TaSL: Continual Dialog State Tracking via Task Skill Localization and Consolidation

Yujie Feng[1], **Xu Chu**[2,3,4], **Yongxin Xu**[2,3], **Guangyuan Shi**[1], **Bo Liu**[1], **Xiao-Ming Wu**[1]*
[1]Department of Computing, The Hong Kong Polytechnic University, Hong Kong S.A.R.
[2]School of Computer Science, Peking University, Beijing, China
[3]Key Laboratory of High Confidence Software Technologies, Ministry of Education, Beijing, China
[4]Center on Frontiers of Computing Studies, Peking University, Beijing, China
yujie.feng@connect.polyu.hk, xiao-ming.wu@polyu.edu.hk

Abstract

A practical dialogue system requires the capacity for ongoing skill acquisition and adaptability to new tasks while preserving prior knowledge. However, current methods for Continual Dialogue State Tracking (DST), a crucial function of dialogue systems, struggle with the catastrophic forgetting issue and knowledge transfer between tasks. We present TaSL, a novel framework for task skill localization and consolidation that enables effective knowledge transfer without relying on memory replay. TaSL uses a novel group-wise technique to pinpoint task-specific and task-shared areas. Additionally, a fine-grained skill consolidation strategy protects task-specific knowledge from being forgotten while updating shared knowledge for bi-directional knowledge transfer. As a result, TaSL strikes a balance between preserving previous knowledge and excelling at new tasks. Comprehensive experiments on various backbones highlight the significant performance improvements of TaSL over existing state-of-the-art methods. The source code[1] is provided for reproducibility.

1 Introduction

With the rising popularity of conversational digital assistants, it is imperative for dialogue systems to integrate new services while sustaining proficiency in prior tasks seamlessly. Traditional research, often conducted within specific domains offline, falls short in adaptability to new scenarios (Ni et al., 2023). Retraining pre-trained language models (PLMs) from scratch is both challenging and resource-intensive (Liu et al., 2023), highlighting the necessity for efficient continual learning (CL) approaches in dialogue systems (Ke and Liu, 2022). Dialogue state tracking (DST), crucial for task-oriented dialogue systems, dynamically updates

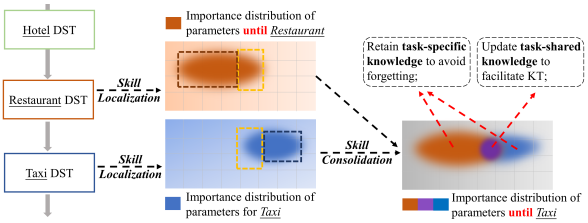

Figure 1: Conceptual illustration of TaSL. By identifying task-relevant areas across both previously accumulated and current tasks, we can consolidate the task-specific and task-shared parameters to facilitate efficient knowledge transfer and mitigate forgetting.

(domain, slot, value) triplets to capture user intentions precisely. The urgent demand for advancing DST models to accommodate emerging services has catalyzed the development of the Continual DST task (Cho et al., 2023).

An effective Continual DST system must address the issue of catastrophic forgetting (McCloskey and Cohen, 1989), where a model's proficiency in old tasks diminishes after learning new ones. It should also promote knowledge transfer (KT) (Ke et al., 2021) across domains[2] to enhance end-task performances. Knowledge transfer includes forward transfer, which improves new task performance using knowledge from previous tasks, and backward transfer, which enhances performance on previous tasks after learning a new relevant task. Striking a balance between retaining previous knowledge and excelling in new tasks is vital for success.

However, current Continual DST methods (Madotto et al., 2020; Liu et al., 2021; Cho et al., 2023; Feng et al., 2024) mainly focus on mitigating forgetting through memory replay or regularization, overlooking the advantages of KT that can be derived from the inherent correlations between different DST domains.

Task correlation in Continual DST is quite ev-

*Corresponding author.
[1]https://github.com/WoodScene/TaSL

[2]In this work, different *domains* in Continual DST are equivalent to different *tasks* in CL.

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1266–1279

ident. For instance, domains like "Hotel" and "Restaurant" share semantically similar slots, such as "area" and "bookday", highlighting the need for models to identify and handle common information types. The similarity in these domain-shared slots is crucial for enabling KT. However, learning domain-specific slots like "food" for "Restaurant" could introduce unique information that disrupts the retention of previously acquired knowledge, leading to catastrophic forgetting.

To address these challenges, we introduce **Ta**sk **S**kill **L**ocalization and Consolidation (TaSL), a framework designed to improve KT between tasks without relying on memory replay. This is achieved by identifying and consolidating the importance distribution of model parameters across tasks. TaSL initially employs a group-wise importance-aware ***skill localization*** technique that utilizes gradient trajectories to pinpoint tiny regions in the model that store crucial knowledge for the current task. By comparing the importance distribution with those of previous tasks, we can differentiate between task-specific and task-shared parameters, as illustrated in Figure 1. Our innovative ***skill consolidation*** phase then categorically integrates weights from previous tasks with the current one, enabling effective KT while minimizing forgetting.

In detail, the importance-aware skill localization method employs a new group-wise metric to compute importance scores, effectively quantifying the significance of each "***skill unit***"[3] for the current task. Our approach, focusing on parameter space rather than dataset-driven categorization of domain-shared and domain-specific slots, offers a more robust solution that accurately identifies task-specific and task-shared knowledge, overcoming inaccuracies caused by dataset noise.

Our skill consolidation stage, then based on a fine-grained model averaging strategy, effectively manages different types of knowledge. We enable forward KT to new tasks by starting with a model initialized with weights from previously fine-tuned tasks, thus using past knowledge to improve learning for new tasks without restrictions. For backward KT, we merge knowledge from both current and past accumulated tasks into localized task-shared skill units, thereby enhancing their capability. To prevent catastrophic forgetting, we consolidate the integrity of skill units containing previous task-specific knowledge, ensuring they remain unaffected by new task learning. Through extensive experiments on different parameter-level backbones (from 60M to 7B), TaSL exhibits superior performance in mitigating forgetting and showcases remarkable capabilities for KT, significantly outperforming state-of-the-art methods.

Our main contributions include:

- We propose a novel task skill localization and consolidation (TaSL) **framework** for CL.
- We develop new group-wise skill localization and fine-grained skill consolidation **techniques**.
- Extensive **evaluation** on continual DST tasks shows TaSL effectively enables knowledge transfer, resulting in a 3.1% absolute increase in Avg. JGA and an 8.8% absolute boost in BWT metrics compared to previous SOTA methods.

2 Related Work

2.1 Continual Dialogue State Tracking

Continual Learning (CL) in task-oriented dialogue systems focuses on perpetually integrating knowledge from data streams for future application. Three kinds of CL methods have been developed. Architecture-based methods propose dynamically adding model weights when learning new data (Geng et al., 2021; Lu et al., 2021b; Yang et al., 2023). Replay-based methods store and replay some training samples from previous tasks (Hou et al., 2019; Lu et al., 2021a; Xu et al., 2023a). Regularization-based methods employ additional loss functions to solidify new knowledge (Li and Hoiem, 2017; Xu et al., 2023b).

