This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2021.3100397, IEEE Transactions on Industrial Informatics

Memory-assistant Collaborative Language Understanding for Artificial Intelligence of Things

Ming Yan, Cen Chen, Jiawei Du, Xi Peng, *Member, IEEE*, Joey Tianyi Zhou, Zeng Zeng, *Senior Member, IEEE*

***Abstract*—Artificial intelligence shows promising efforts in collaborating the language models with the artificial intelligence of things (AIoT), promoting the edging intelligence on natural language understanding. To adapt to the limited computational resources in AIoT, the large language models (e.g., transformer) are compressed into light-weight models, which always results in poor feature representation and unsatisfactory performance on downstream tasks, especially on those low-resource language understanding tasks. To address the above issues, we propose a method named memory-assistant multi-task learning (MAMT), where an auxiliary memory module is introduced to promote multi-task learning, which serves as a surrogate of target domain representation and performs instance-level weighted multi-task learning. More importantly, our MAMT module is in a plug-and-play fashion. Thus, researchers can plug in it to conduct collaborative training and plug it out for AIoT model inference without extra computation burdens. Experiments demonstrate that MAMT significantly improves the performance of light-weight transformer models and show its superiority over the state-of-the-arts on eight GLUE sub-tasks.**

***Index Terms*—AIoT, natural language understanding, neural networks, auxiliary memory, multi-task learning.**

I. Introduction

BENEFITING from the large corpora and millions of parameters, large transformer-based language models have achieves lots of success in various natural language understanding (NLU) tasks [1]. However, the tremendous parameters and high computation cost of the transformer model greatly hinder its application to the low-resourced artificial intelligence of things (AIoT), which mainly equipped low clock frequency computing (i.e., megahertz) and limited memory space (i.e., kilobytes or megabytes) [2], [3]. Fortunately, the model compression for transformer models provides a feasible solution for the resource-constrained edging devices such as model distillation, pruning, and quantization [4]. Although these compression methods is able to obtain light-weight transformer models (i.e., MobileBERT [5], SqueezeBERT [6]) through discarding unimportant features and neural structures, they cannot achieve desirable performance in complex real-world scenarios. This issue is propagated and exaggerated in downstream language understanding tasks. Thus, how to improve the performance of light-weight transformers on language understanding tasks is crucial in AIoT landing. Multi-task learning [7] is one of the popular approaches to improve the performance of a light-weight transformer through leveraging similar auxiliary tasks to learn a task-agnostic feature representation. However, directly combining all tasks to conduct multi-task learning is not an optimal solution as it will lead to a biased representation for the data-rich domains and ignore the target domain with sparse data. Especially for the low-resource tasks of natural language understanding, e.g., RTE task of GLUE benchmark [8], this issue will become more serious with limited data than the rich data task. Therefore, it is important to conduct a discriminative learning on different source tasks rather than treat all the tasks equally.

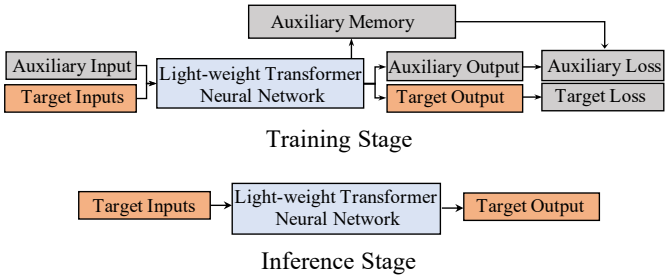

Fig. 1. The block diagram of memory-assistant multi-task learning (MAMT). The auxiliary memory and auxiliary task training modules can be plugged in for discriminative multi-task learning in the training stage and plugged out for inference without extra computing cost.

In this paper, we propose a novel method, named memory-assistant multi-task learning (MAMT), to collaborate among NLU tasks for AIoT. Our MAMT follows a plug-and-play fashion that auxiliary-task module and auxiliary memory module can be unplugged in model deployment for the AIoT scenarios (Fig. 1). Different from the traditional multi-task learning paradigm that directly combines and randomly mixes all training tasks and feeds into the transformer model, our MAMT employs an auxiliary memory to enhance the multi-task learning and conducts instance-level discriminative multi-task learning from the auxiliary task samples. Specifically, the auxiliary memory module of MAMT is a surrogate of feature distribution for the target task, which stores the target domain features and updates stored features in model training. Then, the auxiliary memory computes the similarity between the auxiliary task features to its stored target features. Finally, MAMT adopts the similarity score as the weight of the

This work was supported in part by Singapore A*STAR AME Programmatic Funding under Grant A18A1b0045, in part by the National Key R&D Program of China under Grant 2020YFB1406702. (Corresponding author: Cen Chen and Zeng Zeng, e-mail: chenc@i2r.a-star.edu.sg, zengz@i2r.a-star.edu.sg)

Ming Yan and Jiawei Du and Joey Tianyi Zhou are with IHPC, Agency for Science Technology and Research, Singapore, 138668, Singapore

Cen Chen and Zeng Zeng are with I2R, Agency for Science Technology and Research, Singapore, 138668, Singapore

Xi Peng is with Computer Science Department, Sichuan University, Chengdu, 610065, China.

Manuscript received March 26, 2021; revised June 25, 2021.

