoiem, 2017), which matches the prediction of the current network with that of the original network by knowledge distillation.

EMR (Wang et al., 2019), which alleviates forgetting by randomly storing some old samples.

AdapterCL (Madotto et al., 2020), which adds the model parameters to learn new data.

EMAR (Han et al., 2020), which selects representative samples based on only one prototype and consolidates the model through the prototype.

FineTune, which simply fine-tunes the pre-trained model on new data.

UpperBound, which uses training samples from all observed domains to train the model. We regard it as the upper bound of the benchmark.

[1] https://github.com/huggingface

Method	MultiWOZ		SGD	
	Avg	Whole	Avg	Whole
FineTune	27.15	10.99	16.09	12.67
EWC	31.77	20.20	22.87	19.55
LwF	32.21	22.35	18.78	14.74
EMR	37.06	33.75	28.44	25.40
AdapterCL	38.83	34.27	29.75	25.25
EMAR	39.06	35.89	30.79	26.24
KPN (Ours)	**43.19**	**40.14**	**34.43**	**34.51**
UpperBound	47.40	45.02	40.74	46.46

Table 2: Average JGA (%) of all steps ("Avg" column) and whole JGA (%) on the whole testing data ("Whole" column) after the last step.

4.4 Main Results

Figure 4 shows the JGA scores over the observed domains during the whole lifelong learning process. We also list the results after the last step in Table 2. From the results, we can observe that:

(1) Our proposed method KPN significantly outperforms other baselines and achieves state-of-the-art performance in both the MultiWOZ and SGD benchmarks. For example, compared to EMAR, our method achieves 4.25% and 8.27% improvements of the whole joint goal accuracy on the MultiWOZ benchmark and the SGD benchmark, respectively. It verifies the effectiveness of our method on the DLL-DST task.

(2) At each step of the domain-lifelong learning process, there is a performance gap between EMAR and our method KPN. The reason is that EMAR ignores the problems of expression diversity and combinatorial explosion in the DLL-DST task. Therefore, EMAR fails to reserve the most representative samples and tends to gradually forget the previous knowledge of the original model,

Method	MultiWOZ		SGD	
	Avg	Whole	Avg	Whole
KPN (Ours)	**43.19**	**40.14**	**34.43**	**34.51**
- MPR	42.09	38.63	32.17	30.91
+ iCaRL	41.76	37.01	31.21	28.39
+ K-Means	42.57	39.16	32.81	31.61

Table 3: Ablation studies of multi-prototype enhanced retrospection. We compare our method with different data selection methods. iCaRL (Rebuffi et al., 2017) uses one prototype to select samples. K-Means (Han et al., 2020) selects samples by clustering.

Method	MultiWOZ		SGD	
	Avg	Whole	Avg	Whole
KPN (Ours)	**43.19**	**40.14**	**34.43**	**34.51**
- EFD	41.56	37.55	31.67	30.76
- CPD	42.08	39.50	33.09	32.66
- EFD and CPD	40.89	36.85	31.51	28.17
- DFD	42.35	39.17	32.37	28.74
- GPD	41.15	37.86	31.62	28.46
- DFD and GPD	40.31	37.38	30.85	27.49
- MSKD	39.08	36.10	30.06	26.57
- MPR and MSKD	37.06	33.75	28.44	25.40

Table 4: Ablation studies of multi-strategy knowledge distillation. We compare different knowledge distillation strategies.

eventually resulting in catastrophic forgetting. The architecture-based method, AdapterCL, greatly increases the computation time due to the need to select the parameters to be executed. Besides, because AdapterCL only trains domain-specific parameters, it has weak representation capabilities for each domain and achieves low performance.

(3) FineTune always achieves the worst results on both benchmarks, which confirms that catastrophic forgetting is indeed a major difficulty in the DLL-DST task. In addition, there is still a gap between our method and the upper bound. It indicates that, although we have proposed an effective approach for the DLL-DST task, there remain numerous challenges to be addressed.