In the realm of Continual DST, pioneering efforts by Madotto et al. (2020) and Liu et al. (2021) have leveraged these CL strategies to set benchmark performance using PLMs. The DST-EGQA approach by Cho et al. (2023) reformulates the DST task to an example-guided question-answering task, aiming to align distribution shifts across different domains to mitigate forgetting. However, these methods overlook DST task correlations that could enhance knowledge transfer. The recent Continual Prompt Tuning (CPT) method by Zhu et al. (2022) attempts knowledge transfer via domain-specific soft prompts but depends on inefficient memory replay and extensive retraining. This dataset-driven approach is inefficient and lacks robustness.

Our TaSL innovates by distinguishing between domain-specific and domain-shared knowledge

[3]The definition of skill units is model-specific. For example, the *Query* matrix within the self-attention layer may be regarded as a skill unit. For detailed definitions, please refer to Appendix B.

within the parameter space, then leveraging the skill consolidation process for effective knowledge transfer and forgetting mitigation.

2.2 Task Skill Localization

Research indicates that model parameters contribute unevenly to performance (Michel et al., 2019). Panigrahi et al. (2023) introduced the concept of "skill localization" to identify crucial parameters within PLMs, suggesting that fine-tuning critical parameters nearly matches the effect of full fine-tuning. However, their method requires additional time for identifying and retraining key parameters post-fine-tuning, lowering efficiency.

Drawing inspiration from the pruning community, previous studies have used gradient-based metrics to identify important parameters during fine-tuning. Sensitivity-based scoring (Sanh et al., 2020; Zhang et al., 2022b) assesses the impact on training loss and sensitivity smoothing, as applied by Zhang et al. (2022a), eliminates unnecessary parameters for more efficient fine-tuning. However, these approaches, focusing on individual parameter importance, often lead to element-wise pruning with huge computational and storage burdens (Feng et al., 2023a). Based on these advances, we introduce a new importance-aware skill localization method, for the first time, that distinguishes between task-specific and shared parameters to mitigate forgetting in CL.

3 Proposed Method: TaSL

Problem Formulation In continual DST, we aim to train a model $f : \mathcal{X} \times \mathcal{T} \rightarrow \mathcal{Y}$ across a sequence of dialogue domains $\mathcal{T}_1, \ldots, \mathcal{T}_K$. Each dialogue domain has its dataset $\mathcal{D}_k$ for $\mathcal{T}_k$. This model predicts the target y based on input x and task $\mathcal{T}_k \in \mathcal{T}$. The notation f_k refers to the model after training on task $\mathcal{T}_k$, while $\hat{f}_k$ denotes the model after averaging for $\hat{f}_{k-1}$ and f_k. Within a given task $\mathcal{T}_k$, a dialogue with M turns of interaction between the system and the user is denoted as $\mathcal{X}_M = \{(A_1, U_1), (A_2, U_2) \ldots, (A_M, U_M)\}$, where A and U represent the system's response and the user's input, respectively.

Each task $\mathcal{T}_k$ is associated with a predefined slot set[4] $\mathcal{S} = \{S_1, \ldots, S_J\}$, where J is the total number of slots. The objective of DST is to predict the dialogue state $\mathcal{B}_m$ based on the dialogue context $\mathcal{X}_m$. The dialogue state is a collection of (slot, value) pairs, expressed as $\mathcal{B}_m = \{(S_1, V_1^m), \ldots, (S_J, V_J^m)\}$, where V_J^m is the value for slot S_J at turn m. DST involves training a model $f : \mathcal{X}_m \oplus S_j \rightarrow V_j^m$, with $\oplus$ denotes simple text concatenation.

Overview TaSL includes two key components: (i) ***Skill Localization***, utilizing a new group-wise importance metric to accurately identify the importance distribution of parameters across tasks, and (ii) ***Skill Consolidation***, which employs a novel fine-grained model averaging strategy to integrate model weights from both current and past tasks for effective knowledge transfer. Figure 2 provides a comprehensive overview of TaSL, with the following subsections detailing each component.

3.1 Importance-aware Skill Localization

To address the substantial computational and storage demands imposed by previous parameter-level importance calculation methods (Konishi et al., 2023), we propose a new group-wise metric for evaluating the importance of each skill unit u:

$$\mathcal{I}(u) = \frac{1}{d_1 \times d_2} \sum_{i=1}^{d_1} \sum_{j=1}^{d_2} s(w_{ij}) \tag{1}$$

where w_{ij} denotes the trainable parameters, and $d_1 \times d_2$ represents the total parameter count in a skill unit u. $\mathcal{I}(u)$ measures the collective importance of all parameters within each skill unit, where higher values signify increased importance. The function $s(\cdot)$ is a designated importance function for individual parameters, defined as the magnitude of the gradient-weight product:

$$I\left(w_{ij}\right) = \left| w_{ij} \nabla_{w_{ij}} \mathcal{L} \right| \tag{2}$$

This approximates the loss change when a parameter is zeroed out. If removing a parameter has a significant influence, then the model is sensitive to it, and we should retain it (Liang et al., 2021).

However, sensitivity in Eq. (2) may not reliably indicate importance (Zhang et al., 2022b). This metric, calculated from a sampled mini-batch, suffers from variability due to stochastic sampling and training dynamics, introducing large uncertainty in estimating sensitivity. To mitigate this, we apply sensitivity smoothing and uncertainty quantification (Zhang et al., 2022a):

$$\bar{I}^{(t)}\left(w_{ij}\right) = \alpha_1 \bar{I}^{(t-1)}\left(w_{ij}\right) + \left(1 - \alpha_1\right) I^{(t)}\left(w_{ij}\right) \tag{3}$$

[4]To specify the domain to which a slot belongs, a slot is defined as the concatenation of the specific domain and the slot name, e.g., "<restaurant-area>".

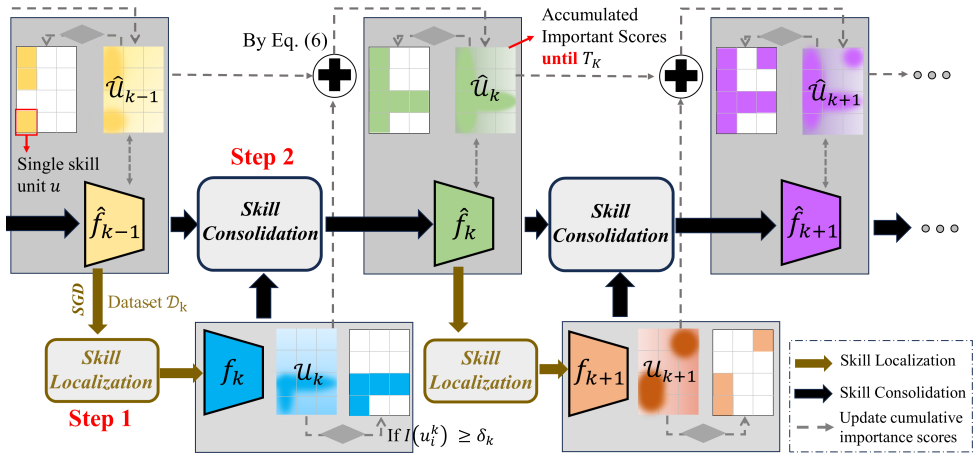

Figure 2: **Overview of TaSL. Step 1**: We compute the importance scores of skill units for the current task $\mathcal{T}_k$ using our importance-aware skill localization method during fine-tuning. **Step 2**: Based on a fine-grained model averaging strategy, the skill consolidation method merges the model $\hat{f}_{k-1}$, which accumulates knowledge of all previous tasks, with the current task's model f_k. The integration is guided by the importance distributions of skill units across various tasks. We then update the cumulative importance scores for all skill units until task $\mathcal{T}_k$ using Eq. (6). This process is designed to be iteratively repeated with the introduction of each subsequent task.

