AdapterShare: Task Correlation Modeling with Adapter Differentiation

Zhi Chen[1]*, **Bei Chen**[2], **Lu Chen**[1], **Kai Yu**[1], **Jian-Guang Lou**[2]
[1]X-LANCE Lab, Department of Computer Science and Engineering
MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
[2]Microsoft Research Asia
{zhenchi713, chenlusz, kai.yu}@sjtu.edu.cn, {beichen, jlou}@microsoft.com

Abstract

Thanks to the development of pre-trained language models, multitask learning (MTL) methods have achieved great success in natural language understanding. However, current MTL methods pay more attention to task selection or model design to fuse as much knowledge as possible, while the intrinsic task correlation is often neglected. It is important to learn sharing strategies among multiple tasks rather than sharing everything. In this paper, we propose AdapterShare, an adapter differentiation method to explicitly model task correlation among multiple tasks. AdapterShare is automatically learned based on the gradients on tiny held-out validation data. Compared to single-task learning and fully shared MTL methods, our proposed method obtains obvious performance improvements. Compared to the existing MTL method AdapterFusion, AdapterShare achieves an absolute average improvement of 1.90 points on five dialogue understanding tasks and 2.33 points on NLU tasks. Our implementation is available at https://github.com/microsoft/ContextualSP.

1 Introduction

With the development of transformer-based pre-trained language models (PLMs), natural language understanding (NLU) has made great progress as a downstream task. There are two main ways to leverage PLMs in NLU tasks. One is the fine-tuning method, which updates the pre-trained language model directly on a target task. The other one is *adapters* (Rebuffi et al., 2017; Houlsby et al., 2019), which introduces a small number of task-specific parameters on a fixed PLM. When training on the target task, only the introduced parameters are updated. Compared to the fine-tuning method, adapter is memory-efficient, since the introduced parameters are much less than those of the PLM. In this paper, we focus on the approach using adapters.

*Work done during internship at Microsoft Research Asia.

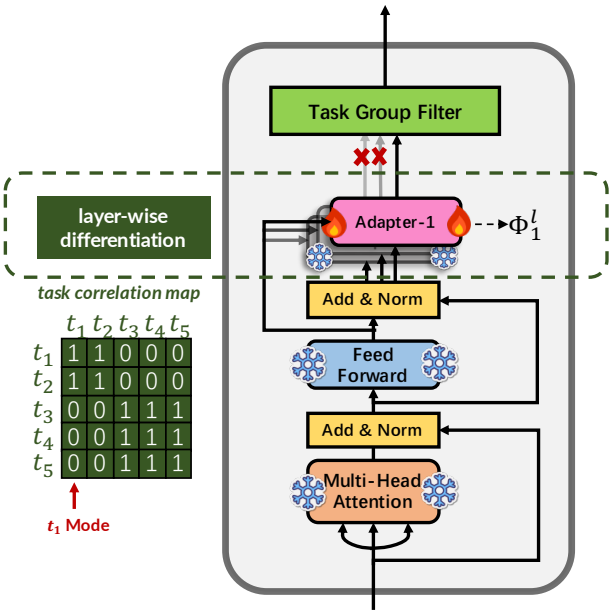


Figure 1: The architecture of the adapters with task correlation modeling method.

To transfer the knowledge of different tasks, Stickland and Murray (2019) proposed a multitask learning (MTL) method to update the weights of a shared adapter using the weighting of the objective functions of all target tasks. The shared adapter captures the common structure underlying all the target tasks. This is a typical multitask learning method based on an implicit assumption that all tasks benefit from each other, where all parameters of the adapter are shared during multitask training. In other words, the task correlation has not been modeled in the traditional MTL method. In this paper, we propose a robust adapter differentiation method, called AdapterShare, to model the correlation of all target tasks explicitly. As shown in Figure 1, during the multitask learning process, the sharing strategy of adapter at each PLM layer is automatically learned according to the adapter gradients on small-scale held-out validation data. The learned sharing strategy can be regarded as a discrete task correlation map.

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 10645 - 10651
December 7-11, 2022

The closest work is AdapterFusion (Pfeiffer et al., 2021), which is a two-stage learning method. The first stage is to train task-wise adapters separately, and the second stage is to fuse all task-wise adapters with attention mechanism for each target task. The two-stage method is sensitive to the initialization of attention weights. Once there are two tasks that hurt each other, it is hard to assign zero to the corresponding adapter using soft attention mechanism. Compared to AdapterFusion, our proposed AdapterShare learns all the adapters and their task correlation simultaneously. We adopt a discrete format to represent task correlation, where at each PLM layer, every two tasks either share the adapter (1 in the task correlation map) or not (0 in the task correlation map).