4.5 Ablation Experiment

In this work, we propose a model-agnostic domain-lifelong learning method, KPN, which consists of two core components: multi-prototype enhanced retrospection and multi-strategy knowledge distillation. In this section, we show ablation studies of the two components.

4.5.1 Effectiveness of Multi-Prototype Enhanced Retrospection

We conduct experiments to verify the effectiveness of the proposed multi-prototype enhanced retrospection. The results are shown in Table 3. From the results, we can see that:

(1) For "- MPR", we remove multi-prototype enhanced retrospection and randomly select samples. Our method KPN outperforms this variant by 1.51% and 3.6% in terms of the whole JGA. The results show that the multi-prototype enhanced retrospection is effective in selecting the most representative samples from diverse dialogues.

(2) In addition, we compare our method with previous data selection methods. For "+ iCaRL" (Rebuffi et al., 2017), the model computes only one prototype for each domain based on the hidden state of the [CLS] token and selects samples based on this prototype. For "+ K-Means" (Han et al., 2020), this model selects diverse samples by choosing the central samples of clusters in the [CLS] hidden vector space. KPN significantly outperforms "+ iCaRL" and "+ K-Means". "+ iCaRL" is even worse than the random selection "- MPR" because the [CLS] prototype is often not representative for any slot. By contrast, our method adopts multiple prototypes based on the slot representation, which effectively selects the most representative samples.

4.5.2 Effectiveness of Multi-Strategy Knowledge Distillation

To gain more insights into the multi-strategy knowledge distillation, we test many variants of KPN. The results are shown in Table 4. We can see that:

(1) Removing any knowledge distillation strategy, encoder feature distillation (EFD), classifier prediction distillation (CPD), decoder feature distillation (DFD), or generation prediction distillation (GPD), brings performance degradation. If we remove all knowledge distillation strategies (MSKD), the performance further declines. It shows that these knowledge distillation strategies can retain performance on old domains by effectively preserving the knowledge of the original model from multiple perspectives.

(2) When we remove both multi-prototype enhanced retrospection and multi-strategy knowledge distillation (i.e., the model EMR), the performance drops significantly. It indicates that simultaneously exploiting the two components is very effective.

Number	KPN (Ours)		EMAR	
	Avg	Whole	Avg	Whole
50	**43.19**	**40.14**	**39.06**	**35.89**
40	40.47	37.10	37.82	32.93
30	39.65	35.68	32.83	26.31
20	37.94	34.71	29.37	23.05
10	35.05	29.73	28.64	21.16

Table 5: Effect of the number of reserved samples. We compare our method KPN with EMAR on the MultiWOZ benchmark.

4.6 Discussion: Memory Capacity

As shown in Table 5, we test the models that reserve different numbers of samples. Both EMAR and our method KPN achieve performance improvements as the number of reserved samples increases. In each case, our method significantly outperforms EMAR. Our method using only 30 samples achieves comparable performance to EMAR using 50 samples. It demonstrates the effectiveness of our proposed method. The proposed multi-prototype enhanced retrospection effectively selects the most representative samples. The proposed multi-strategy knowledge distillation alleviates the impact of combinatorial explosion.

5 Related Work

5.1 Dialogue State Tracking

Dialogue state tracking (DST) is an active research area recently, where typical DST models can be mainly divided into two categories: discriminative DST methods and generative DST methods. Discriminative DST methods use predefined values as categories to simplify DST as a multi-class classification task (Bohus and Rudnicky, 2006; Williams et al., 2013; Henderson et al., 2014a). These methods mainly focus on modeling the relation between slots and dialogue history, such as NBT (Mrkšić et al., 2017), GLAD (Zhong et al., 2018), SST (Chen et al., 2020), and CHAN (Shan et al., 2020). Generative DST methods treat dialogue state tracking as a generation task (Rastogi et al., 2017; Xu and Hu, 2018; Wu et al., 2019). By generating values from the dialogue history and the vocabulary, generative DST methods handle unknown values that are not predefined in the ontology. Therefore, generative DST methods dominate this research, such as SpanPtr (Xu and Hu, 2018), COMER (Ren et al., 2019), BERT-DST (Chao and Lane, 2019), TRADE (Wu et al., 2019), SAS (Hu et al., 2020), and SOM-DST (Kim et al., 2020).

