# MeLL: Large-scale Extensible User Intent Classification for Dialogue Systems with Meta Lifelong Learning

Chengyu Wang[1*], Haojie Pan[1*], Yuan Liu[1], Kehan Chen[1], Minghui Qiu[1†], Wei Zhou[1], Jun Huang[1], Haiqing Chen[1], Wei Lin[1], Deng Cai[2]
{chengyu.wcy,haojie.phj,xuanjing.ly,kehan.ckh,minghui.qmh,fayi.zw,huangjun.hj}@alibaba-inc.com
{haiqing.chenhq,weilin.lw}@alibaba-inc.com,dengcai@gmail.com
[1] Alibaba Group, China [2] State Key Lab of CAD & CG, Zhejiang University, China

## ABSTRACT

User intent detection is vital for understanding their demands in dialogue systems. Although the User Intent Classification (UIC) task has been widely studied, for large-scale industrial applications, the task is still challenging. This is because user inputs in distinct domains may have different text distributions and target intent sets. When the underlying application evolves, new UIC tasks continuously emerge in a large quantity. Hence, it is crucial to develop a framework for *large-scale extensible UIC* that continuously fits new tasks and avoids catastrophic forgetting with an acceptable parameter growth rate. In this paper, we introduce the **Me**ta **L**ifelong **L**earning (MeLL) framework to address this task. In MeLL, a BERT-based text encoder is employed to learn robust text representations across tasks, which is slowly updated for lifelong learning. We design global and local memory networks to capture the cross-task prototype representations of different classes, treated as the meta-learner quickly adapted to different tasks. Additionally, the Least Recently Used replacement policy is applied to manage the global memory such that the model size does not explode through time. Finally, each UIC task has its own task-specific output layer, with the attentive summarization of various features. We have conducted extensive experiments on both open-source and real industry datasets. Results show that MeLL improves the performance compared with strong baselines and also reduces the number of total parameters. We have also deployed MeLL on a real-world e-commerce dialogue system AliMe and observed significant improvements in terms of both F1 and the resources usage.

## CCS CONCEPTS

• **Information systems → Query intent**; • **Computing methodologies → Lifelong machine learning**.

[*] Chengyu Wang and Haojie Pan contributed equally to this work.
[†] Minghui Qiu is the corresponding author.


*KDD '21, August 14–18, 2021, Virtual Event, Singapore*

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467107

## KEYWORDS

user intent classification, lifelong learning, meta-learning, pre-trained language model, dialogue systems

**ACM Reference Format:**
C. Wang, H. Pan, Y. Liu, K. Chen, M. Qiu, W. Zhou, J. Huang, H. Chen, W. Lin, D. Cai. 2021. MeLL: Large-scale Extensible User Intent Classification for Dialogue Systems with Meta Lifelong Learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3447548.3467107

## 1 INTRODUCTION

Dialogue systems are human-machine interaction systems that give responses to users' queries or execute commands based on the natural language conversations between the two parties. Due to their usefulness, they are often integrated into a variety of industrial-scale applications, such as customer service in e-commerce [50], restaurant booking applications [15], and conversational information seeking [9]. To fulfill users' needs, it is highly essential to understand users' intents based on the input queries issued by users or users' responses to actions previously taken by the systems.

In the literature, the task of *User Intent Classification* (UIC) has been extensively studied, mostly addressed by learning text-to-label classifiers [13, 29, 42]. Recently, the emergence of pre-trained language models [11, 38, 48] makes it natural to fine-tune these models to obtain such classifiers. For real-world applications, however, solving the UIC task is non-trivial as UIC is highly domain-specific where each domain may have different input text distributions and class label sets (i.e., all the possible intents). For example, the e-commerce hotline service in Alime [1] has over two hundred service domains such as retail, trip agents and express delivery. For each domain, the customized hotline agent is required to perform a number of UIC tasks that automatically classify users' responses into a pre-defined set of intents on different questions. The total number of UIC tasks would easily exceed one thousand. Additionally, the task number is continuously growing through time as new domains gradually appear. Hence, it is crucial to develop a *large-scale **E**xtensible **U**ser **I**ntent **C**lassification* (EUIC) framework used in dialogue systems.

To address the problem, a naive approach is to train task-wise UIC models. This type of methods is unsuitable for industrial-scale applications with three reasons. i) The total number of model parameters is continuously growing, linearly proportional to the number

[1] https://www.alixiaomi.com/

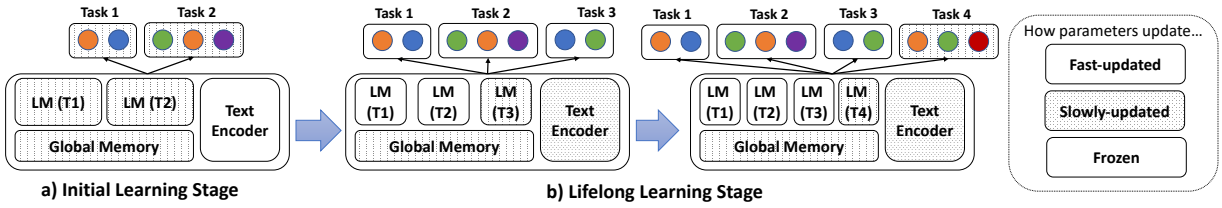



**Figure 1: The high-level framework MeLL for large-scale EUIC tasks in dialogue systems. "LM (T$n$)" refers to the local memory network for the $n$-th UIC task. (Best viewed in color.)**

of tasks. Given the fact that current pre-trained language models have billions of parameters [11, 38], training such models for UIC would easily involve trillions of parameters, leading to the *parameter explosion* problem. ii) As UIC tasks across domains share some similarities, the single-task approach is unable to learn transferable knowledge from other tasks, which is crucial to improve the UIC performance [35]. iii) The methods unavoidably bring engineering burdens when an increasing number of models need to be maintained. Yet another popular approach is multi-task training across tasks, where a shared encoder is utilized to capture the common knowledge and each task has its own prediction head [27, 44]. When a novel UIC task emerges, we may need to re-train the model for previous tasks, which is computationally expensive and difficult to keep the performance stable for existing UIC tasks.

Recently, *lifelong learning* has been attracted attention from the research community. It is a learning paradigm that continuously accumulates knowledge learned in the past and uses it to help future task learning [2, 33]. When lifelong learning is applied to large-scale EUIC, we only need to maintain a relatively small number of model parameters when new tasks continuously arrive, alleviating the *parameter explosion* effect. In lifelong learning, it is challenging to address the *catastrophic forgetting* problem [46] where the model "forgets" how to solve existing tasks when it learns new tasks. This is particularly undesirable as we wish to keep the performance of existing tasks stable when we learn new UIC tasks. Another similar paradigm is *meta-learning* [17], aiming to obtain a *meta-learner* across tasks such that it can quickly fit new tasks with few data samples. By obtaining a *meta-learner*, the transferable knowledge across different UIC tasks can be acquired and passed to new tasks. The major drawback is that the *meta-learner* (the BERT model [11] in our case) should adapt to each task separately, unable to avoid *parameter explosion*. Hence, a natural question arises: *is it possible to design a continual learning framework for large-scale EUIC that can both i) maintain the performance of existing UIC tasks when the model fits new UIC tasks and ii) have an acceptable parameter growth rate when the number of new UIC tasks increases?*

In this work, we introduce the ***Me**ta **L**ifelong **L**earning* (MeLL) framework for large-scale EUIC, with the high-level framework illustrated in Figure 1. It has a shared network architecture for learning a continuously growing number of UIC tasks, consisting of three parts: the text encoder, the global memory network and local memory networks. The text encoder is built based on BERT [11] to generate robust text representations (either for users' queries or responses). The parameters are slowly updated for lifelong representation learning, which ensures that the update operations invoked by new tasks do not have significantly negative effects on existing tasks. Inspired by prototype-based meta-learning [43], the global network stores the cross-task prototype representations of different classes. The memory units are fast updated, capturing the transferable knowledge across tasks and making our model fit new tasks easily. As the number of distinct classes of all tasks is continuously growing, we use the LRU (Least Recently Used) replacement policy to manage the global memory such that the size does not explode through time. We fuse the features generated by the text encoder and the global memory network by the attention mechanism, and use them to learn the final task-specific UIC classifiers. After the learning process is finished, we copy the task-related prototype representations into the task's own local memory network, with parameters frozen. During the inference time, we use the text encoder and the task's own local memory network for feature generation.