2 Problem Statement

As discussed, the existing multitask learning methods tend to share all parameters. It assumes that all target tasks benefit from each other. However, in practice, it can be detrimental to assume correlation in a set of tasks and simply put them together for learning (Bonilla et al., 2007). In this paper, we propose an approach to learn task correlation automatically. The task correlation indicates that all the target tasks are clustered into several task groups. The tasks in the same task group share the parameters. We maintain the task correlation map at the granularity of each transformer layer of pretrained language models. With *adapters* training strategy, the learning process can be formalized as:

$$\Phi_i \longleftarrow \text{argmin}(L_{\Phi_i}(D_i; \Theta_0, \Phi_i)), \quad (1)$$

where Θ_0 is initialized parameters of PLM, Φ_i is the adapter parameters of i-th task t_i, D_i is the annotated training samples of i-th task and $L_{\Phi_i}(\cdot)$ is the loss function of target task. The *adapters* consists of adapter networks at all PLM layers:

$$\Phi_i = \{\Phi_i^1, \Phi_i^2, \ldots, \Phi_i^L\}, \quad (2)$$

where L is the layer number of PLM and Φ_i^l is the adapter parameters of l-th PLM layer for the task group containing task t_i. As mentioned, the task correlation is at layer granularity. If task t_j is in the same task group as task t_i at l-th layer, the adapter parameters are shared between these two tasks, which means $\Phi_i^l = \Phi_j^l$. The task group at l-th PLM layer is defined by layer-wise task correlation map M^l. For example, as shown in Figure 1, there are two task groups: $G_1^l = G_2^l = \{t_1, t_2\}$, $G_3^l = G_4^l = G_5^l = \{t_3, t_4, t_5\}$ according to the task correlation map M^l, where $M^l(i,j) = 1$ means t_i and t_j is in the same group at l-th layer. In the next section, we will introduce how to learn the layer-wise task correlation map.

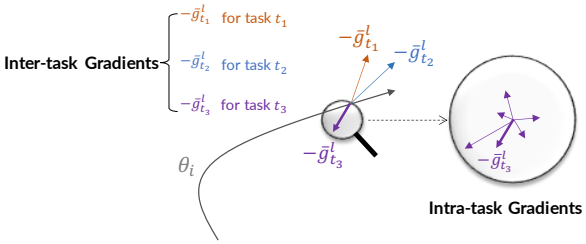


Figure 2: Calculated inter-task and intra-task gradients on tiny task-wise held-out validation sets.

3 AdapterShare

In this section, we first introduce the adopted task correlation learning method in general. Then we reveal the problem of existing neural differentiation algorithm and improve it in our proposed task correlation learning algorithm, AdapterShare. Note that in the following, all learnable parameters are adapters, while the parameters of PLM are fixed.

3.1 Adapter Differentiation

We model task correlation in a discrete format. The discrete task correlation map divides all the target tasks into several task groups. The tasks in the same task group benefit from each other. The main challenge is how to quantify the effects of two different tasks. Inspired by the parameter differentiation method (Wang and Zhang, 2021), we leverage **interference degree** as the effect metric. The interference degree of two tasks is the negative value of the inter-task gradient cosine similarity on the shared parameters. The inter-task gradient is calculated on tiny held-out validation data, which contains validation samples of all tasks. Formally, the interference degree of a task group is:

$$\mathcal{I}(\Phi_i^l; G_i^l) = \max_{t_i, t_j \in G_i^l} -\frac{\bar{\mathbf{g}}_{t_i}^l \cdot \bar{\mathbf{g}}_{t_j}^l}{\|\bar{\mathbf{g}}_{t_i}^l\| * \|\bar{\mathbf{g}}_{t_j}^l\|}, \quad (3)$$

$$\bar{\mathbf{g}}_{t_i}^l = \nabla L_{\Phi_i^l}(H_i; \Theta_0, \Phi_i^l), \quad (4)$$

where $\bar{\mathbf{g}}_{t_i}^l$ is the inter-task gradient of shared adapter in task group G_i^l, calculated on the held-out validation data H_i of task t_i. The inter-task gradient $\bar{\mathbf{g}}_{t_i}^l$ is accumulated gradient of all the samples in the held-out validation data of task t_i. If the