Despite the great progress in single-domain or multi-domain DST tasks, previous DST methods usually assume the training data is fixed, containing predefined domains. They can not learn newly emerging domains online, which makes it impractical in real-world applications. Our method handles the domain-lifelong learning problem, where data of new domains continually arrives, whether it is new single-domain or new multi-domain data.

5.2 Lifelong Learning

Lifelong learning, also called continual learning, is a long-standing research topic in machine learning, which enables models to perform online learning on new data (Cauwenberghs and Poggio, 2000; Kuzborskij et al., 2013). Architecture-based methods dynamically extend the model architecture to learn new data (Fernando et al., 2017; Shen et al., 2019). However, the model size grows rapidly with the increase of new data, which limits the application of architecture-based methods. Existing lifelong learning methods can be divided into two main categories: regularization-based methods (Zenke et al., 2017; Aljundi et al., 2018) and replay-based methods (Rebuffi et al., 2017; Hou et al., 2019). Regularization-based methods design reasonable metrics to identify the parameters important to old data and slow down the update of them (Kirkpatrick et al., 2017; Li and Hoiem, 2017). Replay-based methods retain the previous knowledge by storing a small amount of old data (Wang et al., 2019; Han et al., 2020). In addition, generative replay-based methods generate old data to alleviate catastrophic forgetting (Shin et al., 2017; Kemker and Kanan, 2018; Ostapenko et al., 2019; Zhai et al., 2019). Although lifelong learning has been widely investigated in NLP and CV scenarios (Kou et al., 2020; Kundu et al., 2020), its exploration in DST is relatively rare.

In other dialogue tasks, Lee (2017) fine-tunes a dialogue model trained on open-domain dialogues to learn task-oriented dialogues. However, their setting is only a one-step incremental process. Shen et al. (2019) continually train a slot-filling model on new data from the same domain. Madotto et al. (2020) introduce continual learning into multiple dialogue tasks. However, they ignore cross-domain dialogues that exist widely in the real world. In addition, they only adopt a plain architecture-based method, which does not address the main challenges of the dialogue tasks.

In contrast to previous work, we formally introduce domain-lifelong learning into DST, which is practical in real-world applications. In addition, we propose Knowledge Preservation Networks to handle the main challenges of the DLL-DST task.

6 Conclusion

In this paper, we introduce domain-lifelong learning into dialogue state tracking and propose Knowledge Preservation Networks to overcome catastrophic forgetting. To handle expression diversity, we propose multi-prototype enhanced retrospection to reserve the most representative samples. Moreover, to alleviate the adverse effects of combinatorial explosion, we propose multi-strategy knowledge distillation to learn the previous knowledge of the original model. Experimental results on the MultiWOZ and SGD benchmarks demonstrate the effectiveness of our model.

Acknowledgements

The authors wish to thank the anonymous reviewers for their valuable comments and helpful suggestions which greatly improved the paper quality. The work is supported by the National Key Research and Development Program of China (2020AAA0106400) and the National Natural Science Foundation of China (61922085, 61976211). The work is also supported by the Beijing Academy of Artificial Intelligence (BAAI2019QN0301), the Key Research Program of the Chinese Academy of Sciences under Grant (ZDBS-SSW-JSC006), the independent research project of the National Laboratory of Pattern Recognition, China and the Youth Innovation Promotion Association CAS, China.

References

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. 2018. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 139–154.

Dan Bohus and Alex Rudnicky. 2006. A k hypotheses + other belief updating model. In *AAAI Workshop on Statistical and Empirical Approaches to Spoken Dialogue Systems, 2006, Boston, MA*.