$$\bar{U}^{(t)}(w_{ij}) = \alpha_2 \bar{U}^{(t-1)}(w_{ij}) + (1-\alpha_2)\left|I^{(t)}(w_{ij}) - \bar{I}^{(t)}(w_{ij})\right| \tag{4}$$

where α_1 and α_2 are smoothing factors, and t is the iteration number. $\bar{I}^{(t)}$ represents smoothed sensitivity by exponential moving average and $\bar{U}^{(t)}$ is the uncertainty term quantified by the local variation between $I^{(t)}$ and $\bar{I}^{(t)}$. Importance is then defined by multiplying $\bar{I}^{(t)}$ and $\bar{U}^{(t)}$, providing a more accurate importance assessment for $s(\cdot)$:

$$s^{(t)}(w_{ij}) = \bar{I}^{(t)}(w_{ij}) \cdot \bar{U}^{(t)}(w_{ij}) \tag{5}$$

Calculating current task importance scores. To compute the importance score of each skill unit for task $\mathcal{T}_k$, we employ Eq. (1) during fine-tuning. The model f with n skill units is denoted as $\mathcal{U} = \{u_1, \ldots, u_n\}$, with their importance scores for task $\mathcal{T}_k$ denoted by $\mathcal{I}(\mathcal{U}_k) \in \mathbb{R}^n$. The detailed computation process is provided in Algorithm 1.

Computing accumulated importance scores for previous tasks. After computing importance scores for each skill unit at the current task $\mathcal{T}_k$, it is essential to compare these with scores from all previously learned tasks to distinguish between task-specific and task-shared parameters. To avoid the inefficiency of storing scores for each past task, we aggregate importance scores from all prior tasks into a cumulative score for tasks up to $\mathcal{T}_{k-1}$. This method allows for the iterative refinement of accumulated scores without separately saving past task scores. The skill units with these cumulative scores up to $\mathcal{T}_{k-1}$ are denoted as $\hat{\mathcal{U}}_{k-1}$, calculated using:

$$\mathcal{I}(\hat{\mathcal{U}}_{k-1}) = \beta \text{Norm}(\mathcal{I}(\hat{\mathcal{U}}_{k-2})) + (1-\beta)\text{Norm}(\mathcal{I}(\mathcal{U}_{k-1})) \tag{6}$$

where $\beta \in [0, 1]$, and Norm$(\cdot)$ normalizes importance scores to the [0, 1] range, thus resolving discrepancies across models. The initial scores, $\mathcal{I}(\hat{\mathcal{U}}_1)$, are set to be equal to $\mathcal{I}(\mathcal{U}_1)$. Following this, the importance distribution for skill units up to task $\mathcal{T}_{k-1}$ is combined with that of the current task, $\mathcal{T}_k$, to facilitate the skill consolidation process.

3.2 Skill Consolidation

After skill localization, the subsequent vital phase involves consolidating this knowledge into a unified framework. This process demands a sophisticated model averaging approach considering various factors to optimize task performance. Traditional coarse-grained model averaging assumes that all model weights are equally important for the training task (Kirkpatrick et al., 2017; Eddine Marouf et al., 2023), which can be written as the following iterative computation format:

$$\hat{f}_k = \lambda \hat{f}_{k-1} + (1-\lambda) f_k \tag{7}$$

However, this method may overemphasize weights irrelevant to the current task, contaminating previously acquired task-specific knowledge and leading to forgetting. To counteract this, we introduce a fine-grained averaging strategy focusing

Algorithm 1 Importance-aware Skill Localization

Input: Training dataset $\mathcal{D}_k$ for task $\mathcal{T}_k$; total training iterations T; hyperparameters α_1, α_2.
  for $t = 1, \ldots, T$ **do**
    Sample a mini-batch from $\mathcal{D}_k$ and compute the gradient $\nabla\mathcal{L}$;
    Compute the sensitivity $I\left(w_{ij}\right)$ via Eq. (2);
    Update $\bar{I}^{(t)}$ via Eq. (3) and $\bar{U}^{(t)}$ via Eq. (4);
  end for
  Compute the importance score $\mathcal{I}(u_i^k)$ for each skill unit u_i^k by Eq. (1), for $i = 1, \ldots, n$.
Output: f_k and importance scores $\mathcal{I}(\mathcal{U}_k)$ for $\mathcal{U}_k$.

on skill units rather than the entire model. Our approach distinguishes between task-shared and task-specific skill units, categorically applying weighted averaging to parameters within each skill unit.

We initially set importance thresholds δ using quantiles to select the top 20% of skill units based on importance scores. A skill unit u_i^k is deemed important (denoted as $(u_i^k)^+$) if its score $\mathcal{I}(u_i^k)$ is above δ_k, and unimportant ($(u_i^k)^-$) otherwise.

Our fine-grained averaging strategy customizes parameter combination for each skill unit, based on its importance under different tasks, as follows:

$$\hat{u}_i^k = \begin{cases} \gamma\hat{u}_i^{k-1} + (1-\gamma)u_i^k, & \text{if } (\hat{u}_i^{k-1})^+, (u_i^k)^+ \\ \hat{u}_i^{k-1}, & \text{if } (\hat{u}_i^{k-1})^+, (u_i^k)^- \\ u_i^k, & \text{if } (\hat{u}_i^{k-1})^-, (u_i^k)^+ \\ \frac{1}{2}(\hat{u}_i^{k-1} + u_i^k), & \text{if } (\hat{u}_i^{k-1})^-, (u_i^k)^- \end{cases} \tag{8}$$

This strategy performs the element-wise adjustment of parameters within each skill unit based on its relevance to previous and current tasks, using hyperparameter γ to control their influences.

In the scenario where a skill unit u_i is significant for both past and present tasks (case 1), we integrate newly acquired knowledge into this task-shared skill unit to enable backward KT. If a skill unit u_i is crucial solely for previous tasks (case 2), we maintain the knowledge within this previous task-specific skill unit untouched to prevent the contamination of historical knowledge with task-irrelevant information. In contrast, for a skill unit important only to the current task (case 3), since the model f_k is trained on the initialization of $\hat{f}_{k-1}$, the historically learned knowledge is utilized to enhance the performance of the current task, enabling forward KT. Thus, we ensure the integrity of parameters within this current task-specific skill unit, preserving essential knowledge for excelling in the new task. We adopt a straightforward averaging for units not pertinent to either task (case 4).