Algorithm 1: Task Correlation Learning

Set all the elements of task correlation maps to one: $\{M^l\}_{l=1}^{L}$.
Initialize the adapter parameters: $\{\Phi_i^l\}_{l=1}^{L}$, where $\Phi_0^l = \cdots = \Phi_N^l$.
// Prepare the data for N tasks
Training dataset: $\{D_i\}_{i=1}^{N}$.
Held-out validation dataset: $\{H_i\}_{i=1}^{N}$.
// Training process of each epoch
for *i in 1,2,...,N* **do**
  1.Sample a mini-batch b_i from D_i.
  2.Switch the adapters into i-th task mode according to $\{M^l\}_{l=1}^{L}$: Φ_i.
  3.Compute loss as Eq. 1 and Update Φ_i.
// Detect adapter differentiation
for *l in 1,2,...,L* **do**
  Task group set: $\{G_i^l\}_{i=1}^{N}$
  for G_i *in* $\{G_i^l\}_{i=1}^{N}$ **do**
    for t_i *in* G_i **do**
      // Consistency of intra-task gradients
      4.Split H_i into $H_{i,0}$ and $H_{i,1}$.
      5.Calculate $\bar{\mathbf{g}}_{t_i,0}^l$ and $\bar{\mathbf{g}}_{t_i,1}^l$ as Eq. 4.
      6.Calculate $\mathcal{C}(\Phi_i^l)$ as Eq. 5.
    if *all* $\mathcal{C}(\Phi_i^l) > \alpha$ **then**
      7.Calculate $\bar{\mathbf{g}}_{t_i}^l$ as Eq. 6.
      8.Calculate $\mathcal{I}(\Phi_i^l; G_i^l)$ as Eq.3.
      if *any* $\mathcal{I}(\Phi_i^l; G_i^l) > 0$ **then**
        9. Adapter differentiation.
        10.Update M^l.

interference degree $\mathcal{I}(\Phi_i^l; G_i^l) > 0$, it indicates that there are at least two tasks in this task group that have conflicting optimum directions. For example, as shown in Figure 2, $\bar{\mathbf{g}}_{t_1}^l$ and $\bar{\mathbf{g}}_{t_2}^l$ have similar global optimum directions, while $\bar{\mathbf{g}}_{t_3}^l$ has the opposite direction to the other two tasks. It suggests that t_3 may hinder the other two tasks t_1 and t_2. These three tasks need to be divided into two different groups: $G_1^l = G_2^l = \{t_1, t_2\}$ and $G_3^l = \{t_3\}$. The dividing process is named **adapter differentiation**, where one task group is split into two subgroups. In detail, adapter differentiation has three steps: **1)** The two tasks with the highest interference degree are taken as representatives and put into two different subgroups; **2)** Every other task in the current task group is compared with these two representatives and added to the subgroup with the lower interference degree; **3)** The parameters of two differentiated adapters are copied from the original adapter. The elements in the task correlation map M^l will change from 1 to 0, if two tasks belong to different task groups.

At the beginning of the training process, we set all elements of the task correlation map to 1, which means that all adapter parameters are shared among all tasks. Then, we periodically calculate the interference degree of the current task groups to activate the adapter differentiation operation when the interference degree is greater than 0. Once adapter differentiation starts, the task correlation map will be permanently changed.

Corpora	#Sample	$\mathbf{I}_{(\text{Token})}$	$\mathbf{I}_{(\text{Turn})}$	$\mathbf{O}_{(\text{Token})}$	Task
SAMSUM (2019)	14732	104.95	11.2	20.31	DS
TASK (2019)	2205	34.92	2.8	10.84	DC
BANK77 (2020)	12081	21.64	1	3.14	ID
RES8K (2020)	15270	14.44	1	3.38	SF
WOZ2.0 (2017)	7608	78.96	4.6	1.30	DST

Table 1: Statistics of five dialogue understanding datasets. $\mathbf{I}_{(\text{Token})}$ and $\mathbf{I}_{(\text{Turn})}$ mean the average length of the split tokens and the average turns of the input dialogue content. $\mathbf{O}_{(\text{Token})}$ means the average length of the split tokens of the task-specific output.