Pengfei Cao, Yubo Chen, Jun Zhao, and Taifeng Wang. 2020. Incremental event detection via knowledge consolidation networks. In *Proceedings of EMNLP*, pages 707–717.

Gert Cauwenberghs and Tomaso Poggio. 2000. Incremental and decremental support vector machine learning. *Advances in neural information processing systems*, 13:409–415.

Guan-Lin Chao and Ian Lane. 2019. Bert-dst: Scalable end-to-end dialogue state tracking with bidirectional encoder representations from transformer. *Proceedings of Interspeech 2019*, pages 1468–1472.

Lu Chen, Boer Lv, Chi Wang, Su Zhu, Bowen Tan, and Kai Yu. 2020. Schema-guided multi-domain dialogue state tracking with graph attention neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7521–7528.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of EMNLP*, pages 1724–1734.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT, Volume 1*, pages 4171–4186.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyag Gao, and Dilek Hakkani-Tur. 2019. Multiwoz 2.1: Multi-domain dialogue state corrections and state tracking baselines. *arXiv preprint arXiv:1907.01669*.

Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A. Rusu, Alexander Pritzel, and Daan Wierstra. 2017. Pathnet: Evolution channels gradient descent in super neural networks. *CoRR*, abs/1701.08734.

Robert M French. 1999. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135.

Silin Gao, Yichi Zhang, Zhijian Ou, and Zhou Yu. 2020. Paraphrase augmented task-oriented dialog generation. In *Proceedings of ACL*, pages 639–649.

Xu Han, Yi Dai, Tianyu Gao, Yankai Lin, Zhiyuan Liu, Peng Li, Maosong Sun, and Jie Zhou. 2020. Continual relation learning via episodic memory activation and reconsolidation. In *Proceedings of the 58th ACL*, pages 6429–6440.

Matthew Henderson, Blaise Thomson, and Jason D. Williams. 2014a. The second dialog state tracking challenge. In *Proceedings of the 15th SIGDIAL*, pages 263–272.

Matthew Henderson, Blaise Thomson, and Steve Young. 2014b. Word-based dialog state tracking with recurrent neural networks. In *Proceedings of SIGDIAL*, pages 292–299.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. 2019. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 831–839.

Jiaying Hu, Yan Yang, Chencai Chen, Liang He, and Zhou Yu. 2020. SAS: Dialogue state tracking via slot attention and slot information sharing. In *Proceedings of the 58th ACL*, pages 6366–6375.

Ronald Kemker and Christopher Kanan. 2018. Fearnet: Brain-inspired model for incremental learning. In *Proceedings of the 6th ICLR*.

Sungdong Kim, Sohee Yang, Gyuwan Kim, and Sang-Woo Lee. 2020. Efficient dialogue state tracking by selectively overwriting memory. In *Proceedings of the 58th ACL*, pages 567–582.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Xiaoyu Kou, Yankai Lin, Shaobo Liu, Peng Li, Jie Zhou, and Yan Zhang. 2020. Disentangle-based Continual Graph Representation Learning. In *Proceedings of the 2020 Conference on EMNLP*, pages 2961–2972.

Jogendra Nath Kundu, Rahul Mysore Venkatesh, Naveen Venkat, Ambareesh Revanur, and R Venkatesh Babu. 2020. Class-incremental domain adaptation. *arXiv preprint arXiv:2008.01389*.

Ilja Kuzborskij, Francesco Orabona, and Barbara Caputo. 2013. From n to n+ 1: Multiclass transfer incremental learning. In *Proceedings of the IEEE Conference on CVPR*, pages 3358–3365.

Sungjin Lee. 2017. Toward continual learning for conversational agents. *arXiv preprint arXiv:1712.09943*.

Wenqiang Lei, Xisen Jin, Min-Yen Kan, Zhaochun Ren, Xiangnan He, and Dawei Yin. 2018. Sequicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In *Proceedings of ACL*, pages 1437–1447.