Skill consolidation is performed before starting a new task in CL, utilizing the averaged model for subsequent task initialization. Only the importance scores of $\hat{\mathcal{U}}_{k-1}$ and $\mathcal{U}_k$ are retained for use between tasks, starting with $\hat{\mathcal{U}}_1 = \mathcal{U}_1$ estimated from f_1 on D_1. Detailed implementation of TaSL algorithm can be found in the Appendix (Algorithm 2).

4 Experiments and Analysis

Dataset We use the continual learning for DST setup proposed by Zhu et al. (2022), which uses 15 single domains from the Schema-Guided Dialog dataset (SGD) (Rastogi et al., 2020). We aggregate our results over the same five domain orders to make the most reliable comparisons with prior works. Comparing results with the same order is crucial as the results can have significant variance depending on the chosen domains and their order. More details about data statistics, task selection, and orderings can be found in the Appendix A.

Evaluation Protocol We evaluate DST performance using the widely adopted Joint Goal Accuracy (JGA) metric (Wu et al., 2019), which indicates the percentage of turns for which all slot values are correctly predicted. We denote $a_{j,i}$ as the JGA on the test set of task $\mathcal{T}_i$ right after training on task $\mathcal{T}_j$. The performance of Continual DST is assessed using three metrics from Zhu et al. (2022):

$$\textbf{Avg. JGA} = \frac{1}{K}\sum_{i=1}^{K} a_{K,i} \tag{9}$$

$$\textbf{FWT} = \frac{1}{K-1}\sum_{i=2}^{K} a_{i-1,i} \tag{10}$$

$$\textbf{BWT} = \frac{1}{K-1}\sum_{i=1}^{K-1} a_{K,i} - a_{i,i} \tag{11}$$

Avg. JGA represents the average JGA across all tasks after training on the final task $\mathcal{T}_K$. Forward Transfer (FWT) evaluates a model's generalization ability by measuring the averaged zero-shot performance. Backward Transfer (BWT) assesses the impact of learning on subsequent tasks on a previous task. Negative BWT indicates the model lost some previously acquired knowledge.

Method	Memory-Free	Avg. JGA	FWT	BWT
Fine-tuning (Madotto et al., 2020)	✓	$44.1_{0.9}$	$8.3_{1.0}$	$-36.6_{3.9}$
EWC (Kirkpatrick et al., 2017)		$47.9_{1.1}$	$8.4_{0.9}$	$-38.1_{4.1}$
AdapterCL (Madotto et al., 2020)		$49.8_{1.7}$	-	-
DST-EGQA (Cho et al., 2023)		$55.5_{3.5}$	$23.6_{2.1}$	$-19.1_{4.2}$
RoS (Feng et al., 2024)		$59.0_{3.9}$	$25.5_{2.0}$	$-17.9_{3.7}$
TaSL (ours)		$\mathbf{62.1}_{2.0}$	$\mathbf{26.6}_{1.5}$	$\mathbf{-9.1}_{2.2}$
Replay (Madotto et al., 2020)	✗	$58.6_{3.5}$	$10.9_{0.5}$	$-3.2_{2.3}$
CPT (Zhu et al., 2022)		$61.2_{2.5}$	$13.7_{0.8}$	$0.5_{0.4}$
DST-EGQA (Cho et al., 2023)		$68.9_{0.3}$	$22.5_{1.8}$	$-5.9_{1.9}$
RoS (Feng et al., 2024)		$72.1_{0.8}$	$26.7_{2.0}$	$-2.6_{1.5}$
CPT Multi-task (Zhu et al., 2022)	-	$64.0_{1.9}$	-	-
DST-EGQA Multi-task (Cho et al., 2023)		$74.2_{1.8}$	-	-
RoS Multi-task (Cho et al., 2023)		$76.3_{0.3}$	-	-

Table 1: CL results of various methods, all utilizing the same T5-small backbone, on 15 different tasks from the SGD dataset. Means and standard variances are reported across five domain permutations. The last two rows provide the multi-tasking results, which serve as an upper bound. Our memory replay-free TaSL outperforms the previous best method, RoS, by achieving a 3.1% absolute improvement on avg. JGA and an 8.8% absolute increase in BWT. Additionally, TaSL exceeds the performance of the majority of memory replay methods and nearly matches the upper bound of the CPT multi-task method.

Baselines We evaluate our method with the following Continual DST baselines: ***Fine-tuning:*** Continuously fine-tune the backbone model on new task data. ***Replay:*** Randomly save $|\mathcal{M}|$ samples from the training set of each previous task $\mathcal{T}_i$ in memory $\mathcal{M}_i$ and jointly train the model on new task data $\mathcal{D}_K$ and memory $\mathcal{M}_{<k}$. ***EWC:*** Maintain a memory but leverage it to compute the Fisher information matrix for regularization (Kirkpatrick et al., 2017). ***AdapterCL:*** Freeze the pre-trained model and independently train a residual Adapter (Houlsby et al., 2019) for each task (Madotto et al., 2020). ***Continual Prompt Tuning (CPT) (Zhu et al., 2022):*** Freeze the backbone model and continually train soft prompts with memory-guided knowledge transfer in both forward and backward directions. ***DST-EGQA (Cho et al., 2023):*** Reformulate DST as a QA task to mitigate forgetting with retrieval-augmented in-context learning. ***RoS (Feng et al., 2024):*** Utilize knowledge distillation to enhance the meta-reasoning ability of student models, thereby mitigating forgetting.

Training Details We utilize four distinct parameter-level backbones for experiments: T5-small (Raffel et al., 2020), T5-base, Flan-T5-large (Chung et al., 2022), and LLaMA-7B (Touvron et al., 2023). For the T5 series model, we perform full fine-tuning across all parameters. For LLaMA-7B, we adopt the Parameter-Efficient Fine-Tuning technique, specifically Low-Rank Adaptation (LoRA) (Hu et al., 2021), to expedite the training process. For TaSL, we set the hyperparameters α_1 and α_2 in Eq. (3) and Eq. (4) to 0.85, and set β in Eq. (6) and γ in Eq. (8) to 0.7. The memory size per task $|\mathcal{M}|$ is maintained at 50, aligning with previous studies. Detailed training settings are provided in Appendix B.

Following this, we compare TaSL with baselines in Sec. 4.1, and present a comprehensive ablation study in Sec. 4.2. Subsequently, we delve deeper into the underlying success of our proposed importance-aware skill localization (Sec. 4.3) and skill consolidation techniques (Sec. 4.4), and get some insightful findings from this exploration.

4.1 Main Results

Overall CL results of different methods at the same T5-small backbone are summarized in Table 1.