Corpora	#Train	#Dev.	#Test	#Label	Task
WNLI (2012)	634	71	146	2	NLI
RTE (2018)	2500	276	3000	2	NLI
CoLA (2019)	8500	1000	1000	2	ACC
SST-2 (2013)	67000	872	1800	2	SEN
STSB (2017)	7000	1500	1400	1	SIM

Table 2: Statistics of five natural language understanding datasets.

3.2 Avoiding Over-Differentiation

So far, we have introduced the basic adapter differentiation method for learning task correlation. However, in practice, we find a problem called over-differentiation: the basic adapter differentiation method has an unstable training process, in which the update of the task correlation map is irreversible. At the beginning of the training process, the shared adapter parameters are fragile and the **inter-task** gradients have a big bias on the held-out validation data. Thus, the adapter differentiation operation needs to be cautious. In our proposed AdapterShare, we add another line of defense to activate the differentiation. We have to make sure that the inter-task gradient is trusted. As shown in Figure 2, we can see that each inter-task gradient is accumulated by **intra-task** gradients, while the intra-task gradients vary within a task.

To alleviate this issue, we randomly split all the intra-task gradients into two groups and calculate the accumulated intra-task gradients of these two groups: $\bar{\mathbf{g}}_{t_i,0}^l$ and $\bar{\mathbf{g}}_{t_i,1}^l$. Then, we use their cosine

DU Tasks (T5)	Methods			
	ST	MT	AdapterFusion	AdapterShare
SAMSUM (R-L)	48.80	47.78	47.36	**49.12**
TASK (BLEU)	88.45	89.54	89.92	**90.20**
BANK77 (ACC.)	91.58	89.25	91.10	**93.15**
REST8K (F1)	97.28	96.41	95.93	**97.58**
WOZ2.0 (JGA)	91.25	90.70	89.12	**92.89**
OVERALL	83.47	82.74	82.69	**84.59**

Table 3: Results on five dialogue understanding tasks with the backbone T5.

NLU Tasks (BERT)	Methods			
	ST	MT	AdapterFusion	AdapterShare
WNLI (ACC.)	56.34	**61.97**	56.33	**61.97**
RTE (ACC.)	66.06	77.61	70.75	**77.62**
COLA (MCC.)	58.02	59.06	60.23	**60.64**
SST-2 (ACC.)	**93.12**	92.66	**93.12**	92.77
STSB (Spearman)	88.78	89.28	**89.88**	88.96
OVERALL	72.46	76.12	74.06	**76.39**

Table 4: Results on five natural language understanding tasks with the backbone BERT.

similarity as the consistency of inter-task gradient, calculated as:

$$\mathcal{C}(\Phi_i^l) = \frac{\bar{\mathbf{g}}_{t_i,0}^l \cdot \bar{\mathbf{g}}_{t_i,1}^l}{\|\bar{\mathbf{g}}_{t_i,0}^l\| * \|\bar{\mathbf{g}}_{t_i,1}^l\|}. \quad (5)$$

The adapter differentiation on a task group can be activated only when all tasks in this task group have consistency values greater than the threshold α. The inter-task gradient of task t_i is equal to the sum of two accumulated intra-task gradients, formalized as:

$$\bar{\mathbf{g}}_{t_i}^l = \bar{\mathbf{g}}_{t_i,0}^l + \bar{\mathbf{g}}_{t_i,1}^l. \quad (6)$$

To distinct with basic adapter differentiation method, we name the improved method as robust adapter differentiation. The details of task correlation learning are shown in Algorithm 1.

4 Experiments

4.1 Datasets

We evaluate our proposed AdapterShare on five dialog understanding (DU) datasets (shown in Table 1) and five natural language understanding (NLU) datasets (shown in Table 2). There are five different dialog understanding tasks in DU datasets. DS, DC, ID, SF and DST represent dialogue summary, dialogue completion, intent detection, slot filling and dialogue state tracking, respectively. Five NLU datasets are chosen from GLUE benchmark, spanning four different NLU tasks. NLI, ACC, SEN and SIM indicate natural language inferencing, acceptability, sentiment and similarity, respectively.

4.2 Experimental Setup

In order to investigate the proposed AdapterShare training method, we compare it with ST, MT and AdapterFusion. ST trains a separate adapter for each target task. MT trains the adapters on all the target tasks (Stickland and Murray, 2019). AdapterFusion fuses the separated ST adapters on the target task with attention mechanism.