From the architecture design, we can see that MeLL successfully accomplishes the goal of large-scale EUIC. The encoder is regarded as the *slow learner* that continuously digests the transferable representation learning knowledge and passes it to specific tasks. The global memory network is the *fast learner*, capable of encoding specific knowledge of the given task quickly. The LRU replacement policy and the copy mechanism from the global memory network to local ones alleviate *catastrophic forgetting*, without adding too many parameters to the model. [2]

In summary, we make the following major contributions:

- We introduce the task of *large-scale EUIC*, which is vital for understanding users' intents in dialogue systems with a large, increasing number of UIC tasks involved.
- We propose the *Meta Lifelong Learning* (MeLL) framework to address this task. It employs a slowly updated text encoder to learn representations across tasks and global/local memory networks to learn task semantics. It enables effective knowledge transfer through time and alleviates catastrophic forgetting and parameter explosion at the same time.
- We conduct extensive experiments on public and real-world industry datasets. The results show that MeLL consistently outperforms strong baselines.
- We deploy MeLL on a real-world dialogue system AliMe and observe significant improvements in an online A/B test.

The rest of this paper is summarized as follows. Section 2 summarizes the related work. Section 3 introduces the MeLL framework in detail. In Section 4, we evaluate the performance of MeLL in

[2] Although we focus on UIC tasks in this paper, the MeLL framework is general and can be applied to other tasks, which is beyond the scope of this paper.

various aspects, specifically focusing on the industrial deployment. Finally, we summarize our paper in Section 5.

## 2 RELATED WORK

**User Intent Classification.** The techniques of UIC are originally applied to information seeking in search engines, which help the search engines to understand the search queries sent by users [1, 21]. As dialogue systems usually provide better user experience by the interaction between the systems and users, UIC for dialogue systems gains more popularity. For example, Qu et al. [37] present the MSDialog dataset, which is annotated with user query intents in question answering systems. Different from classification-based approaches, Zhang et al. [49] characterize users' intents by generating textual descriptions. UIC can be also formulated as a ranking problem. In [47], the Intent-Aware Ranking with Transformers (IART) model is proposed to select suitable answers considering query intents based on attention mechanisms. MeLL differentiates itself from these approaches in that it considers solving a large number of UIC tasks in the lifelong learning setting, which is crucial for industrial applications.

**Lifelong Learning.** Lifelong learning, or continual learning is a machine learning paradigm which focuses on solving an unlimited sequence of tasks with the help of previously learned tasks [2, 33]. A key challenge of developing lifelong learning algorithms is to improve the performance of future tasks while avoiding the catastrophic forgetting of existing tasks [18, 46]. Typical methods include experience replay [16, 20, 40], knowledge distillation [6, 26], transfer learning [5, 19, 39], etc. In real-world, industrial applications, it is costly to apply experience replay to a large number of history tasks or store these trained models for knowledge distillation. In MeLL, we employ both slow and fast learner (i.e., the text encoder and the global memory network) to transfer knowledge from existing tasks to new tasks.

**Meta-learning.** The goal of meta-learning is to train meta-learners that can adapt to a variety of tasks with little training data available [17]. Meta-learning is extensively employed in computer vision, formulated as a K-way N-shot few-shot learning problem. Typical applications include few-shot image classification [25], objection detection [12] and many others. The applications of meta-learning in NLP have not been frequently studied, with works such as [31, 32, 34]. Compared to previous works, the MeLL framework is not a typical K-way N-shot algorithm, but leverages the idea of meta-learning to learn the text encoder that captures the transferable knowledge across tasks. The fast update mechanism and the prototype representations used in the global memory are similar to several meta-learning neural networks [30, 43].

**Pre-trained Language Models.** Recently, the rapid emergence of large-scale pre-trained language models has brought the research frontiers of NLP to a new era [36]. Among these models, BERT [11] is probably the most influential and popular model, which learns contextual token representations by a stack of transformer encoders, using two self-supervised learning objectives: masked language modeling and next sentence prediction. ALBERT [23] is a similar pre-trained language model that reduces the sizes of BERT-style models by using parameter sharing and factorization strategies. These models deal with languages with fixed-length contexts. To break the barrier limited by vanilla transformers, Transformer-XL [10] learns long-term contextual dependencies beyond a fixed length by recurrence and relative positional encoding. XLNet [48] further improves the performance of Transformer-XL, using the auto-regressive learning objective.

Besides the transformer encoder architecture, the encoder-decoder architecture has also been applied, such as T5 [38] and GPT-3 [3], which contain 11 billion and 175 billion model parameters, respectively. Despite their effectiveness, the large size of modern language models brings significant challenges for deployment in industrial applications where a lot of tasks need to be solved. In MeLL, we address this issue by using slowly and fast updated meta-learners, capable of handling an increasing number of tasks without introducing too many new parameters into the model.

## 3 MELL: THE PROPOSED FRAMEWORK

### 3.1 Overview

We start with some basic notations. Let $\mathcal{T}_n$ represent the $n$-th UIC task. $\mathcal{D}_n = \{(x_{n,i}, y_{n,i})\}$ is the training set of $\mathcal{T}_n$ where $x_{n,i}$ is the $i$-th input sample in $\mathcal{D}_n$ (i.e., the input text, which is a user query or response depending on the application scenario) and $y_{n,i}$ is the corresponding intent label of $x_{n,i}$. The goal of UIC $\mathcal{T}_n$ in the single-task setting is to learn a classifier $f_n$ that correctly predicts the intent label $y_{n,i} \in \mathcal{Y}_n$ of the input $x_{n,i}$, where $\mathcal{Y}_n$ is the classification label set of the task $\mathcal{T}_n$.

In the *large-scale EUIC* setting, we consider the situation where we are faced with an unlimited sequence of UIC tasks $\mathcal{T}_1, \mathcal{T}_2, \cdots$. In real-world applications, very often we have a small number of UIC tasks available to begin with. Hence, let $\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_N$ be $N$ existing base UIC tasks, $\mathcal{T}_{N+1}, \mathcal{T}_{N+2}, \cdots$ be the unlimited sequence of new UIC tasks. Our goal is to build a learning system $\mathcal{F} = \{f_1, f_2, \cdots, f_N, f_{N+1}, f_{N+2}, \cdots\}$ that continuously support obtaining classifiers for new UIC tasks $(f_{N+1}, f_{N+2}, \cdots)$ while maintaining the performance of existing classifiers $(f_1, f_2, \cdots, f_N)$. To be more specific, in the initial stage, we are given $N$ training sets $\mathcal{D}_1, \mathcal{D}_2, \cdots \mathcal{D}_N$ to train a multi-task UIC model for the $N$ base tasks. After that, our model can be automatically extended to an unlimited number of new tasks $\mathcal{T}_{N+1}, \mathcal{T}_{N+2}, \cdots$ where new tasks arrive sequentially. To alleviating the problems of *catastrophic forgetting* and *parameter explosion*, we design the model structure of MeLL, presented in Figure 2. It has different computation graphs during training and inference. Overall, it has four major components:

**Text Encoder:** We employ BERT [11] as our model backbone to learn the universal, deep representations of input texts across tasks. Here, we denote the representations of $x_{n,i}$ to be $Q(x_{n,i})$. As new UIC tasks continuously arrive, the BERT parameters are *slowly updated* to digest transferable knowledge across multiple tasks.