Zhizhong Li and Derek Hoiem. 2017. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947.

David Lopez-Paz and Marc'Aurelio Ranzato. 2017. Gradient episodic memory for continual learning. In *Advances in neural information processing systems*, pages 6467–6476.

Andrea Madotto, Zhaojiang Lin, Zhenpeng Zhou, Seungwhan Moon, Paul Crook, Bing Liu, Zhou Yu, Eunjoon Cho, and Zhiguang Wang. 2020. Continual learning in task-oriented dialogue systems. *arXiv preprint arXiv:2012.15504*.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. 2017. Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of ACL*, pages 1777–1788.

Oleksiy Ostapenko, Mihai Marian Puscas, Tassilo Klein, Patrick Jähnichen, and Moin Nabi. 2019. Learning to remember: A synaptic plasticity driven framework for continual learning. In *Proceedings of the CVPR 2019*, pages 11321–11329.

Yawen Ouyang, Moxin Chen, Xinyu Dai, Yinggong Zhao, Shujian Huang, and Jiajun Chen. 2020. Dialogue state tracking with explicit slot connection modeling. In *Proceedings of the 58th ACL*, pages 34–40.

Abhinav Rastogi, Dilek Hakkani-Tür, and Larry P. Heck. 2017. Scalable multi-domain dialogue state tracking. In *2017 IEEE ASRU Workshop*, pages 561–568.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2019. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. *arXiv preprint arXiv:1909.05855*.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. 2017. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010.

Liliang Ren, Jianmo Ni, and Julian McAuley. 2019. Scalable and accurate dialogue state tracking via hierarchical sequence generation. In *Proceedings of EMNLP-IJCNLP*, pages 1876–1885.

Yong Shan, Zekang Li, Jinchao Zhang, Fandong Meng, Yang Feng, Cheng Niu, and Jie Zhou. 2020. A contextual hierarchical attention network with adaptive objective for dialogue state tracking. In *Proceedings of ACL*, pages 6322–6333.

Yilin Shen, Xiangyu Zeng, and Hongxia Jin. 2019. A progressive model to enable continual learning for semantic slot filling. In *Proceedings of the 2019 Conference on EMNLP-IJCNLP*, pages 1279–1284.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. 2017. Continual learning with deep generative replay. In *Proceedings of the NeurIPS 2017*, pages 2990–2999.

Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pages 4077–4087.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Hong Wang, Wenhan Xiong, Mo Yu, Xiaoxiao Guo, Shiyu Chang, and William Yang Wang. 2019. Sentence embedding alignment for lifelong relation extraction. In *Proceedings of the 2019 Conference of the NAACL-HLT*, pages 796–806.

Jason Williams, Antoine Raux, Deepak Ramachandran, and Alan Black. 2013. The dialog state tracking challenge. In *Proceedings of SIGDIAL*, pages 404–413.

Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. 2019. Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th ACL*, pages 808–819.

Puyang Xu and Qi Hu. 2018. An end-to-end approach for handling unknown slot values in dialogue state tracking. In *Proceedings of the 56th ACL*, pages 1448–1457.

Hong-Ming Yang, Xu-Yao Zhang, Fei Yin, and Cheng-Lin Liu. 2018. Robust classification with convolutional prototype learning. In *Proceedings of the IEEE Conference on CVPR*, pages 3474–3482.

Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of IEEE*, 101(5):1160–1179.

Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual learning through synaptic intelligence. *Proceedings of machine learning research*, 70:3987.

Mengyao Zhai, Lei Chen, Frederick Tung, Jiawei He, Megha Nawhal, and Greg Mori. 2019. Lifelong GAN: continual learning for conditional image generation. In *Proceedings of the ICCV 2019*, pages 2759–2768.

Victor Zhong, Caiming Xiong, and Richard Socher. 2018. Global-locally self-attentive encoder for dialogue state tracking. In *Proceedings of ACL*, pages 1458–1467.