TaSL demonstrates superior CL performance through effective knowledge transfer. Unlike vanilla fine-tuning, which suffers from catastrophic forgetting, our approach demonstrates a substantial improvement in Avg. JGA, increasing it from 44.1% to 62.1%, and shows marked advancements in both FWT and BWT.

TaSL not only exceeds the CPT method, which relies on memory replay, advancing the Avg. JGA from 61.2% to 62.1%, but also establishes a new

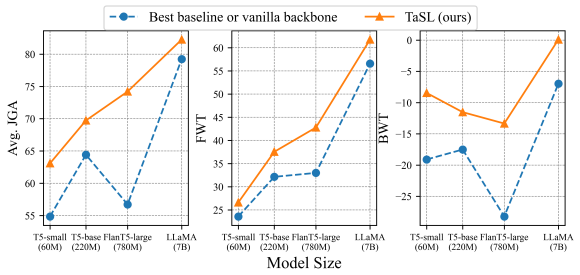

Figure 3: Performance of TaSL w/ different backbones.

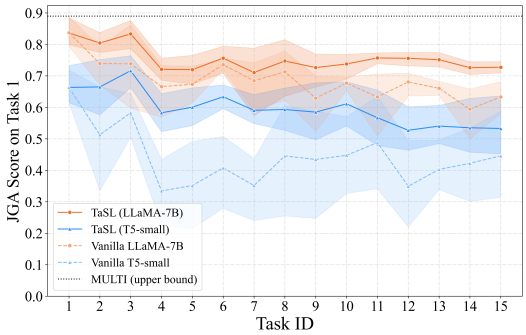

Figure 4: Performance trajectory of Task 1 during the Continual DST learning process.

SOTA across all metrics against the top baseline, DST-EGQA. We achieve an impressive increase in Avg. JGA from 59.0% to 62.1% (3.1% absolute improvement) and elevate BWT from -17.9% to -9.1%, exceeding it by more than 8% and displaying robust backward KT capabilities. Additionally, TaSL obtains the highest FWT scores across all baselines under various conditions.

Remarkably, without relying on memory replay, our method nearly matches the performance of DST-EGQA with memory replay, particularly in BWT, with a minimal difference of 3.2% (-9.1% vs. -5.9%). Moreover, our Avg. JGA score closely approaches the upper bound performance set by the CPT multi-task strategy at 64%, underscoring the effectiveness of our fine-grained model averaging strategy that meticulously accounts for domain-shared and domain-specific parameters.

TaSL consistently demonstrates superior performance across various backbones. To further substantiate the effectiveness of our framework, we conducted experiments using a variety of parameter-level backbones, illustrated in Figure 3, highlighting performance gains with increasing model size. Notably, TaSL achieves a breakthrough by recording a positive BWT score on LLaMA-7B, without employing any memory replay techniques. Across different backbones, our method consistently outperformed traditional approaches. For instance, in Flan-T5-large, TaSL significantly boosts the Avg. JGA metric from 56% to 74%, also achieving the most substantial improvements in both FWT and BWT metrics — rising from 33% to 43% and improving from -28% to -13%, respectively. These results further validate the generality of our proposed framework.

Fine-grained model averaging can effectively mitigate catastrophic forgetting. To rigorously evaluate our model's effectiveness in counteracting forgetting, we analyzed its performance on the initial task after training on subsequent tasks. Figure 4 illustrates that our method results in a notably slower forgetting rate, manifesting as a nearly 8% average decrease in performance after training on the last task. This contrasts sharply with vanilla backbones, which display a substantial performance reduction of 20% on average, thereby underscoring our method's superior capacity to mitigate forgetting. Moreover, an intriguing observation is the enhancement in performance on task 1 after training on task 3, highlighting our model's effective backward KT ability.

4.2 Ablation Study

This section we assess the effects of importance-aware skill localization and fine-grained skill consolidation, with the results discussed below. For hyperparameter sensitivity, see Appendix D.

Various importance scoring methods for skill localization. Our method calculates importance scores by Eq. (1). As shown in Table 2, we explore alternative importance scoring approaches: (i) modifying $s(\cdot)$ in Eq. (1) only to include sensitivity, as in Eq. (2); and (ii) using absolute gradients, $\left|\nabla_{w_{ij}} \mathcal{L}\right|$, for importance assessment (Michel et al., 2019). The results indicate that using moving averages for importance scoring outperforms the alternatives, with the other two variants leading to a maximum performance decrease by 3.26%, 2.24%, and 4.11% across the three metrics. This highlights the value of accurate skill localization in improving model performance.

Fine-grained vs. coarse-grained model averaging for skill consolidation. We compared our fine-grained averaging strategy against two coarse-grained strategies: (i) Weight-Ensemble, which averages weights uniformly as per Eq. (7), and

Method	Avg. JGA	FWT	BWT
vanilla T5-small	44.10	8.32	-36.63
$s(\cdot) = I(\cdot)$	60.48	24.39	-10.81
$s(\cdot) = \left\|\nabla_{w_{ij}} \mathcal{L}\right\|$	58.82	24.80	-13.22
TaSL (ours)	**62.08**	**26.63**	**-9.11**

Table 2: Ablation study. Evaluating the impact of importance scoring variations on skill localization.

Method	Avg. JGA	FWT	BWT
vanilla T5-small	44.10	8.32	-36.63
Weight-Ens.	53.23	21.73	-18.28
EMA	52.56	22.27	-16.80
Fine-grained (ours)	**62.08**	**26.63**	**-9.11**

Table 3: Ablation study. Comparing coarse- and fine-grained model averaging methods on skill consolidation.

(ii) Exponential Moving Average (EMA) (Szegedy et al., 2016), applying a running average of parameters at each fine-tuning iteration. Results are detailed in Table 3.

Weight-Ensemble significantly improves upon the vanilla model, highlighting coarse-grained averaging's benefits for Continual DST. EMA generally surpasses Weight-Ensemble but falls short of our fine-grained approach due to its overuse of averaging, with frequent parameter adjustments within the same task possibly resulting in less optimal outcomes. Our method solely averages weights after each task, enhancing computational efficiency.

4.3 Visualization of Skill Units

We visualized the distribution of importance scores for the skill units across tasks and models, as shown in Figure 5, leading to several critical insights:

- There is a noticeable variation in the importance of skill units for the same task, with important skill units in LLaMA-7B making up about 20% of all trainable LoRA parameters.
- The distribution of important skill units is task-dependent, indicating both task-shared and specific parameters, confirming TaSL's validity.
- Lower layers, nearer to the input, are more crucial for the DST task compared to upper layers.
- Within each layer, the importance of the attention layer, especially the V (value) and O (output) matrices, consistently exceeds that of the Q (query), K (key) matrices, and the MLP layer.

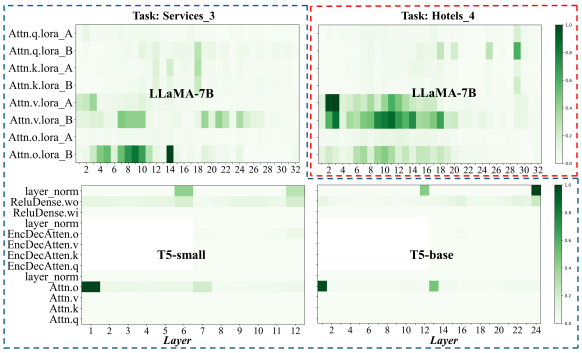

Figure 5: Visualization of importance scores for skill units across different backbone models and tasks.