**Global Memory Network:** The global memory network is only applied during the training time, which contains a certain number of "slots" (denoted as $G$) to store class representations. For each task $\mathcal{T}_n$, we use the class label set $\mathcal{Y}_n$ as the task meta-information to retrieve class representations for feature computation afterwards. The memory units here are *fast updated* to acquire knowledge from new tasks. In our setting, as the sum of the numbers of distinct classes is increasing, we use the LRU replacement policy such that there are only a fix number of "slots" (denoted as $K$) in the memory.

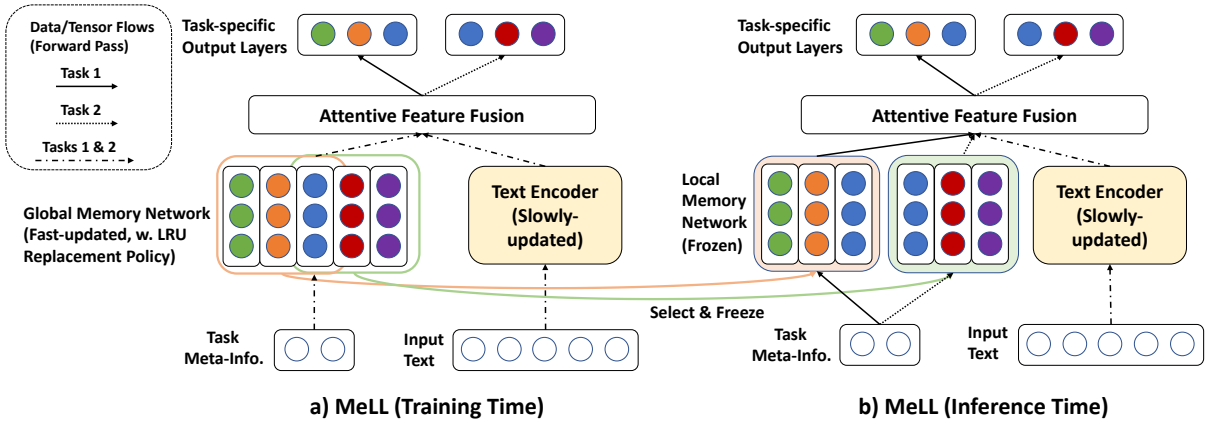



**Figure 2: Model structure of MeLL for large-scale EUIC. During training, the features used for UIC are attentively generated by the slowly-updated text encoder and the fast-updated global memory network. After the training process of a specific task, the respective class representations are copied from the global memory to a task-specific local memory network. During inference, we use the text encoder and the task-specific local memory to generate features for prediction. (Best viewed in color.)**

**Local Memory Networks:** The fast update of the global memory may significantly affect the final features used for prediction. To guarantee the performance of previous tasks is not affected, for each task $T_n$, we have a separate local memory $\mathcal{L}_n$ that copies the respective class representations from the global network with parameters frozen. During the inference time, we directly use the task-specific local memory $\mathcal{L}_n$ for feature computation.

**Task-specific Networks:** Finally, we use class representations and text representations $Q(x_{n,i})$ to generate attentive features for user intent prediction, denoted as $Att(x_{n,i})$. Each task has its own prediction head, treated as the prediction function $f_n$.

The training and inference procedures are summarized in Algorithm 1 and Algorithm 2. Specifically, we only show the inference procedure w.r.t. a specific task $\mathcal{T}_n$ (which is the same in either initial or lifelong learning stages). Details will be presented below.

## 3.2 Text Encoder

The text encoder is employed to learn deep contextual representations of input $x_{n,i}$. In MeLL, we follow common practice to take BERT [11] as default to generate the pooled output embeddings as $Q(x_{n,i})$. Similar encoder-based pre-trained language models can also be applied, such as ALBERT [23]. Note that in the lifelong learning stage, the learning rate of the encoder parameters should be set to a smaller value to avoid catastrophic forgetting of previously learned tasks.

## 3.3 Global and Local Memory Networks

The global memory network stores $K$ "slots" of class representations. Let $\mathcal{Y}_N$ be the collection of distant classes across the $N$ tasks, i.e., $\mathcal{Y}_N = \bigcup_{n=1}^{N} \mathcal{Y}_n$. We set $K \geq |\mathcal{Y}_N|$. In the initial learning stage, we set the global memory $G$ as follows.

For a class label $y^{(m)} \in \mathcal{Y}_N$, let $\mathcal{T}^{(m)}$ be the collection of tasks that involve $y^{(m)}$, i.e.,$\mathcal{T}^{(m)} = \{\mathcal{T}_n | n \in \{1, \cdots, N\}, y^{(m)} \in \mathcal{Y}_n\}$. $\mathcal{D}_n^{(m)}$ is the subset of $\mathcal{D}_n$ such that $\mathcal{D}_n^{(m)} = \{(x_{n,i}, y_{n,i}) \in \mathcal{D}_n | y_{n,i} = y^{(m)}\}$. We have the prototype representation vector $G_N^{(m)}$ of the class label $y^{(m)}$ as:

$$G_N^{(m)} = \frac{1}{|\mathcal{T}^{(m)}|} \sum_{\mathcal{T}_n \in \mathcal{T}^{(m)}} \frac{1}{|\mathcal{D}_n^{(m)}|} \sum_{(x_{n,i}, y_{n,i}) \in \mathcal{D}_n^{(m)}} Q(x_{n,i}) \quad (1)$$

**Algorithm 1** MeLL Training Procedure

1: // Initial Learning Stage
2: Initialize global memory $G$ based on $\mathcal{D}_1, \mathcal{D}_2, \cdots \mathcal{D}_N$.
3: **while** not converge **do**
4: Sample a task $\mathcal{T}_n$ from $\mathcal{T}_1, \mathcal{T}_2, \cdots \mathcal{T}_N$.
5: Read a batch $\{(x_{n,i}, y_{n_i})\}$ from $\mathcal{D}_n$.
6: Run through BERT to obtain representations $\{Q(x_{n,i})\}$.
7: Read global memory $G$ with the task meta-info. $\mathcal{Y}_n$ and text representations $\{Q(x_{n,i})\}$ to generate features $\{Att(x_{n,i})\}$ and pass them to the output layer $f_n$.
8: Update parameters of $f_n$, $G$ and the text encoder by back propagation.
9: **end while**
10: Create local memories $L_1, L_2, \cdots, L_N$ for $\mathcal{T}_1, \mathcal{T}_2, \cdots \mathcal{T}_N$, with all parameters frozen.
11: // Lifelong Learning Stage (Assume task $\mathcal{T}_j$ arrives, $j > N$.)
12: Update global memory $G$ based on $\mathcal{D}_j$ w. LRU replacement.
13: Train the model with a new task-specific output layer $f_j$ and a smaller learning rate on BERT. Parameters of $f_n$, $G$ and BERT are updated.
14: Create local memory $L_j$ for $\mathcal{T}_j$ with all parameters frozen.

**Algorithm 2** MeLL Inference Procedure

1: Read a batch $\{(x_{n,i})\}$ from an unlabeled dataset of task $\mathcal{T}_n$.
2: Run through BERT to obtain representations $\{Q(x_{n,i})\}$.
3: Read local memory $L_n$ with the task meta-info. $\mathcal{Y}_n$ and text representations $\{Q(x_{n,i})\}$ to generate features $\{Att(x_{n,i})\}$ and pass them to the task-specific output layer $f_n$.
4: Make predictions $\{\hat{y}_{n,i}\}$ based on $f_n(Att(x_{n,i}))$.