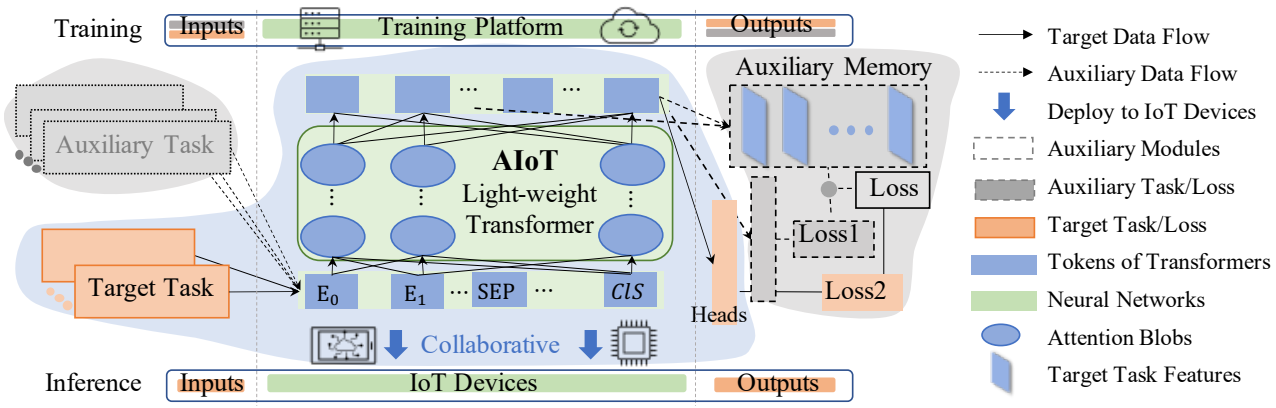

Fig. 2. The overview of memory-assistant multi-task learning (MAMT). AIoT light-weight transformer is an optimized transformer model for collaborative language understanding in AIoTs. The gray auxiliary memory module serves as a surrogate of feature distribution for target task. All the grey background modules can be unplugged in model reference.

auxiliary task in multi-task learning, and collaborates with light-weight transformer model training for learning a target-task-oriented feature. Moreover, our MAMT is independent of backbone models. Its performance improves with an advance of light-weight backbone. Fig. 2 shows an overview of our proposed method in AIoT scenarios. Here we conclude our contributions as follows:

- MAMT augments the multi-task learning with discriminative learning on source tasks, which helps light-weight transformer models conduct a discriminative learning for source tasks on instance-level.
- MAMT introduces an auxiliary memory module to compute the similarity between different source tasks to the target task on feature space. The auxiliary memory module is in a plug-and-play fashion without incurring extra computing budgets in the inference stage.
- MAMT significantly improves the performance of light-weight transformers in diverse natural language understanding tasks, and shows its superior performance on eight GLUE sub-tasks.

The rest of this paper is organized as follows. In Section II, we review related works in artificial intelligent of things, multi-task learning, and natural language understanding. Then, we present the methodology in Section III, and experiment results in Section IV, respectively. We also give a ablation study, parameter analysis in Section V, then conclude the work in Section VI.

II. Related Works

A. Artificial Intelligent of Things

Artificial intelligence of things (AIoT) incorporates the internet of things with artificial intelligence, which enhances machine intelligence capability on data processing and analysis [9]. Nowadays, edge devices spring up into our living world with volume data, which far exceeds the general computing capability of traditional cloud-based artificial intelligence. AIoT is a new IoT era by sinking the artificial intelligent analysis from cloud to edge [10], [11]. [2] provides a successful application for AIoT, which employs machine learning to conserve position confidentiality of roaming PBSs users. LACC greatly improves the traditional greedy approach, incorporated with linear integer programming module, which provides a high-through communication solution for intelligent transportation system [12]. According to the infrastructure, AIoT works can be divided into hardware-based methods and algorithm-based methods. In the hardware-based methods, some researchers optimize memory migration schemes, which improve IO energy consumption and memory footprint usage to collaborate artificial neural networks with IoTs [13]. In algorithm-based methods, GRTT presents an efficient routing solution for workforce monitoring which greatly saved the energy consuming [14]. But, few research works apply the light-weight transformer model to collaborative language understanding in AIoT.

B. Multi-task Learning

Multi-task learning is a learning paradigm that aims to learn a general feature representation for performance boosting [7]. Depending on the model parameter sharing paradigms, multi-task learning can be categorized into hard-parameter sharing and soft-parameter sharing. The hard-parameter paradigm shares backbones, which learns the generic feature with the mixed tasks [15]. For example, MT-DNN employs a transformer model that performs multi-task learning by the joint classification and regression tasks [16]. Similarly, MMM [17] collaborates transformer model with multi-task learning on question answering task. In contrast, the soft-parameter paradigm conducts discriminate learning with individual backbones. MMoE [18] performs the soft regularization by a gate function that controls the multi-task features pass-through to output classifier. Overall, the hard-parameter paradigm shares backbones, which learns the generic feature with the mixed tasks. The soft-parameter paradigm conducts discriminate learning with individual backbones. However, few works combine those two learning paradigms into one framework that conducts generic feature learning with a shared backbone and conducts discriminative learning to avoid shifting. This combination learning paradigm is urgent for AIoT, which

employs a shared light-weight transformer model saving computing resources and employs discriminative learning to avoid learning issues on low-resource training data.

C. Natural Language Understanding

Natural language understanding (NLU) is a sub-task of natural language processing researches on the capability of language understanding in a machine [19]. To evaluate the machine language understanding capability, a general language evaluation benchmark (GLUE) is proposed with various NLU tasks: sentiment analysis, textual similarity, recognizing textual entailment, and natural language inference [8]. Recently, the transformer-based language model made a breakthrough in GLUE tasks with a large superiority over previous methods [1], [20]. The language models, with larger training corpus and parameters, are benefited with more knowledge and higher learning capability. Consequently, later research works pushed the language model sizes sharply increased from million-level to trillion-level in the short two years, and the training corpus as well as increased from GBs to TBs [21]. Those huge parameters greatly hinder its applications to source-limited AIoT scenarios. Some light-weight transformer works aim to optimize the model size and reduce resource reliance [4], [5]. However, those light-weight transformers mainly focus on model squeezing in the inference stage, which neglects improving performance in the training stage.