Seq. length	Method	Task1	Task2	Task3
2 (LLaMA-7B)	Upper bound	92.3	86.1	-
	Fine-tuning	73.1	86.1	-
	Coarse-grained	82.3	71.3	-
	TaSL (ours)	**86.9**	**83.3**	-
3 (T5-small)	Upper bound	89.2	81.5	64.4
	Fine-tuning	53.1	62.0	64.4
	Coarse-grained	58.5	64.6	43.7
	TaSL (ours)	**80.0**	**74.1**	**63.8**

Table 4: Analysis of knowledge balancing across old and new tasks. All results are reported in JGA(%).

4.4 Improved Balance in Knowledge Transfer

This section evaluates the effectiveness of our fine-grained model averaging method in achieving the optimal balance between preserving previous knowledge and excelling at new tasks in CL, comparing it to coarse-grained approaches.

Table 4 shows that for a sequence of two tasks, vanilla fine-tuning on LLaMA-7B results in a notable decline in historical Task 1's performance (from 92.3% to 73.1%), indicating notable forgetting. Coarse-grained averaging mitigates this to an extent, reducing the decline to 82.3% but impacting new task performance to 71.3%. Our method effectively lessens forgetting (improving to 86.9%) while also maintaining better performance on Task 2, with less than a 3% reduction.

As tasks increase to three, our method more effectively compensates for losses on new tasks with gains on historical tasks. Vanilla fine-tuning on T5-small results in a combined 55.6% drop in Tasks 1 and 2, while our approach only shows a 16.6% decrease and the loss on task 3 is less than 1% due to TaSL's effective KT ability.

5 Conclusion

In this paper, we introduce a novel TaSL method to enhance Continual DST performance by facilitating effective knowledge transfer across tasks. Our approach leverages an innovative importance-aware skill localization technique and a skill consolidation strategy to differentiate between domain-specific and domain-shared parameters, mitigating forgetting. Comprehensive experiments showcase our method's exceptional ability to balance preserving past knowledge and excelling in new tasks.

Limitations

TaSL excels at precisely distinguishing between task-specific and shared parameters through importance-aware skill localization. However, the current importance scoring criteria, relying on first-order gradients, may lack precision. The Hessian matrix often captures the actual importance, but computing these second-order gradients incurs significant computational costs. Therefore, future improvements should focus on developing more accurate and efficient skill localization methods.

In addition, in skill consolidation, the challenge lies in better integrating model parameters. Under our fine-grained model averaging strategy (Eq. (8)), selecting different weighted combinations could impact the overall performance. Although we investigated the model's sensitivity to various hyperparameter settings (Appendix D), with results showing stable and consistently strong performance across different combinations, there is still potential for further improvement. For instance, developing adaptive methods to select optimal weights or devising more efficient model averaging strategies could further enhance model performance.

Acknowledgments

We thank the anonymous reviewers for their valuable feedback. This research was partially supported by the grant of HK ITF ITS/359/21FP.

References

Hyundong Cho, Andrea Madotto, Zhaojiang Lin, Khyathi Raghavi Chandu, Satwik Kottur, Jing Xu, Jonathan May, and Chinnadhurai Sankar. 2023. Continual dialogue state tracking via example-guided question answering. *arXiv preprint arXiv:2305.13721*.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. 2022. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416*.

Imad Eddine Marouf, Subhankar Roy, Enzo Tartaglione, and Stéphane Lathuilière. 2023. Weighted ensemble models are strong continual learners. *arXiv e-prints*, pages arXiv–2312.

Yujie Feng, Bo Liu, Xiaoyu Dong, Zexin Lu, Li-Ming Zhan, Xiao-Ming Wu, and Albert YS Lam. 2024. Continual dialogue state tracking via reason-of-select distillation. In *Findings of the Association for Computational Linguistics: ACL 2024*.

Yujie Feng, Zexin Lu, Bo Liu, Liming Zhan, and Xiao-Ming Wu. 2023a. Towards llm-driven dialogue state tracking. *arXiv preprint arXiv:2310.14970*.

Yujie Feng, Jiangtao Wang, Yasha Wang, and Xu Chu. 2022. Spatial-attention and demographic-augmented generative adversarial imputation network for population health data reconstruction. *IEEE Transactions on Big Data*.

Yujie Feng, Jiangtao Wang, Yasha Wang, and Xu Chu. 2023b. Towards sustainable compressive population health: a gan-based year-by-year imputation method. *ACM Transactions on Computing for Healthcare*, 4(1):1–18.

Yujie Feng, Jiangtao Wang, Yasha Wang, and Sumi Helal. 2021. Completing missing prevalence rates for multiple chronic diseases by jointly leveraging both intra-and inter-disease population health data correlations. In *Proceedings of the Web Conference 2021*, pages 183–193.

Binzong Geng, Fajie Yuan, Qiancheng Xu, Ying Shen, Ruifeng Xu, and Min Yang. 2021. Continual learning for task-oriented dialogue system with iterative network pruning, expanding and masking. *arXiv preprint arXiv:2107.08173*.

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/CVF conference on computer vision and pattern recognition*, pages 831–839.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Zixuan Ke and Bing Liu. 2022. Continual learning of natural language processing tasks: A survey. *arXiv preprint arXiv:2211.12701*.

Zixuan Ke, Bing Liu, Nianzu Ma, Hu Xu, and Lei Shu. 2021. Achieving forgetting prevention and knowledge transfer in continual learning. *Advances in Neural Information Processing Systems*, 34:22443–22456.

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.

Tatsuya Konishi, Mori Kurokawa, Chihiro Ono, Zixuan Ke, Gyuhak Kim, and Bing Liu. 2023. Parameter-Level Soft-Masking for Continual Learning. In *Proc. of ICML*.

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

Chen Liang, Simiao Zuo, Minshuo Chen, Haoming Jiang, Xiaodong Liu, Pengcheng He, Tuo Zhao, and Weizhu Chen. 2021. Super tickets in pre-trained language models: From model compression to improving generalization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6524–6538.

Bo Liu, Liming Zhan, Zexin Lu, Yujie Feng, Lei Xue, and Xiao-Ming Wu. 2023. How good are large language models at out-of-distribution detection? *arXiv preprint arXiv:2308.10261*.

Qingbin Liu, Pengfei Cao, Cao Liu, Jiansong Chen, Xunliang Cai, Fan Yang, Shizhu He, Kang Liu, and Jun Zhao. 2021. Domain-lifelong learning for dialogue state tracking via knowledge preservation networks. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2301–2311.

Zexin Lu, Keyang Ding, Yuji Zhang, Jing Li, Baolin Peng, and Lemao Liu. 2021a. Engage the public: Poll question generation for social media posts. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 29–40.