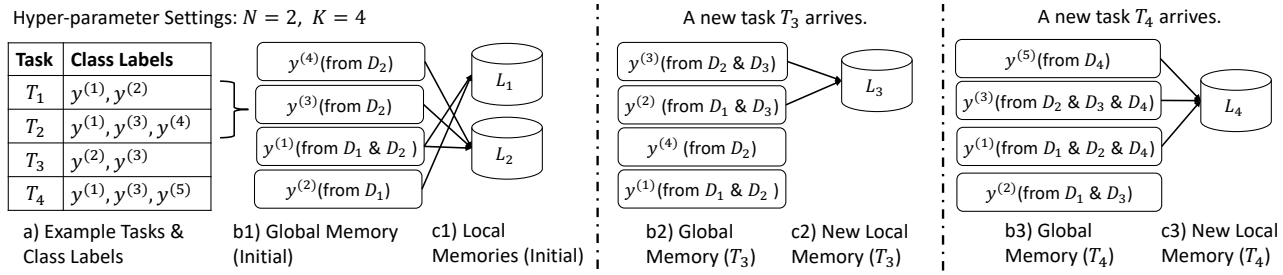



**Figure 3: A toy example of how the global memory and local memories update through time.**

We can see that, $G_N^{(m)}$ is the averaged pooled result of the prototype vectors of all the tasks $\mathcal{T}^{(m)}$. By aggregating all the $|\mathcal{Y}_N|$ class representations $G_N^{(m)}$, we finish the initial computation of $G$, treated as the high-level representations of all the $N$ tasks.

We further consider the lifelong learning case. For generality, we assume the model has already been trained over $j-1$ tasks $\mathcal{T}_1, \cdots, \mathcal{T}_{j-1}$ (with $j > N$), and a new task $\mathcal{T}_j$ arrives. For a class $y^{(m)} \in \mathcal{Y}_j$, if the corresponding class representation $G_{j-1}^{(m)}$ is present in $G$. The update rule is as follows:

$$G_j^{(m)} = (1-\gamma)G_{j-1}^{(m)} + \frac{\gamma}{|\mathcal{D}_j^{(m)}|} \sum_{(x_{n,i}, y_{n,i}) \in \mathcal{D}_j^{(m)}} Q(x_{n,i}) \quad (2)$$

where $\gamma \in (0,1)$ is a pre-defined hyper-parameter, balancing the relative importance of existing tasks and the new task. Readers may notice that the "size" of the global memory has a limit of $K$. When it is already full, we remove one item in $G$ that has been least recently visited. The new class representation is inserted into $G$, computed as: $G_j^{(m)} = \frac{1}{|\mathcal{D}_j^{(m)}|} \sum_{(x_{n,i}, y_{n,i}) \in \mathcal{D}_j^{(m)}} Q(x_{n,i})$. We employ the LRU replacement policy because we obverse that the topic trend of tasks may drift over time. The recently updated class representations have a large probability to be used again in near future. A toy sample is shown in Figure 3.

At the end of the task-wise learning process, for a current task $\mathcal{T}_n$, we copy the class representations in $G$ w.r.t. classes $\mathcal{Y}_n$ to its own local memory $\mathcal{L}_n$ for the inference purpose, with all parameters frozen. Hence, when the global memory network is fast updated, the changes to $G$ will not affect the inference for existing tasks.

## 3.4 Feature Fusion and Model Output

After we have introduced the generation of $Q(x_{n,i})$ and $G$, we now discuss the forward pass in MeLL. Assume we are learning the task $\mathcal{T}_n$, with the current training instance as $(x_{n,i}, y_{n,i}) \in \mathcal{D}_n$. We use the class label set $\mathcal{Y}_n$ to query $G$, to generate the current class representation $G_n^{(m)}$ for each class $y^{(m)} \in \mathcal{Y}_n$. [3] The attentive score $\alpha^{(m)}(x_{n,i})$ is computed as:

$$\alpha^{(m)}(x_{n,i}) = \text{softmax}(Q(x_{n,i})^T \cdot G_n^{(m)}) \quad (3)$$

Note that the computation of the attentive score is slightly different from standard practice. The final attentive feature set sent to the task-specific output layer $Att(x_{n,i})$ is computed as follows:

$$Att(x_{n,i}) = Q(x_{n,i}) + \sum_{y^{(m)} \in \mathcal{Y}_n} \alpha^{(m)}(x_{n,i}) \cdot G_n^{(m)} \quad (4)$$

where $Q(x_{n,i})$ is regarded as the residual here. The prediction result $\hat{y}_{n,i}$ is given by: $\hat{y}_{n,i} = f_n(Att(x_{n,i}))$.

As the gradients of the parameters in the output layer, the text encoder and the global memory are fully differential, we update these parameters by back propagation. The parameters in local memory networks are not updated during back propagation.

## 3.5 Algorithmic Analysis

We further give a deeper analysis on MeLL, focusing on how lifelong learning affects the computational complexity. Assume we are learning a new task $\mathcal{T}_j$ ($j > N$). Let $\mathcal{M}$ be the total number of parameters in the BERT encoder, and $d$ be the dimension of token embeddings. It is trivial to derive that the global memory, the local memory for $\mathcal{T}_j$, the attentive fusion layer and the output layer for $\mathcal{T}_j$ have $K \cdot d$, $|\mathcal{Y}_j| \cdot d$, 0 and $(d+1)|\mathcal{Y}_j|$ parameters, respectively. The total number of parameters w.r.t. all the first $j$ tasks is $\mathcal{M} + K \cdot d + (2d+1) \cdot |\bigcup_{n=1}^{j} \mathcal{Y}_j|$, of which $\mathcal{M} + (d+1) \cdot |\bigcup_{n=1}^{j} \mathcal{Y}_j|$ parameters are trainable during back propagation. As the BERT encoder has the most parameters, the increasing number of tasks brings minimal effect on model size. Therefore, the *parameter explosion* issue is successfully addressed by MeLL.

Additionally, different from lifelong learning algorithms with replay strategies [20, 40], our approach does not require any replay operations. Instead, we employ the copy mechanism and the slow update of the text encoder to avoid *catastrophic forgetting*. Refer to the experiments for detailed results.

# 4 EXPERIMENTS

## 4.1 Datasets

To evaluate the effectiveness of MeLL, we construct two datasets, including a fused dataset for query intent classification in task-oriented dialogues (TaskDialog-EUIC) and a real-world e-commerce dataset for response intent classification in hotline agents (Hotline-EUIC). TaskDialog-EUIC is built from three public datasets: Snips [7], TOP semantic parsing [14] and Facebook Multilingual Task Oriented Dataset [41]. We fuse samples from those datasets and split them into multiple tasks with overlapped label sets. Overall, we obtain 90 tasks, containing more than ten thousands samples in total. Hotline-EUIC is collected and annotated from the hotline

[3] The methods for feature fusion and model output are the same in both initial and lifelong learning stages. If we are dealing with tasks in the initial stage ($n \leq N$), we use the memory $G_N^{(m)}$ instead of $G_n^{(m)}$ to compute the attentive score.

data produced from an e-commerce dialogue system AliMe [24]. We first use an industrial Automatic Speech Recognition (ASR) system with high accuracy to convert hotline audios to texts, and hire annotators to filter out texts with ASR errors and annotate the intentions of the remaining texts. This dataset also has 90 tasks from various fields, including retail, trip agents and express delivery with a variety of user intent sets in each field. We randomly select 30% of all the tasks for each dataset as "base tasks" that are used for initial model training, and take the rest as new tasks used in the lifelong learning stage. We randomly split the data in each task into training/development/testing sets. More details about dataset construction and statistics can be found in Appendix A.1.