III. Methodology

First, we show the overall pipeline of MAMT in Fig. 3. Our proposed memory-assistant multi-task (MAMT) learning method mainly contains three parts: the light-weight transformer backbone, auxiliary memory module, and multi-task loss.

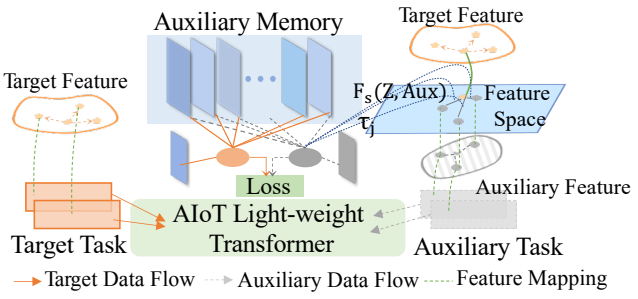

Fig. 3. Overall Pipeline of MAMT. The orange color modules are related to the target task, and the gray color modules are related to the source task.

To fit into the AIoT devices with limited computing resources, we choose the compressed language model as the backbone (MobileBERT [5]), which can help our model gain general knowledge from the universal corpus. The auxiliary memory is proposed to measure the similarity between the auxiliary task to the target task. The similarity score is used to construct weights for different tasks to promote those relevant tasks that brings in the improvement and avoid the undesirable impact from irrelevant auxiliary tasks. The weights are finally used to construct the following multi-task learning loss function.

$$\ell(X_i, Y_i) = \begin{cases} \ell_t(X_i, Y_i), \text{ if } X_i \text{ from target tasks,} \\ \tau_i \ell_a(X_i, Y_i), \text{ if } X_i \text{ from auxiliary tasks,} \end{cases} \quad (1)$$

where, ℓ_t is the target loss, ℓ_a is the auxiliary task loss, and τ_i is the weight of auxiliary task loss. Different from the traditional multi-task learning that treats all tasks equally, our MAMT conducts discriminative learning among the auxiliary tasks by assigning task-related weights. Next, we are going to show how to get τ_i in Equation 1.

A. Auxiliary Memory

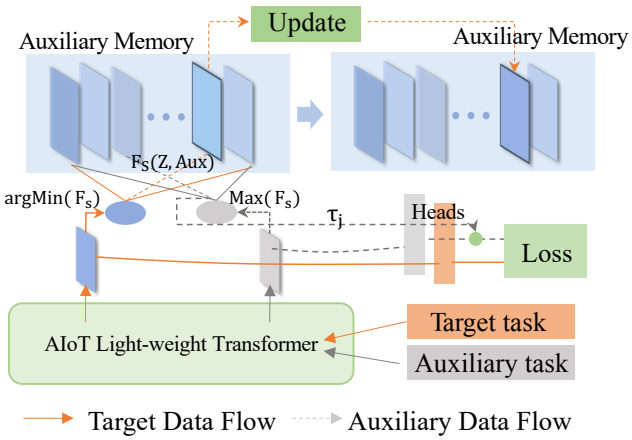

Fig. 4. Auxiliary Memory. The auxiliary memory updates features from target task, and computes similarity scores for auxiliary tasks.

Fig. 4 summarizes the auxiliary memory module in MAMT. The auxiliary memory is first initialized randomly, and then the feature with the lowest similarity score is updated with a new target feature. Here, we define the similarity score between the input feature to auxiliary memory feature in Equation 2:

$$F_s(Z_i, Aux_j) = \frac{Z_i^\top Aux_j}{\max(||Z_i||_2, ||Aux_j||_2)}, \quad (2)$$

where Z_i denotes the feature of ith input example, and Aux_j denotes the jth feature in auxiliary memory.

Note that different strategies are applied in updating features for target and auxiliary tasks. For the target tasks, we first determine which feature should be updated based on the following equation,

$$j^* = \underset{j}{\operatorname{argmin}}(F_s(Z_i, Aux_j)), \ j \in \mathcal{J}, \quad (3)$$

where j^* denotes the index of selected feature in auxiliary memory from the whole index set $\mathcal{J}$. Then, we replace the selected feature Aux_{j^*} with a target task feature Z_i. In this way, auxiliary memory continuously updates its features along with model training to align with the target domain. Thus, the auxiliary memory can function as a surrogate of target feature distribution with a cheap storage cost. To increase the representation capability of auxiliary memory, we also introduce a random feature replacement to avoid the auxiliary memory converge to local minimum optimization.

Different from target tasks update, there has no feature updating in auxiliary task learning. The first step is to construct the similarity score between auxiliary task features to auxiliary memory. To avoid the instability in optimizing the auxiliary task, we further introduce a minimal threshold ϕ. The similarity score is defined as follows,

$$S_j = mean_j(F_s(Z_i, Aux_j)),\ j \in \mathcal{J}, \tag{4}$$

Then, the weight on auxiliary task loss τ_i is computed as follows,

$$\tau_i = \min\{\phi, S_j\}. \tag{5}$$

Our MAMT augments the vanilla multi-task learning with these weights and promotes the features learned through multi-task learning close to the target domain. In addition, the auxiliary memory is a plug-and-play module in MAMT, and it is removable in AIoT deployment. Thus the target task inference computing cost and latency will remain the same.