Zexin Lu, Jing Li, Yingyi Zhang, and Haisong Zhang. 2021b. Getting your conversation on track: Estimation of residual life for conversations. In *2021 IEEE Spoken Language Technology Workshop (SLT)*, pages 1036–1043. IEEE.

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. Elsevier.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? *Advances in neural information processing systems*, 32.

Jinjie Ni, Tom Young, Vlad Pandelea, Fuzhao Xue, and Erik Cambria. 2023. Recent advances in deep learning based dialogue systems: A systematic survey. *Artificial intelligence review*, 56(4):3055–3155.

Abhishek Panigrahi, Nikunj Saunshi, Haoyu Zhao, and Sanjeev Arora. 2023. Task-specific skill localization in fine-tuned language models. *arXiv preprint arXiv:2302.06600*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551.

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8689–8696.

Victor Sanh, Thomas Wolf, and Alexander M Rush. 2020. Movement pruning: adaptive sparsity by fine-tuning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 20378–20389.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

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 Annual Meeting of the Association for Computational Linguistics*, pages 808–819.

Yongxin Xu, Xu Chu, Kai Yang, Zhiyuan Wang, Peinie Zou, Hongxin Ding, Junfeng Zhao, Yasha Wang, and Bing Xie. 2023a. Seqcare: Sequential training with external medical knowledge graph for diagnosis prediction in healthcare data. In *Proceedings of the ACM Web Conference 2023*, pages 2819–2830.

Yongxin Xu, Kai Yang, Chaohe Zhang, Peinie Zou, Zhiyuan Wang, Hongxin Ding, Junfeng Zhao, Yasha Wang, and Bing Xie. 2023b. Vecocare: visit sequences-clinical notes joint learning for diagnosis prediction in healthcare data. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI-23*, pages 4921–4929.

Kai Yang, Yongxin Xu, Peinie Zou, Hongxin Ding, Junfeng Zhao, Yasha Wang, and Bing Xie. 2023. Kerprint: local-global knowledge graph enhanced diagnosis prediction for retrospective and prospective interpretations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 5357–5365.

Qingru Zhang, Minshuo Chen, Alexander Bukharin, Pengcheng He, Yu Cheng, Weizhu Chen, and Tuo Zhao. 2022a. Adaptive budget allocation for parameter-efficient fine-tuning. In *The Eleventh International Conference on Learning Representations*.

Qingru Zhang, Simiao Zuo, Chen Liang, Alexander Bukharin, Pengcheng He, Weizhu Chen, and Tuo Zhao. 2022b. Platon: Pruning large transformer models with upper confidence bound of weight importance. In *International Conference on Machine Learning*, pages 26809–26823. PMLR.

Qi Zhu, Bing Li, Fei Mi, Xiaoyan Zhu, and Minlie Huang. 2022. Continual prompt tuning for dialog state tracking. *arXiv preprint arXiv:2203.06654*.

A Dataset Statistics

Here, we offer a detailed description of the dataset used in Continual DST. Table 5 displays the number of slots for each of the 15 services used in our experiments and the count of samples in the training, validation, and test sets. Table 6 illustrates the training sequence for these 15 tasks in the context of continual learning.

B Implementation Details

Definition of Skill Units In our Importance-aware Skill Localization technique, to circumvent the high computational resource demands of parameter-level localization, we introduced a novel group-wise metric, redefining the skill unit u as the basic element for computing importance scores. However, the division of skill units varies across different backbone models due to variations in parameter quantities and architectural designs (e.g., decoder-only and encoder-decoder architectures). The specific distinctions are as follows:

- **Encoder-decoder architecture backbones.** For these backbones (Feng et al., 2021), such as T5-small, T5-base, and T5-large, we perform full-parameter fine-tuning during training. For organizational simplicity, we divided the models based on the different functionalities within the transformer blocks, as depicted in Table 7. For instance, in T5-small, with both encoder and decoder comprising 6 transformer blocks each, the total comes to 131 skill units for T5-small, 257 skill units for T5-base, and 558 for T5-large.
- **Decoder-only architecture backbone.** The LLaMA-7B model we utilized falls into this category (Feng et al., 2023b). Leveraging Parameter-efficient Fine-tuning Techniques (PEFT) to expedite training, we treat the matrices A and B in LoRA as individual basic skill units. And we add LoRA adapters to attention layers in LLaMA. Each layer, as detailed in Table 8, comprises 8 distinct skill units. Given that LLaMA-7B consists of 32 layers, it is thereby segmented into 256 skill units in total.

Model training details For different backbones, we utilized the following hyperparameters:

- **T5-small (60M)**, **T5-base (220M)**, and **FLAN-T5-lare (780M)**: Training was conducted with a learning rate of 3e-4, batch size of 8, maximum input length of 512, maximum target length of 128, and 5 epochs.
- **LLaMA (7B)**: Utilizing LORA for efficiency, with a learning rate of 3e-4, batch size of 128, a cutoff length of 512, and 5 epochs. Lora settings were r = 8, alpha = 16, dropout = 0.05, targeting modules [[q_proj,k_proj,v_proj,o_proj]]. For testing, settings included temperature = 0.02, top_p = 0, top_k = 1, num_beams = 1, max new tokens = 128.

Experiments are carried out using 2 NVIDIA A100 with 80GB memory. Results are averaged across five different task orders and include the standard error in the tables and plots provided (Feng et al., 2022).

C Additional Results

To further validate TaSL's effectiveness in more complex continual learning scenarios, we have conducted additional experiments to verify its performance on transitioning between different datasets,

Task ID	Service	# Slots	# Dialogs			# Samples			Avg. tokens	
			Train	*Dev*	*Test*	*Train*	*Dev*	*Test*	*Context*	*Query*
30	services_4	5	86	13	25	680	97	208	154	49
31	flights_1	10	560	80	160	4680	667	1379	168	10
32	services_3	5	131	19	38	959	143	290	143	54
33	flights_3	8	65	10	19	420	75	116	133	79
34	trains_1	7	58	9	17	415	67	117	131	76
35	homes_2	8	62	9	18	424	56	139	140	89
36	rentalcars_2	6	77	11	23	631	91	185	157	61
37	restaurants_1	9	256	37	74	2098	297	581	153	10
38	music_1	6	68	10	20	468	73	142	118	61
39	hotels_4	7	80	12	23	559	99	141	134	72
40	media_2	5	32	4	10	215	29	71	112	59
41	hotels_3	6	90	13	26	737	100	193	157	64
42	rentalcars_3	7	44	7	13	332	55	99	148	72
43	hotels_1	7	99	14	29	868	105	250	161	71
44	homes_1	7	244	35	70	1829	282	540	159	81

Table 5: Statistics of the 15 services we used in experiments.