## 4.2 Baselines and Evaluation Metrics

We compare MeLL against the following strong baselines:
**MTL:** uses the multi-task fine-tuning approach [45] on all the tasks. In this setting, we assume that the datasets of all the tasks are available for us and do not apply the lifelong learning setting. This model can produce the *upper-bound model performance* in our work.
**Single:** trains one BERT classifier [11] for each task. It unavoidably suffers from the parameter explosion problem when the number of tasks is large.
**Lifelong-freeze:** first uses the multi-task fine-tuning approach [45] on the $N$ base tasks. Next, it freezes the BERT encoder and only tunes the task-specific output layer for each new task.
**Lifelong-seq:** is similar to "Lifelong-freeze" except that the BERT encoder will also be tuned in a sequential manner when a new task arrives. Hence, it can suffer from the catastrophic forgetting problem.
**Lifelong-replay:** is an extension of "Lifelong-seq" that employs randomly-sampled data from previous tasks as experience replay to re-train models for previous tasks [4].

For evaluation, we average all the accuracy and macro-F1 scores across all tasks to compare different methods.

## 4.3 Implementation Details

We use `bert-base-uncased` (L=12, H=768, A=12, Total Parameters=110M) [4] as the initialization of the BERT encoder of all of the methods on TaskDialog-EUIC, and use `roberta-tiny-chinese`[5] (L=2, H=768, A=12, Total Parameters=31M) on Hotline-EUIC. For MeLL on both datasets, we use {EN=10, BS=64, SL=64, LR=1e-4} as the hyper-parameters for multi-task learning on the *base tasks*, where EN is the number of epochs, BS is the batch size, SL is the sequence length and LR is the learning rate. We set the learning rate to be 1e-6 for tuning the BERT encoder and 1e-3 for tuning the task-specific classification heads. For the LRU replacement policy, we set the maximum memory size as the distinct number of the labels in *base tasks*. All of the algorithms are implemented in PyTorch and run on 4 Tsela V100 GPUs. One can find more implementation details of baseline models in Appendix A.2.2.

[4]https://github.com/google-research/bert
[5]We choose a small pre-trained language model since it is more suitable for online deployment in our application. The pre-training details of this model and the comparison between different Chinese pre-trained models can be found in Appendix A.2.1.

## 4.4 Experimental Results and Analysis

*4.4.1 Comparing with baselines.* Table 1 shows the general testing performance over TaskDialog-EUIC and Hotline-EUIC. From the results, we have the following findings:

- MeLL consistently improves the performance, especially on new tasks. Specifically, MeLL achieves 1.65% accuracy and 1.48% F1 improvements compared with the best baseline model on TaskDialog-EUIC, and 2.72% accuracy and 5.43% F1 improvements on Hotline-EUIC. The improvement rates on new tasks are also significant. MeLL has 2.36% F1 improvements compared with the best baseline on new tasks of the TaskDialog-EUIC dataset and 5.05% on the Hotline-EUIC dataset.
- Compared with the Single baseline, Lifelong-freeze has better overall performance but worse performance in new tasks on the Hotline-EUIC dataset, which suggests that freezing the BERT encoder is not a good solution for the EUIC task. A common practice of lifelong learning (i.e., Lifelong-unfreeze) does not work well in the EUIC problem, either. We have observed that the performance drops dramatically (especially on base tasks) when the BERT encoder is fine-tuned sequentially because of the catastrophic forgetting problem. Adding the replay mechanism can ease that problem but the performance is still not satisfactory.
- Compared with the Single baseline, MeLL has a much smaller model size but a significant performance gain. Compared with Lifelong-replay, the training data and training steps of MeLL are much smaller. MeLL also has improvements on the performance, especially on the new tasks. Compared with MTL, MeLL has a performance gap but is close, since new tasks for MeLL arrives sequentially. The gap is smaller than other baselines, which proves the effectiveness of the global memory of the meta-knowledge.

*4.4.2 The influence of model components on the Hotline-EUIC dataset.* In this experiment, we explore how the three components influence the final performance of MeLL: i) Meta knowledge, ii) Slow learner iii) LRU replacement policy. The results can be found in Table 2. We can see that the meta knowledge that we introduce plays an important role in the overall model performance. After the transferable knowledge is passed from existing tasks to new tasks, the model can achieve a better performance. When the BERT encoder is not tuned, the performance also drops, which proves the necessity to use BERT as a slow learner in MeLL. We can also find that introducing the LRU replacement policy only has 0.39% performance drop, which proves the effectiveness of this policy.

*4.4.3 The influence of the learning rate.* We explore how the learning rates of the slow and fast learners on lifelong learning phase affect the final performance, shown in Figure 4.

For the slow learner, we find that i) 1e-5 is too large to tune the BERT encoder and the performance drops dramatically (about 8% drops on accuracy and 10% drops on F1); ii) the performance through smaller learning rates (1e-7, 5e-7, 1e-6) is steady and 1e-6 is the best hyper-parameter setting of the learning rate of the slow learner; iii) the learning rate of the slow learner affects *base tasks*

**Table 1: Comparison of different models over two datasets. * denotes that it can be unfair to directly compare MeLL with these models. Compared with MTL (upper-bound), MeLL has no global view of the all tasks. Compared with Single, MeLL needs only one global model instead of one model for each task. Compared with Lifelong-replay, MeLL does not need to store data from previous tasks nor perform experience replay.**

| Task | TaskDialog-EUIC | | | | Hotline-EUIC | | | |
|---|---|---|---|---|---|---|---|---|
| Results | All tasks | | New tasks | | All tasks | | New tasks | |
| | Accuracy | F1 | Accuracy | F1 | Accuracy | F1 | Accuracy | F1 |
| MTL (Upper-bound)* | 0.9597 | 0.9590 | 0.9568 | 0.9562 | 0.9788 | 0.9480 | 0.9832 | 0.9523 |
| Single* | 0.9006 | 0.8974 | 0.9005 | 0.8969 | 0.9196 | 0.8685 | 0.9239 | 0.8814 |
| Lifelong-freeze | 0.9214 | 0.9194 | 0.9015 | 0.8988 | 0.9401 | 0.8798 | 0.9259 | 0.8501 |
| Lifelong-seq | 0.3140 | 0.2043 | 0.3447 | 0.2455 | 0.4517 | 0.3485 | 0.5272 | 0.4238 |
| Lifelong-replay* | 0.6225 | 0.5481 | 0.5485 | 0.4573 | 0.8215 | 0.8260 | 0.9420 | 0.8553 |
| MeLL | **0.9379** | **0.9342** | **0.9271** | **0.9224** | **0.9673** | **0.9341** | **0.9675** | **0.9319** |

**Table 2: Ablation study on the Hotline-EUIC dataset.**

| Ablation | F1 | Improv. Rate |
|---|---|---|
| MeLL | 0.9341 | N/A |
| w/o Meta knowledge | 0.9178 | -1.63% |
| w/o Slow learner | 0.9269 | -0.72% |
| w/o LRU replacement policy | 0.9380 | +0.39% |

more than *new tasks* since the classification heads for *base tasks* have no chance to adjust once the BERT encoder has changed.