B. Optimization

In this section, we summarize all the optimization steps in Algorithm 1. For model deployment, MAMT only conducts the feedforward parts (i.e., lines 6-7) with an appropriate downstream head. In other words, all auxiliary memory updating (lines 8-16) and backpropagation (lines 17-33) can be removed in the AIoT model inference stage. The target task training and auxiliary task training are presented in lines 4-14 and lines 16-21, respectively. In MAMT, we first initialize our backbone model with the light-weight transformer model (i.e., MobileBERT), and randomly initialize the auxiliary memory module as well as the memory updating rate ψ. Our memory module is initialized with the features of target domain. Lastly, we initialize the data sampling policy with the sampling ratio P_d to be a ratio between auxiliary data size and target data size.

Next, we introduce the training procedure of MAMT. The first step is to sample the training data from the initialized task distribution P_{mt}, and then the data batch is fed into the transformer model for feature encoding. The following learning procedure is separated for the target task and the auxiliary task.

In the target task learning part, MAMT contains target task learning and auxiliary memory updating. As the MAMT learning (lines 4-14) is a dynamic system that the feature representation as well as changes with the training iteration (more details in discussion subsection V-C). An intuitive updating strategy that keeping the auxiliary memory module consistent with the target domain and replace the feature with the lowest similarity score with a new target feature. However, this updating strategy may result in the auxiliary memory module only keeping the most similar target domain features. To address this issue, a random updating policy (lines 7-12) is introduced to prevent the auxiliary memory module from being stuck in the local minimum.

In the auxiliary task learning part, a weighted back-propagation with the similarity score is adopted. Thus, our MAMT conducts the weighted multi-task learning (line 24) at

Algorithm 1: Memory-assistant Multi-task Learning

Initialize: Backbone M with pretrained weights ;
Memory module random updating rate ψ ;
Auxiliary memory module Aux randomly ;
Data sampling policy $P_d(X_t, Y_t; X_a, Y_a)$;
input : Target domain samples (X_t, Y_t);
Source domain samples (X_a, Y_a);
output : Fine-tuned model M.

1 **while** *sample training batch* (x_i, y_i) *based on the sampling ratio* P_d **do**
2 encode inputs X_i with backbone M ;
3 $Z_i = f(W_m, x_i)$;
4 **if** $x_i \in X_t$ **then**
5 select target domain feature ;
6 $j^* = \text{argmin}_j(F_s(Z_i, Aux_j))$;
7 **if** *random*$() \leq \psi$ **then**
8 update auxiliary memory ;
9 $Aux_{j^*} \leftarrow Z_i$;
10 **else**
11 randomly update Aux_j with Z_i ;
12 **end**
13 compute loss ℓ_t;
14 $\ell_t(X_i, Y_i)$, ;
15 **else**
16 compute similarity score ;
17 $S_j = mean_j(F_s(Z_i, Aux_j))$;
18 compute weight ;
19 $\tau_i = \min\{\phi, S_j\}$;
20 compute auxiliary loss ;
21 $\tau_i \ell_a(X_i, Y_i)$;
22 **end**
23 **if** *do target finetuning* **then**
24 update data sampling policy P_d ;
25 **end**
26 total loss ;
27 $\ell = \ell_t(X_i, Y_i) + \tau_i \ell_a(X_i, Y_i)$;
28 update gradient ;
29 $W_m \leftarrow W_m + \frac{\partial \ell}{\partial W_m}$.
30 **end**

instance-level rather than domain-level. Different from traditional multi-task learning that puts the whole target domain in fine-tuning step at one time, our MAMT conducts target fine-tuning by gradually increasing the sampling ratio P_d of target domain tasks. The sampling ratio P_d is defined as follows,

$$P_d = \begin{cases} \dfrac{N(X_d)}{N(X_t + X_a)}, & \text{if } X_d \text{ from auxiliary tasks,} \\ \dfrac{N(X_d) + \upsilon N(X_a)}{N(X_t + X_a)}, & \text{if } X_d \text{ from target task,} \end{cases} \tag{6}$$

where $N(X)$ denotes the number of data points in X, and υ is a ratio between the current epoch index to the total epoch size. From Equation 6, the sampling ratio P_d of auxiliary task has no change, and the target task sampling ratio increases with the model training process. More in-depth analysis please refer

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2021.3100397, IEEE Transactions on Industrial Informatics

to our ablation study in subsection V-A.

IV. Experiments

We conduct extensive experiments to evaluate the performance of our memory-assistant multi-task (MAMT) on nine tasks of GLUE benchmark [8]. We compare MAMT with different light-weight transformer methods including TinyBERT [4], DistillBERT [22], BERT-of-Theseus [23] and MobileBERT [5], as well as the BERT [1] on base-size version.

A. Dataset

This section gives a brief introduction of our dataset: general language evaluation benchmark (GLUE). As the inputs of out model are tokens, we evaluate different sub-tasks under different token lengths. An overview of GLUE token length distribution is provided in Fig. 5.

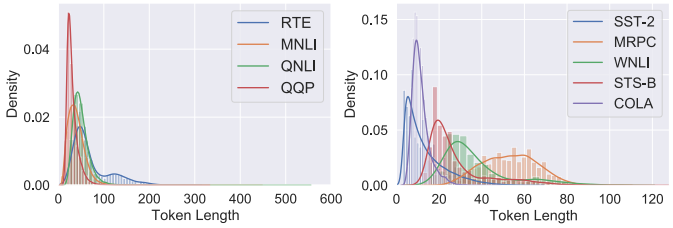

Fig. 5. Token length distributions of GLUE.

From the token length distributions, we can observe that RTE, MNLI, QNLI, and QQP have a longer token length than the rest tasks. Moreover, the tasks with shorter token length have a similar distribution, while the longer token length task has a diverse distribution. In general, most sample length of GLUE tasks is 128 that makes the default token length set to 128 in our experiments.