Task order	Tasks' IDs in order														
Order1	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44
Order2	39	33	36	42	40	37	38	34	32	35	41	31	30	44	43
Order3	30	41	38	31	43	39	40	33	34	44	37	36	32	35	42
Order4	43	40	44	38	30	37	31	39	32	35	41	34	33	36	42
Order5	30	33	44	31	38	32	42	40	37	43	36	39	41	35	34

Table 6: Five task orders of all our 15 tasks experiments.

specifically from SGD to MultiWoz. This involved including another 5 distinct domains from the MultiWoz 2.1 dataset, in addition to the 15 domains in SGD, resulting in a total of 20 domains (i.e., tasks). Table 9 presents the performance of various methods, utilizing T5-small as the backbone.

The findings align with the observations from Table 1, although there is a noticeable decrease in the efficacy of all evaluated methods, due to the significant discrepancies in data distribution across the datasets examined. As a memory-free method, our TaSL still significantly outperforms the strongest baseline (i.e., the memory-free version of DST-EGQA) and even surpasses some memory-based methods like "Replay". These findings demonstrate TaSL's robustness and effectiveness, showcasing its capability to handle complex continual learning scenarios.

D Sensitivity Analysis for Hyperparameters

The proposed framework incorporates three key hyperparameters, including the α for computing importance scores in Equations (3) and (4), the β for calculating cumulative importance scores in Equation (6), and the γ for performing weighted averaging within skill units as outlined in Equation (8). Our analysis aims to assess the impact of varying these hyperparameters on our method's performance, testing on the T5-small backbone model.

As evidenced in Table 10, we determine that the optimal setting for α is 0.55. An α value too low results in a performance decline, indicating that the calculated importance scores are not sufficiently accurate. Furthermore, as depicted in the results of Tables 11 and 12, we also find that β and γ values within a normal range do not significantly affect performance. However, excessively high or low values for β and γ may skew the model towards favoring either past or current task knowledge, thereby disrupting the desired balance.

Block Type	Skill Unit Name
Encoder	SelfAttention.q.weight SelfAttention.k.weight SelfAttention.v.weight SelfAttention.0.weight layer.0.layer_norm.weight DenseReluDense.wi.weight DenseReluDense.wo.weight layer.1.layer_norm.weight
Decoder	SelfAttention.q.weight SelfAttention.k.weight SelfAttention.v.weight SelfAttention.0.weight SelfAttention.relative_attention_bias.weight layer.0.layer_norm.weight layer.1.EncDecAttention.q.weight layer.1.EncDecAttention.k.weight layer.1.EncDecAttention.v.weight layer.1.EncDecAttention.o.weight 1.layer_norm.weight

Table 7: Definition of skill units for encoder-decoder architecture backbones at each transformer block.

Block Type	Skill Unit Name
Decoder	self_attn.q_proj.lora_A.default.weight self_attn.q_proj.lora_B.default.weight self_attn.k_proj.lora_A.default.weight self_attn.k_proj.lora_B.default.weight self_attn.v_proj.lora_A.default.weight self_attn.v_proj.lora_B.default.weight self_attn.o_proj.lora_A.default.weight self_attn.o_proj.lora_B.default.weight

Table 8: Definition of skill units for decoder-only architecture backbones at each transformer block.

Nonetheless, the model's performance remains relatively stable across most conditions, indicating a low sensitivity to hyperparameter variations.

About the selection of threshold for important skill units, the Table 13 below shows the model's performance with varying thresholds δ on T5-small.

It can be seen that setting a high threshold (50%) reduces model effectiveness by categorizing less significant skill units as important, which can contaminate historical knowledge and lead to forgetting. Conversely, a 1% threshold still maintains strong performance owing to our effective skill consolidation approach, which effectively preserves task-specific knowledge and prevents forgetting. Considering that the heatmap in Figure 5 displays approximately 20% of the area in darker shades, signifying greater importance, we opted for a 20% threshold to differentiate between important and unimportant skill units.

Method	Memory-Free	Avg. JGA	FWT	BWT
Fine-tuning		20.1	6.6	-53.1
DST-EGQA	✓	40.5	18.4	-37.1
TaSL (ours)		**49.9**	**22.0**	**-23.8**
Replay	✗	47.2	7.3	-16.0
DST-EGQA		51.2	18.5	-21.9

Table 9: Cross-dataset performance of TaSL.

α_1,α_2	avg. JGA	FWT	BWT
fine-tuning	41.6	9.6	-36.7
0.15	61.8	29.7	-10.7
0.35	61.2	30.1	-12.3
0.55	62.8	28.6	-9.5
0.85	60.7	28.9	-10.3
0.95	61.7	30.0	-10.6

Table 10: Performance comparisons of TaSL (using T5-small as the backbone) equipped with different α at task order 1.

E Detailed Algorithm

In this section, we provide the detailed implementation of TaSL algorithm (see Algorithm 2).

β	avg. JGA	FWT	BWT
fine-tuning	41.6	9.6	-36.7
0.1	61.8	29.6	-11.7
0.3	61.5	28.4	-11.4
0.5	62.3	29.5	-10.2
0.7	60.7	28.9	-10.3
0.9	58.2	30.2	-13.0

Table 11: Performance comparisons of TaSL (using T5-small as the backbone) equipped with different β at task order 1.

γ	avg. JGA	FWT	BWT
fine-tuning	41.6	9.6	-36.7
0.1	60.1	28.2	-12.1
0.3	61.7	28.8	-11.2
0.5	63.0	28.4	-10.5
0.7	60.7	28.9	-10.3
0.9	61.7	27.4	-11.5

Table 12: Performance comparisons of TaSL (using T5-small as the backbone) equipped with different γ at task order 1.

Importance Thresholds δ	Avg. JGA	FWT	BWT
1%	62.0	26.3	-9.4
5%	63.4	25.8	-10.1
10%	62.2	26.2	-9.5
20%	62.1	26.6	-9.1
30%	62.7	26.5	-10.0
40%	60.9	24.6	-10.2
50%	54.8	23.4	-10.3

Table 13: Performance comparisons of TaSL (using T5-small as the backbone) equipped with different δ.

Algorithm 2 TaSL

Input: Dataset $\mathcal{D}_k$ for task $k = 1, \ldots, K$; initial pre-trained model f_0; hyperparameters β, γ.

 # sequential tasks.
for task $k = 1, \ldots, K$ **do**
  Get f_k and calculate $\mathcal{U}_k$ by Algorithm (1);
  Calculate δ_k based on $\mathcal{I}(\mathcal{U}_k)$;
  if $k = 1$ **then**
    # initialization at beginning task.
    $\hat{f}_1 \leftarrow f_1, \hat{\mathcal{U}}_1 \leftarrow \mathcal{U}_1, \hat{\delta}_1 \leftarrow \delta_1$;
  else
    # fine-grained model averaging.
    for skill unit $i = 1, \ldots, n$ **do**
      Calculate $\hat{u}_i^k$ by (8);
    end for
    Get the averaged model $\hat{f}_k$ based on $\hat{\mathcal{U}}_k$;
    Calculate accumulated importance score $\mathcal{I}(\hat{\mathcal{U}}_k)$ according to (6);
    Calculate $\hat{\delta}_k$ based on $\mathcal{I}(\hat{\mathcal{U}}_k)$.
  end if
end for