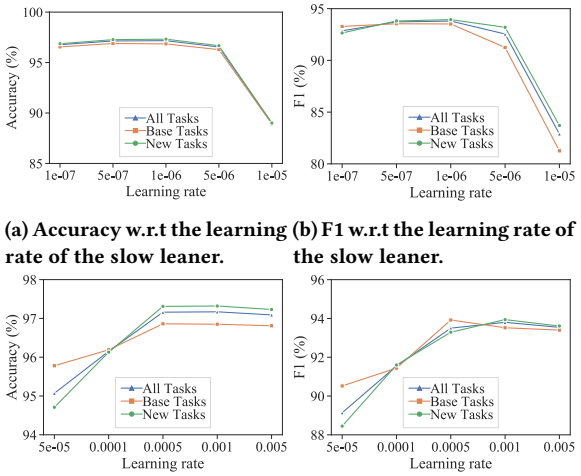
For the fast learner, we find that i) a large learning rate is required. The overall performance quickly improves when the learning rate is increased from 5e-5 to 5e-4 and slowly improves from 5e-4 to 1e-3; ii) On the aspect of the accuracy score, the new tasks are more sensitive to a large learning rate, since we need to tune the head classifier from scratch for the new tasks. iii) The learning rate of the fast learner also influences the base tasks even though only the BERT encoder is slowly updated. That is because the optimization of the head classifier also influences the optimization of the BERT encoder through back propagation.

*4.4.4 The influence of pre-trained language models.* We then use different pre-trained language models as our text encoder on Hotline-EUIC. The results show that MeLL consistently improves the EUIC results. The pre-trained language model that we use are BERT-base, ALBERT-base trained on CLUECorpus2020 [6] and RoBerta-base released in [8]. We also pre-train our two-layer RoBerta-tiny on our own datasets. The result shown in Table 3 confirms that MeLL achieves the best performance with any type of pre-trained language models.

*4.4.5 The influence between base tasks and new tasks.* In this experiment, we analyse two sides of knowledge transferring on the Hotline-EUIC dataset:

**How base tasks influence new tasks.** We show how the number of *base tasks* influences the performance on *new tasks* in Figure 5a.

[6]https://github.com/CLUEbenchmark/CLUEPretrainedModels



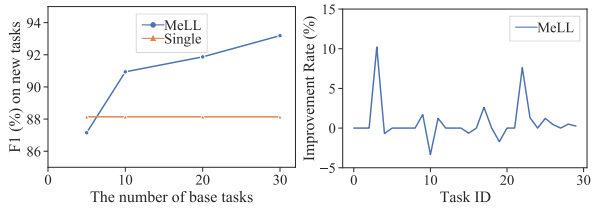

**(a) Accuracy w.r.t the learning rate of the slow leaner.** **(b) F1 w.r.t the learning rate of the slow leaner.**

**(c) Accuracy w.r.t the learning rate of the fast leaner.** **(d) F1 w.r.t the learning rate of the fast leaner.**

**Figure 4: The performance comparison between different learning rates of the slow learner and the fast learner. The learning rate of the fast learner is 1e-3 for sub-figure (a) and (b), and the learning rate of the slow learner is 1e-6 for sub-figure (c) and (d).**

We find that for both of MeLL and the Single model, the performance improves when the number of *base tasks* increases, which confirms that the transferable knowledge can be passed from existing tasks to new tasks. The improvement rate of MeLL increases faster than Single, proving the effectiveness of incorporating the meta knowledge in MeLL.

**How new tasks influence base tasks.** We then explore how model fine-tuning on new tasks influences the performance on *base tasks*. In Figure 5b, we find that most of the base tasks obtain performance gain from new tasks, and the performances on some tasks are even improved by more than 5% in terms of F1. These observations confirm that previous tasks can be positively influenced by new tasks on the MeLL model.




(a) Performance (F1) on new tasks w.r.t the number of *base tasks*.

(b) Performance (F1) improvement rate on *base tasks* after fine-tuning on new tasks.

**Figure 5: The exploration of the influence between base tasks and new tasks of the MeLL model.**

**Table 3: Comparison of different pre-trained models on the Hotline-EUIC dataset.**

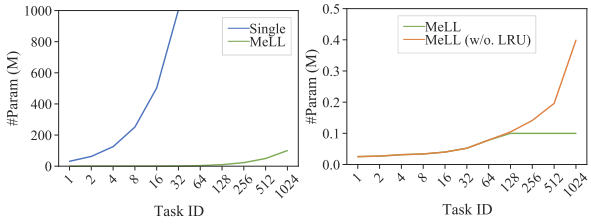
| Model | BERT-base | ALBERT-base | RoBerta-base | RoBerta-tiny (ours) |
|---|---|---|---|---|
| Single | 0.8699 | 0.7895 | 0.8793 | 0.8685 |
| Lifelong-freeze | 0.8704 | 0.8288 | 0.9115 | 0.8798 |
| MeLL | **0.9046** | **0.8854** | **0.9419** | **0.9341** |

## 4.5 Online Deployment

We have also collected and annotated hotline data from AliMe and built a large-scale dataset named Hotline-EUIC-Large to train models for real-world industry deployment. This dataset contains 1,004 UIC tasks from varies domains, including 1,327,441 training samples, 161,098 development samples and 103,868 test samples in total. There are 489 distinct labels and the maximum number of the labels of one task is 24. We train our MeLL model in the order of when the task is created. The first 100 tasks are treated as our *base tasks*. The LRU memory size is set as 100. One can find more details of the dataset and the offline experimental results in the appendix. We then deploy our model in the Alime hotline agent. After deployment, we collect back annotated data of 600 tasks from the online system, compare the performance of our model with the previous single-model system, and report the F1 score in Table 4. The previous online system includes thousands of single models. Each model is a TextCNN model [22] distilled from an fine-tuned ALBERT-base model [23]. Overall, our model improves the overall F1 in 8.61%. More offline results and deployment details are presented in Appendix A.3.

## 4.6 Scalability Analysis

In this section, we explore how MeLL reduces the overall complexity of the deployed models and how LRU helps reduce memory explosion. We simulate the situation where there are 1,024 tasks at most. We compute the numbers of the parameters of Single and MeLL. The overall parameter reduction results can be found in Figure 6a, We can find that the numbers of model parameters in Single and MeLL are linearly increasing but the growth rate of MeLL is in a much slower speed. When the number of the tasks is 1,000, the overall number of parameters of the 1,000 single-task models are




(a) Overall #Param w.r.t Task ID comparison between Single and MeLL.

(b) Memory #Param w.r.t Task ID comparison between MeLL and MeLL (w/o. LRU).

**Figure 6: The parameter scale comparison between Single and MeLL. Here "#Param" means the number of the parameters. Note that the x-axis is in an exponential scale.**

**Table 4: The online performance comparison between MeLL and the online system.**

| Method | F1 | Relative Improv. |
|---|---|---|
| Online system (Single) | 0.8359 | N.A. |
| MeLL (w. LRU) | **0.9079** | 8.61% |

150x larger than the MeLL model. The memory parameter reduction results of the LRU replacement policy can be found in Figure 6b. The curve is overlapped when the global memory is not fully occupied but the number of the parameters of MeLL (w/o. LRU) still increases quickly after the number of labels exceeds the maximum memory size. These two findings confirm the superiority of our MeLL model with the LRU replacement policy.

## 5 CONCLUSION

In this paper, we formally introduce the task of *large-scale EUIC*, and propose the *Meta Lifelong Learning* (MeLL) framework to address this task. MeLL employs a slowly updated text encoder to learn text representations across tasks and the global/local memory networks to learn task semantics. The MeLL framework enables effective knowledge transfer through time and alleviates catastrophic forgetting and parameter explosion problems at the same time. Extensive experiments on both English and Chinese EUIC datasets show the effectiveness of MeLL, which consistently outperforms strong baselines. We have also deploy MeLL on a real industrial dialogue system AliMe. The online A/B test results show the superiority of our method. In the future, we will further explore how MeLL can be employed to solve other tasks and support other applications.