B. Experiment Setting

For demonstrating the versatility of MAMT, we apply the memory-assistant multi-task learning to light-weight transformer (MobileBERT [5]) and none compressed transformer (BERT [1]) as the backbones to be MAMT (MobileBERT) and MAMT (BERT-Base). For light-weight transformer models, we compare our MAMT with several recent proposed edging/AIoT optimized transformer models: MobileBERT [5], TinyBERT [4], DistillBERT [22], and BERT-of-Theseus [23]. Besides, we also compare with two GLUE baselines: OpenAI GPT [5] and BiLSTM+ELMo+Attn [24]. To further verify the generalization of our MAMT, we evaluate our MAMT with the backbone BERT [1] on the base-size version.

The learning rates are set between $3E-5$ to $5E-5$ in our nine evaluation tasks. Concretely, we set the learning rate to $5E-5$ for the small-size CoLA (8.5K) and $3E-5$ for the large-size MNLI (393K) and QNLI (108K). The training iteration and batch size are also variable depending on the data size. Following the work of MobileBERT [5] and TinyBERT [4], we evaluate our MAMT on nine GLUE datasets with default GLUE metrics (Accuracy, F1, Pearson correlation, and Mathew correlation) [8]. Specifically, we employ the accuracy for classification tasks (SST-2, MNLI-m, MNLI-mm, QNLI, RTE), and Mathew correlation for CoLA, Person Correlation for STS-B, and the F1 for MRPC, QQP. All the experiments are evaluated by the official GLUE [8] evaluation server.

C. Results

Experimental results are summarized in Table I. From the results, we can observe that our MAMT (BERT-Base) achieves the best performances over eight benchmark tasks with 1.8% than backbone BERT-Base. Meanwhile, our MAMT (MobileBERT) outperforms all light-weight transformer baselines that are optimized mainly for edge computing. The main reason is that our MAMT is a discriminative multi-task learning method, which helps the light-weight transformer to learn a generalized feature from auxiliary tasks. Moreover, compared to the distillation-based light-weight transformers (DistillBERT, BERT-of-Theseus, and TinyBERT) with a high-resource-consuming teacher model in their training stage, our MAMT is in a plug-and-play manner that only introduces an auxiliary memory module. From the comparison of average scores, we can find that most SOTA light-weight transformer models have comparable performance on the GLUE benchmark, and our MAMT still boosts the vanilla MobileBERT with 1.3% improvement.

Overall, our MAMT works better on small training sample tasks than the large ones. For example, the small training sample RTE improves 8.5% than the backbone MobileBERT. In contrast, the large training sample tasks QQP and QNLI only improved 0.4% and 0.1% on MobileBERT, and the MAMT (BERT) even have no improvements on MNLI-m. We think that the large auxiliary task helps MAMT to learn better features and improve the performance as well. Nevertheless, the model performance improvement is limited on the large size tasks. Besides, we also observe our MAMT boost 8.5% and 10.2% on the RTE than the vanilla backbones with auxiliary task MNLI. The results show the similar tasks in MAMT can boost the performance on small-size dataset. This similarity reflects the training labels that MNLI (entailment, contradiction) overlap with RTE(entailment and not entailment). In contrast, less similar tasks go contrary to small-size MPRC collaborates trained on large MNLI only got 0.2% improvement on MAMT (BERT), and its performance even declines 0.6% than backbone MobileBERT. One hypothesis is dissimilar between RTE (sentence similar classification) to MNLI (entailment task). More discussion about the auxiliary task and target task is provided in subsection V-B.

V. Discussions

A. Ablation Study

In this section, we conduct the ablation study on vanilla multi-task, auxiliary memory module, and finetuning ratio. To explore the effect of different components, we perform sequential multiple task learning (transfer learning, TL) and parallel multiple task learning (multi-task learning, MT) on GLUE tasks. Here, we want to point out that vanilla multi-task learning (MT) is a simplified version of MAMT without the auxiliary memory module. At last, we study the effectiveness

TABLE I
THE GLUE SCORES ON **TEST** SET USING GLUE EVALUATION SERVER.

Names	Params	CoLA 8.5k	SST-2 67K	MRPC 3.7K	STS-B 5.7K	QQP 364K	MNLI-m 393K	MNLI-mm 393K	QNLI 108K	RTE 2.5K	AVG
BiLSTM+ELMo+Attn[5]	-	33.6	90.4	84.4	72.3	63.1	74.1	74.5	78.8	58.9	70.0
DistillBERT[22]	*66M*	49.0	92.5	86.9	81.3	70.1	82.6	81.3	88.9	58.4	76.8
OpenAI GPT[24]	*109M*	47.2	93.1	87.7	84.8	70.1	80.7	80.6	87.2	69.1	77.8
BERT-of-Theseus [23]	*66M*	47.8	92.2	87.6	85.6	71.6	82.4	82.1	89.6	66.2	78.3
TinyBERT-6 [4]	*15M*	51.1	93.1	87.3	83.7	71.6	84.6	83.2	90.4	70.0	79.4
SqueezeBERT[6]	*51M*	46.5	91.4	87.8	86.7	80.3	82.0	81.1	90.1	73.2	79.9
MobileBERT [5]	*25M*	50.5	92.8	**88.8**	84.4	70.2	83.3	82.6	90.6	66.2	78.8
MAMT(MobileBERT)	*25M*	**51.0**	**93.5**	88.2	**85.2**	**70.6**	**84.0**	**83.1**	**90.7**	**74.7**	**80.1**
BERT-Base [1]	*110M*	52.1	93.5	88.9	85.8	71.2	**84.6**	83.4	90.5	66.4	79.6
MAMT(BERT-Base)	*110M*	**54.6**	**93.6**	**89.1**	**86.4**	**71.9**	**84.6**	**84.3**	**91.4**	**76.8**	**81.4**

of finetuning for MAMT by increasing the sampling ratio on the target task. The backbone is MobileBERT, and the experiment results are summarized in Table II. The task in bracket is the auxiliary task. "w/o-AF" denotes MAMT(MNLI) without adjustable task ratio in finetuning, and "w-AF" denotes MAMT with adjustable task ratio in finetuning.

TABLE II
ABLATION STUDY OF MAMT.

Names	CoLA	SST-2	MRPC$^+$	QQP$^+$
TL(QQP)	53.16	91.05	90.28/86.02	-
TL(MNLI)	53.13	92.08	**91.42**/87.5	83.45/87.08
TL(QNLI)	52.86	90.94	89.96/85.78	83.43/87.03
MT(MNLI)	30.46	90.94	83.71/75.49	82.94/83.12
MAMT(w/o-AF)	52.07	90.82	89.69/85.29	82.65/82.86
MAMT(w-AF)	**57.11**	**91.62**	91.29/**87.75**	**83.69/83.59**

"$-$" denotes not available for the setting. "$+$" denotes F1/Accuracy.

From the results shown in Table II, we can observe that transfer learning-based methods (TL) achieves better performance than vanilla multi-task learning (MT). In tasks of SST-2, QQP, MT has $1-2\%$ decline than TL models. The worst performance of MT appears in CoLA, which declines more than 20%. We guess the reason is the domain gap, MNLI is a two sentences entailment task, and CoLA is a single-sentence linguistic acceptability task, which caused the MT performance decline. Similarly, the domain gaps also reflect on the TL models. For example, CoLA got better performance on the auxiliary task of QQP, while the SST-2 got its better performance with auxiliary task MNLI.

To make a fair comparison, we unify the MNLI as the auxiliary task in MT and MAMT. More auxiliary task comparisons on MAMT will discuss in the following subsection. From the results of Table II we can see that MAMT with auxiliary memory can effectively improve the performance of vanilla multi-task learning. These results indicate that simply mixing more data in multi-task learning not always a good solution. Moreover, our MAMT (w-AF) with adjustable target sampling further improved the MAMT performance on all datasets. The CoLA task achieves the best improvement with 5% than MAMT(w/o-AF). So, we can conclude the adjustable target sampling plays an important role in MAMT. Compared to TL's learning paradigm, our MAMT finetuning works similar to TL finetuning by feeding target data samples. Moreover, our MAMT finetuning exists in multi-task learning, which adjusts higher sampling possibilities on the target task.

B. Parameter Sensitivity Study

In this section, we study the parameter sensitivity of MAMT from three aspects: auxiliary tasks, feature extraction, and feature size of the auxiliary memory module.

1) Auxiliary Tasks: To promote the performance of light-weight transform in AIoT scenarios, we study the auxiliary task selection in MAMT and conduct evaluations on the development set of GLUE tasks: MNLI-m, MNLI-mm, STS-B, and QQP. All the results are reported in Table III.

TABLE III
DIFFERENT AUXILIARY TASKS.

Target(Auxiliary)	MNLI-m/mm	STS-B$^+$	QQP$^+$
MAMT(QQP)	82.56/83.17	**88.62/88.41**	-
MAMT(SNLI)	**83.69/83.59**	87.77/87.71	86.67/87.75
MAMT(QNLI)	83.15/83.60	87.45/87.15	86.93/90.03
MAMT(SST-2)	83.36 /83.29	87.74/87.51	**87.73/90.67**

"$-$" denotes not available for the setting. "$+$" denotes F1/Accuracy.

From the result shown in Table III, we can observe MAMT achieves the best performance among different auxiliary tasks, and the similar auxiliary task helps more for the target task. For example, the MNLI-m and MNLI-mm both achieved their best performance (83.69 and 83.59) on the auxiliary task SNLI. The reason is that SNLI and MNLI are both entailment tasks even with the same labels (entailment, neutral, and contradiction). Although the QNLI is the entailment task, its performance slightly inferior to SNLI. The reason is that QNLI has a different data distribution from SNLI with only two labels (entailment and not entailment). Furthermore, similar results also appear in STS-B (sentence similarity comparison) to QQP (question pair), in which performance achieves almost 1% promotion than training with other auxiliary tasks. So, we can conclude that similar auxiliary tasks make more contributions for MAMT in collaborative language understanding.

2) Feature Size in Auxiliary Memory: The auxiliary memory is a core module of MAMT, a surrogate representation of target domain feature distribution. In detail, the auxiliary memory module consists of extracted features from the encoder layers of a transformer. The intuitive idea is that the auxiliary memory with more target features will get a better representation capability. To verify this idea, we study the impact of feature size experiments in the auxiliary memory module by varying feature size in the range $\{N_b, 3N_b, 5N_b, 7N_b\}$, where N_b is the batch size.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TII.2021.3100397, IEEE Transactions on Industrial Informatics

TABLE IV
DIFFERENT FEATURE SIZES OF AUXILIARY MEMORY.

Target(Auxiliary)	N_b	$3N_b$	$5N_b$	$7N_b$
MRPC(STS-B)	88.24	90.97	**91.04**	88.24
RTE(MNLI)	72.92	73.65	73.65	**74.74**
SST-2(MNLI)	**92.09**	91.63	91.63	91.28
MNLI-m(SNLI)	82.80	82.66	**82.90**	82.76
MNLI-mm(SNLI)	82.88	**83.03**	82.95	82.67

"N_b" denotes batch size.