## ACKNOWLEDGMENTS

This work is supported by the National Key Research and Development Program of China (No. 2018AAA0101400), the National Nature Science Foundation of China (Nos. 62036009, 61936006), the Alibaba-Zhejiang University Joint Institute of Frontier Technologies and Innovation Capability Support Program of Shaanxi (No. 2021TD-05). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

## REFERENCES


[1] Ali Ahmadvand, Surya Kallumadi, Faizan Javed, and Eugene Agichtein. 2020. JointMap: Joint Query Intent Understanding For Modeling Intent Hierarchies in E-commerce Search. In *SIGIR*. 1509–1512.

[2] Magdalena Biesialska, Katarzyna Biesialska, and Marta R. Costa-jussà. 2020. Continual Lifelong Learning in Natural Language Processing: A Survey. In *COLING*. 6523–6541.

[3] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *NerIPS*.

[4] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. 2019. Efficient Lifelong Learning with A-GEM. In *ICLR*.

[5] Xinyang Chen, Sinan Wang, Bo Fu, Mingsheng Long, and Jianmin Wang. 2019. Catastrophic Forgetting Meets Negative Transfer: Batch Spectral Shrinkage for Safe Transfer Learning. In *NeurIPS*. 1906–1916.

[6] Yung-Sung Chuang, Shang-Yu Su, and Yun-Nung Chen. 2020. Lifelong Language Knowledge Distillation. In *EMNLP*. 2914–2924.

[7] Alice Coucke, Alaa Saade, Adrien Ball, Théodore Bluche, Alexandre Caulier, David Leroy, Clément Doumouro, Thibault Gisselbrecht, Francesco Caltagirone, Thibaut Lavril, et al. 2018. Snips voice platform: an embedded spoken language understanding system for private-by-design voice interfaces. *arXiv preprint arXiv:1805.10190* (2018).

[8] Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Ziqing Yang, Shijin Wang, and Guoping Hu. 2019. Pre-Training with Whole Word Masking for Chinese BERT. *arXiv preprint* arXiv: 1906.08101 (2019).

[9] Adam Czyzewski, Jeffrey Dalton, and Anton Leuski. 2020. Agent Dialogue: A Platform for Conversational Information Seeking Experimentation. In *SIGIR*. 2121–2124.

[10] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc Viet Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *ACL*. 2978–2988.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*. 4171–4186.

[12] Qi Fan, Wei Zhuo, Chi-Keung Tang, and Yu-Wing Tai. 2020. Few-Shot Object Detection With Attention-RPN and Multi-Relation Detector. In *CVPR*. 4012–4021.

[13] Rashmi Gangadharaiah and Balakrishnan Narayanaswamy. 2019. Joint Multiple Intent Detection and Slot Labeling for Goal-Oriented Dialog. In *NAACL-HLT*. 564–569.

[14] Sonal Gupta, Rushin Shah, Mrinal Mohit, Anuj Kumar, and Mike Lewis. 2018. Semantic parsing for task oriented dialog using hierarchical representations. *arXiv preprint arXiv:1810.07942* (2018).

[15] Matthew Henderson, Ivan Vulic, Iñigo Casanueva, Pawel Budzianowski, Daniela Gerz, Sam Coope, Georgios Spithourakis, Tsung-Hsien Wen, Nikola Mrksic, and Pei-Hao Su. 2019. PolyResponse: A Rank-based Approach to Task-Oriented Dialogue with Application in Restaurant Search and Booking. In *EMNLP-IJCNLP*. ACL, 181–186.

[16] Nithin Holla, Pushkar Mishra, Helen Yannakoudakis, and Ekaterina Shutova. 2020. Meta-Learning with Sparse Experience Replay for Lifelong Language Learning. *arXiv preprint* arXiv: 2009.04891 (2020).

[17] Timothy M. Hospedales, Antreas Antoniou, Paul Micaelli, and Amos J. Storkey. [n.d.]. Meta-Learning in Neural Networks: A Survey. *arXiv preprint* arXiv: 2004.05439 ([n. d.]).

[18] Wenpeng Hu, Zhou Lin, Bing Liu, Chongyang Tao, Zhengwei Tao, Jinwen Ma, Dongyan Zhao, and Rui Yan. 2019. Overcoming Catastrophic Forgetting for Continual Learning via Model Adaptation. In *ICLR*.

[19] Muhammad Irfan, Jiangbin Zheng, Muhammad Iqbal, and Muhammad Hassan Arif. 2021. A novel lifelong learning model based on cross domain knowledge extraction and transfer to classify underwater images. *Inf. Sci.* 552 (2021), 80–101.

[20] David Isele and Akansel Cosgun. 2018. Selective Experience Replay for Lifelong Learning. In *AAAI*. 3302–3309.

[21] Jyun-Yu Jiang and Pu-Jen Cheng. 2016. Classifying User Search Intents for Query Auto-Completion. In *ICTIR*. 49–58.

[22] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. In *EMNLP*. 1746–1751.

[23] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *ICLR*.

[24] F. Li, M. Qiu, H. Chen, X. Wang, X. Gao, J. Huang, J. Ren, Z. Zhao, W. Zhao, L. Wang, and G. Jin. 2017. AliMe Assist: An Intelligent Assistant for Creating an Innovative E-commerce Experience. In *CIKM '17*.

[25] Wenbin Li, Lei Wang, Jinglin Xu, Jing Huo, Yang Gao, and Jiebo Luo. 2019. Revisiting Local Descriptor Based Image-To-Class Measure for Few-Shot Learning. In *CVPR*. 7260–7268.

[26] Zhizhong Li and Derek Hoiem. 2018. Learning without Forgetting. *IEEE Trans. Pattern Anal. Mach. Intell.* 40, 12 (2018), 2935–2947.

[27] Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-Task Deep Neural Networks for Natural Language Understanding. In *ACL*. 4487–4496.

[28] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR* abs/1907.11692 (2019).

[29] Samuel Louvan and Bernardo Magnini. 2020. Recent Neural Methods on Slot Filling and Intent Classification for Task-Oriented Dialogue Systems: A Survey. In *COLING*. 480–496.

[30] Sijie Mai, Haifeng Hu, and Jia Xu. 2019. Attentive matching network for few-shot learning. *Comput. Vis. Image Underst.* 187 (2019).

[31] Farhad Nooralahzadeh, Giannis Bekoulis, Johannes Bjerva, and Isabelle Augenstein. 2020. Zero-Shot Cross-Lingual Transfer with Meta Learning. In *EMNLP*. 4547–4562.

[32] Abiola Obamuyide and Andreas Vlachos. 2019. Model-Agnostic Meta-Learning for Relation Classification with Limited Supervision. In *ACL*. 5873–5879.

[33] German Ignacio Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. 2019. Continual lifelong learning with neural networks: A review. *Neural Networks* 113 (2019), 54–71.

[34] Kun Qian and Zhou Yu. 2019. Domain Adaptive Dialog Generation via Meta Learning. In *ACL*. 2639–2649.

[35] Libo Qin, Xiao Xu, Wanxiang Che, and Ting Liu. 2020. Towards Fine-Grained Transfer: An Adaptive Graph-Interactive Framework for Joint Multiple Intent Detection and Slot Filling. In *EMNLP (Findings)*. 1807–1816.

[36] Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained Models for Natural Language Processing: A Survey. *arXiv preprint* arXiv: 2003.08271 (2020).

[37] Chen Qu, Liu Yang, W. Bruce Croft, Johanne R. Trippas, Yongfeng Zhang, and Minghui Qiu. 2018. Analyzing and Characterizing User Intent in Information-seeking Conversations. In *SIGIR*. 989–992.

[38] 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. *J. Mach. Learn. Res.* 21 (2020), 140:1–140:67.

[39] Matthew Riemer, Ignacio Cases, Robert Ajemian, Miao Liu, Irina Rish, Yuhai Tu, and Gerald Tesauro. 2019. Learning to Learn without Forgetting by Maximizing Transfer and Minimizing Interference. In *ICLR*.