From the numeric results in Table IV, we observe that more features stored in auxiliary memory do not always gain more improvements in MAMT. The performance of RTE grows with feature size raising. While the performance of task SST-2 dramatically declines with feature size increasing. We thought the probable reason for the differences is that the target task RTE and its auxiliary task MNLI are all entailment classification tasks. MAMT helps small dataset RTE achieve progressive increments with feature sizes. In contrast, the semantic classification task SST-2 has a domain gap with auxiliary task MNLI, which gets a performance decline with feature size increment. So, we conclude that the feature size of auxiliary memory has a nonlinear relationship with MAMT performance in GLUE tasks. In other words, the feature size of the auxiliary memory in MAMT is a hyperparameter in collaborative language understanding on AIoT.

3) Feature Extraction: MAMT conducts discriminative multi-task learning on the auxiliary tasks, where the auxiliary task weights are computed by the similarity between the auxiliary task feature to the auxiliary memory. From the view of feature representation, the different layer of our transformer neural network has different semantic representation. In this part, we make further exploration on feature selection in MAMT.

TABLE V
FEATURE EXTRACTION FORM DIFFERENT LAYERS.

Target(Auxiliary)	1 Layer	4 Layers	8 Layers	12 Layers
RTE(MNLI)	**74.01**	**74.01**	73.29	72.20
SST-2(MNLI)	91.51	**92.20**	91.74	92.09
MRPC(STS-B)	90.56	91.52	90.62	**91.67**
MNLI-m(SNLI)	83.35	83.37	83.33	**83.48**
MNLI-mm(SNLI)	**83.58**	83.43	83.31	83.41

From the results of Table V, we can observe RTE and MNLI-mm, features are extracted from the last transformer layer, get better performance than feature extraction from more layers. In contrast, MRPC and MNLI-m gain more promotion with more layers feature extraction. Different feature extraction policy does not play a crucial impact on large size tasks, i.e., the performance of MNLI-m and MNLI-mm have no signification affection. The feature extraction from different layers impacts their performance to the small size tasks (MRPC and RTE). These results indicate that the small-size language understanding task needs more attention in AIoT.

C. Task Similarity

To study the similarity between the target task to auxiliary tasks, we compare different target tasks (RTE and SST-2) with the same auxiliary task (MNLI). We also explore the similarity changes with features extracted from different layers on the same target task (MRPC) and auxiliary task (STS-B). Fig. 6 shows the similarity comparison of MAMT on

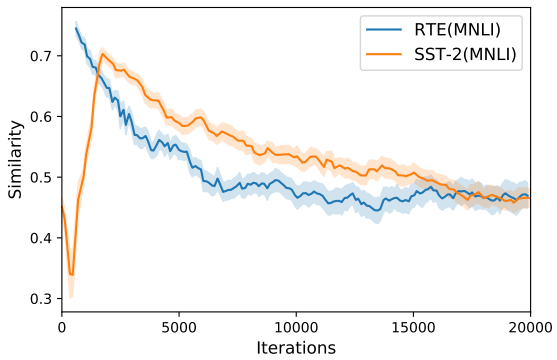

Fig. 6. Similarity comparison of MAMT on different target tasks. The task name outside the bracket denotes the target task, and the task name in the bracket denotes the auxiliary task. Take the "RTE(MNLI)" for example, our MAMT is trained on MNLI (auxiliary task) and RTE (target task).

different target tasks. As the auxiliary task MNLI and target task RTE are both entailment tasks, the similarity of two tasks in RTE(MNLI) has high values around 0.9 at the training start stage. However, with the training iteration, our MAMT learns domain discrepancy between RTE to MNLI, reflecting the curve is similarity decline. In contrast, the target task SST-2 has a domain difference with auxiliary task MNLI, which causes their feature similarity as low as 0.3. We think that the backbone model, pretrained light-weight language model, has no discriminate capability to target task and auxiliary task. In the middle iterations, our MAMT mapping two tasks to a generic feature representation, which reflects a growing similarity. As the RTE and SST-2 are different target tasks, our MAMT trends to learn domain-specific task features, and its similarity goes down in the end iterations.

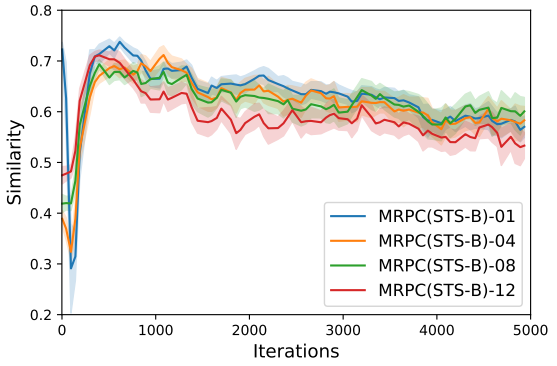

Fig. 7. Similarity comparison of MAMT on different layers feature extraction.

Fig. 7 shows the similarity learning comparison of MPRC(STS-B) on different layer extraction. The main difference exists in the beginning stage. The feature extracted from the first layer holds less discriminative information than the

feature extract from deeper layers, which shows MRPC(STS-B)-01 has a higher similarity than other layers. With training continue, the changing of similarity on MRPC(STS-B) performing a similar way. Moreover, the similarity analysis can help MAMT select the auxiliary tasks.

VI. Conclusion

In this paper, we propose a memory-assistant multi-task (MAMT) method for collaborative language understanding in AIoT scenarios. The MAMT is plug-and-play that can be plugged into the model by multi-task training and be plugged out for inference without extra computation burdens. Moreover, our MAMT employs an auxiliary memory to conduct instance-level discriminative multi-task learning on the source samples, promoting the performance of the light-weight transformer models. The ablation study shows the auxiliary memory is important to discriminative multi-task on auxiliary tasks. We also discuss the hyper-meter sensitivity and visualize the internal similarity changes in its training process.

Our work provides a promising solution to improve the performance of light-weight transformers, especially for the scenarios with low resources in both data and computation. Our method improves the model generalization capability for the low-resource in data through learning from data-rich domains and simultaneously reduces computation cost. Moreover, we consider the discriminative information in multi-task learning to avoid undesirable impacts from irrelevant tasks. Experimental results on the GLUE benchmark demonstrate the efficiency of our memory-assistant multi-task learning method for diverse language understanding tasks. With the more widespread applications of low-resource AIoT, language intelligence to interact with the human is urgently required for AIoTs. In our future work, we will explore how to apply MAMT in other applications (e.g., question answering, phone assistant, etc.). Meanwhile, more advanced light-weight transformers in MAMT will be investigated.

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding." Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186. [Online]. Available: https://www.aclweb.org/anthology/N19-1423

[2] A. K. Sangaiah, D. V. Medhane, T. Han, M. S. Hossain, and G. Muhammad, "Enforcing position-based confidentiality with machine learning paradigm through mobile edge computing in real-time industrial informatics," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 7, pp. 4189–4196, 2019.

[3] R. C. Luo, Chih-Chen Yih, and Kuo Lan Su, "Multisensor fusion and integration: approaches, applications, and future research directions," *IEEE Sensors Journal*, vol. 2, no. 2, pp. 107–119, 2002.

[4] X. Jiao, Y. Yin, L. Shang, X. Jiang, X. Chen, L. Li, F. Wang, and Q. Liu, "TinyBERT: Distilling BERT for natural language understanding," in *EMNLP 2020*. Association for Computational Linguistics, Nov. 2020, pp. 4163–4174.

[5] Z. Sun, H. Yu, X. Song, R. Liu, Y. Yang, and D. Zhou, "MobileBERT: a compact task-agnostic BERT for resource-limited devices," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Jul. 2020, pp. 2158–2170.

[6] F. Iandola, A. Shaw, R. Krishna, and K. Keutzer, "SqueezeBERT: What can computer vision teach NLP about efficient neural networks?" in *Workshop on Simple and Efficient Natural Language Processing*. Association for Computational Linguistics, Nov. 2020, pp. 124–135.

[7] S. Vandenhende, S. Georgoulis, W. Van Gansbeke, M. Proesmans, D. Dai, and L. Van Gool, "Multi-task learning for dense prediction tasks: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2021.

[8] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman, "GLUE: A multi-task benchmark and analysis platform for natural language understanding," in *EMNLP*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 353–355.

[9] A. Arsénio, H. Serra, R. Francisco, F. Nabais, J. Andrade, and E. Serrano, "Internet of intelligent things: Bringing artificial intelligence into things and communication networks," in *Inter-cooperative collective intelligence: Techniques and applications*. Springer, 2014, pp. 1–37.

[10] W. Lawless, R. Mittu, D. Sofge, I. S. Moskowitz, and S. Russell, *Artificial intelligence for the internet of everything*. Academic Press, 2019.

[11] Y. Li, C. Chen, M. Duan, Z. Zeng, and K. Li, "Attention-aware encoder–decoder neural networks for heterogeneous graphs of things," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 4, pp. 2890–2898, 2021.

[12] A. K. Sangaiah, J. S. Ramamoorthi, J. J. P. C. Rodrigues, M. A. Rahman, G. Muhammad, and M. Alrashoud, "Laccvov: Linear adaptive congestion control with optimization of data dissemination model in vehicle-to-vehicle communication," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–10, 2020.

[13] H. Sun, L. Chen, X. Hao, C. Liu, and M. Ni, "An energy-efficient and fast scheme for hybrid storage class memory in an aiot terminal system," *Electronics*, vol. 9, no. 6, 2020. [Online]. Available: https://www.mdpi.com/2079-9292/9/6/1013

[14] A. K. Sangaiah, A. S. Rostami, A. A. R. Hosseinabadi, M. B. Shareh, A. Javadpour, S. H. Bargh, and M. M. Hassan, "Energy-aware geographic routing for real-time workforce monitoring in industrial informatics," *IEEE Internet of Things Journal*, vol. 8, no. 12, pp. 9753–9762, 2021.

[15] X. Zou, L. Zhou, K. Li, A. Ouyang, and C. Chen, "Multi-task cascade deep convolutional neural networks for large-scale commodity recognition," *Neural Computing and Applications*, vol. 32, no. 10, pp. 5633–5647, 2020.

[16] X. Liu, P. He, W. Chen, and J. Gao, "Multi-task deep neural networks for natural language understanding," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, Jul. 2019, pp. 4487–4496. [Online]. Available: https://www.aclweb.org/anthology/P19-1441

[17] D. Jin, S. Gao, J.-Y. Kao, T. Chung, and D. Hakkani-tur, "Mmm: Multi-stage multi-task learning for multi-choice reading comprehension," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, 2020, pp. 8010–8017.

[18] J. Ma, Z. Zhao, X. Yi, J. Chen, L. Hong, and E. H. Chi, "Modeling task relationships in multi-task learning with multi-gate mixture-of-experts," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining*, ser. KDD '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 1930–1939.

[19] Y. Zheng, G. Chen, and M. Huang, "Out-of-domain detection for natural language understanding in dialog systems," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 1198–1209, 2020.

[20] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020. [Online]. Available: http://jmlr.org/papers/v21/20-074.html

[21] K. Heafield, P. Koehn, and A. Lavie, "Language model rest costs and space-efficient storage," in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Jeju Island, Korea: Association for Computational Linguistics, Jul. 2012, pp. 1169–1178. [Online]. Available: https://aclanthology.org/D12-1107

[22] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," 2020.

[23] C. Xu, W. Zhou, T. Ge, F. Wei, and M. Zhou, "BERT-of-theseus: Compressing BERT by progressive module replacing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Nov. 2020, pp. 7859–7869.

[24] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018.