[40] Mohammad Rostami, Soheil Kolouri, and Praveen K. Pilly. 2019. Complementary Learning for Overcoming Catastrophic Forgetting Using Experience Replay. In *IJCAI*. 3339–3345.

[41] Sebastian Schuster, Sonal Gupta, Rushin Shah, and Mike Lewis. 2018. Cross-lingual transfer learning for multilingual task oriented dialog. *arXiv preprint arXiv:1810.13327* (2018).

[42] Jetze Schuurmans, Flavius Frasincar, and Erik Cambria. 2020. Intent Classification for Dialogue Utterances. *IEEE Intell. Syst.* 35, 1 (2020), 82–88.

[43] Jake Snell, Kevin Swersky, and Richard S. Zemel. 2017. Prototypical Networks for Few-shot Learning. In *NIPS*. 4077–4087.

[44] Asa Cooper Stickland and Iain Murray. 2019. BERT and PALs: Projected Attention Layers for Efficient Adaptation in Multi-Task Learning. In *ICML*, Vol. 97. 5986–5995.

[45] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to Fine-Tune BERT for Text Classification?. In *CCL*. 194–206.

[46] Zeke Xie, Fengxiang He, Shaopeng Fu, Issei Sato, Dacheng Tao, and Masashi Sugiyama. 2020. Artificial Neural Variability for Deep Learning: On Overfitting, Noise Memorization, and Catastrophic Forgetting. *arXiv preprint* arXiv:2011.06220 (2020).

[47] Liu Yang, Minghui Qiu, Chen Qu, Cen Chen, Jiafeng Guo, Yongfeng Zhang, W. Bruce Croft, and Haiqing Chen. 2020. IART: Intent-aware Response Ranking with Transformers in Information-seeking Conversation Systems. In *WWW*. 2592–2598.

[48] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *NerIPS*. 5754–5764.

[49] Ruqing Zhang, Jiafeng Guo, Yixing Fan, Yanyan Lan, and Xueqi Cheng. 2020. Query Understanding via Intent Description Generation. In *CIKM*. 1823–1832.

[50] Weisheng Zhang, Kaisong Song, Yangyang Kang, Zhongqing Wang, Changlong Sun, Xiaozhong Liu, Shoushan Li, Min Zhang, and Luo Si. 2020. Multi-Turn Dialogue Generation in E-Commerce Platform with the Context of Historical Dialogue. In *EMNLP (Findings)*. 1981–1990.

# A APPENDIX

## A.1 Dataset Details

The TaskDialog-EUIC dataset is constructed from three open-source task-oriented dialogue datasets: Snips [7], TOP semantic parsing [14] and Facebook Multilingual Task Oriented Dataset [41]. In these datasets, only samples related to query intent classification are utilized. Even though a few samples of these datasets are related to other tasks such as semantic parsing and cross-lingual understanding. We first fuse data samples from those datasets and then split them into multiple tasks with overlapped label sets. To guarantee the variety and the inner-task scenario association for the generated tasks, all intent labels from these three datasets are initially clustered into a certain number of groups according to their averaged word embeddings. During the task generation process, we ensure that all the intent labels within a task belong to the same clustering group.

The Hotline-EUIC and Hotline-EUIC-Large datasets are from the real-world industry product AliMe. In the hotline scenario, AliMe provides services for over 600 businesses. Each business has different questions for the users to answer. Based on the users' responses, the models are required to classify the responses into a variety of answer types. Overall, we have more than 200 domains, resulting in over one thousand tasks for the models to solve. Some domains are semantically different from each other, e.g. map service and health service, while others are close to each other, e.g. food takeout service and express delivery service. Thus, some tasks are close to each other in semantics and share some common user intents. With the development of AliMe, the numbers of domains and tasks are continuously growing. A few example tasks are shown in Table 7.

The statistics of these three datasets are shown in Table 5.

## A.2 Implementation Details

*A.2.1 Pre-training Details.* Here we describe how the pre-trained model `robert-tiny-chinese` used for the Chinese datasets is produced. Following the work [8, 28], we use BERT as the encoder and the Wikipedia text [7] for pre-training. We only use the Masked Language Modeling (MLM) loss and the whole word masking [8] for pre-training. We use the batch size of 256, the sequence length of 512, the learning rate of 1e-4 and the Adam optimizer to train this model. We then further pre-train the model of 100K steps on an unlabeled hotline dataset generated from AliMe, using the learning rate of 1e-5.

*A.2.2 Baseline Details.* For all the models on both two datasets, the number of epochs, the batch size and the sequence length are set to 10, 64 and 128, respectively. The training strategies and the learning rates are a little different from the baseline models:

- **MTL:** Follow the previous work [45], We use data samples of the same task as training data in one batch and tune the learning rate from {1e-3, 5e-4, 1e-4 }.
- **Single:** For each task, we tune the learning rate from {1e-3, 5e-4, 1e-4 }.
- **Lifelong-freeze:** We first uses the multi-task fine-tuning approach [45] on the $N$ base tasks. The learning rate is 1e-4. Then, we freeze the BERT encoder and use the learning rate 1e-3 to tune the prediction heads for each new task.
- **Lifelong-seq:** The muti-task learning process of base tasks are similar to Lifelong-freeze but we use the learning rate of 1e-4 to tune both the BERT encoder and the prediction heads.
- **Lifelong-replay:** The training process and hyper-parameters are same as Lifelong-seq, except for each new task, we sample 7% of the data samples from the previous tasks for experience replay.

[7] https://dumps.wikimedia.org/zhwiki/latest/

**Table 5: Experimental data statistics.**

| | TaskDialog-EUIC | Hotline-EUIC | Hotline-EUIC-Large |
|---|---|---|---|
| #Train. | 12,845 | 90,594 | 1,327,441 |
| #Dev. | 2,569 | 10,114 | 161,098 |
| #Test | 2,569 | 11,803 | 103,868 |
| #Tasks | 90 | 90 | 1,004 |
| #Base tasks | 30 | 30 | 100 |
| #Distinct labels | 26 | 71 | 489 |

**Table 6: Performance of different models on the Hotline-EQID-Large dataset.**

| Results | All tasks | | New tasks | |
|---|---|---|---|---|
| | Accuracy | F1 | Accuracy | F1 |
| Single | 0.9135 | 0.8511 | 0.9107 | 0.8544 |
| Lifelong-freeze | 0.8828 | 0.8030 | 0.8741 | 0.7919 |
| MeLL | **0.9599** | **0.9072** | **0.9567** | **0.9042** |

## A.3 Experimental Details on Large-scale Dataset

We have tested three models on the Hotline-EUIC-Large dataset. The the offline experimental results are shown in Table 6. Compared with the results on Hotline-EUIC in Table 1, we can find the gap between MeLL with Single is steady but the gap between MeLL and Lifelong-freeze is larger, which proves the superiority of MeLL on large-scale EUIC tasks.

**Table 7: Example tasks in the Hotline-EUIC dataset.**

| Domain | Task Description | User Response Intents |
|---|---|---|
| Map | Check whether the shop name is correct<br>Check whether the shop is still open | {Yes, No, Other}<br>{Open, Close, Not sure} |
| Health | Ask about the medication history<br>Ask about the fasting plasma glucose | {1 Year, 1-3 Years, >3 Years}<br>{Normal, Pre-diabetes, Diabetes} |
| Food takeout | Check if the customer is available<br>to pick up the takeout<br>Satisfaction survey | { Available, Not available,<br>Deliver as soon as possible }<br>{ Satisfied, Slow delivery, Food spilled, Not received } |
| Express delivery | Check if the customer is available<br>to pick up the delivery<br>Satisfaction survey | { Available, Not available,<br>Collect the parcels by others }<br>{ Satisfied, Slow delivery, Package damaged, Not received } |