As described in Su et al. (2022) and Chen et al. (2022), the dialogue understanding tasks can be formulated as a unified sequence-to-sequence generation task. For five DU tasks, we leverage T5-base model (Raffel et al., 2020) as the backbone of the generation model. For five NLU tasks, we implement all the experiments based on the released code by Liu et al. (2019). The backbone of NLU tasks is BERT-large (Kenton and Toutanova, 2019). The *adapters* is implemented based on AdapterHub (Pfeiffer et al., 2020), where the pre-trained language models are inherited from HuggingFace library (Wolf et al., 2019). We set the threshold of intra-task consistency α to 0.707 ($\cos(\pi/4)$). The learning rate is 1e-5. We conduct all the experiments on V100 GPU with 16G memory. All the metrics are the higher, the better.

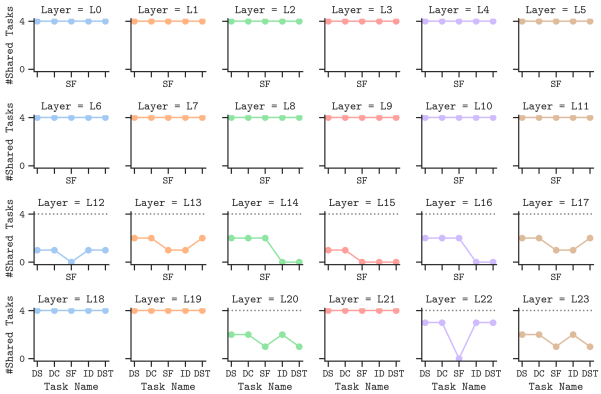


Figure 3: Differentiated adapters on 24 transformer layers of T5. X-axis represents the task name. Y-axis represents the number of shared tasks.

4.3 Results

The proposed AdapterShare adopts a robust adapter differentiation method to learn task correlation. As shown in Table 3, we can find that the proposed AdapterShare can get the best performance than the baselines. Compared with the single-task method, AdapterFusion method can not obtain any performance gain in encoder-decoder setup. In the encoder-only situation, AdapterFusion method can achieve the best performance on two of five tasks, as shown in Table 4. Compared with the single-task method, it actually gets obvious improvements, which is consistent with the original conclusion (Pfeiffer et al., 2021). However, in encoder-only setup, our proposed AdapterShare can still obtain the best performance on three of five tasks and get the best overall score. MT method shares all the parameters among all the tasks. In dialog understanding tasks, the overall score of ST is better than MT, which indicates that there are some tasks hurt by other tasks. The final results on DU tasks further indicate our proposed AdapterShare, which learns the task correlation map, is more efficient than independent training (ST) and complete-sharing methods. The final differentiation architecture on T5 is shown in Figure 3. The four shared tasks mean that all five tasks are shared with each other in the corresponding layer. We can see that the adapter differentiation happens only on T5 decoder side and all the adapters on encoder are shared. This phenomenon is interesting. As we know, inputs in all DU tasks are the dialogue context. The encoder module, as the presentation function, is used to represent the dialogue context. Compared with the encoder, the decoder needs to solve different DU tasks, whose outputs are very different. Various DU tasks need to pay attention to different dialogue context areas. For example, the DST task is more inclined to obtain the entity information mentioned by the user, and the intention detection is more inclined to pay attention to user actions.

We also conduct an ablation study to compare robust adapter differentiation method with basic differentiation method on dialog understanding tasks. The performance curves on the development datasets are shown in Appendix A. It shows that the training process of the robust adapter differentiation method is more stable than the basic method. The metrics of robust method on DU tasks are also higher than the basic differentiation method.

5 Conclusion

In this paper, we propose a robust adapter differentiation method to automatically learn task correlation in the multitask learning setting. On both encoder-decoder and encoder-only PLMs, our proposed method can achieve exciting performance gains compared to the separated training, complete-sharing and AdapterFusion methods. In future work, we will try our method in the domain transfer area, which is a more general scenario than multitask learning.

Limitations

There are two main limitations in this paper. The first one is about the scale of multiple tasks. In the experiments, there are five tasks on dialogue understanding area and natural language understanding area. It is unsure whether the proposed method works in a large-scale task learning setup. The second one is the implicit assumption included in our proposed method that the effect of two tasks are mutual, where one benefits/hurts the other means that the other also benefits/hurts itself. There is currently no evidence for the validity of this assumption. We leave these explorations for future work.

Ethical Considerations

As our adapter differentiation methods are validated on the existing datasets, we follow the original copyright statements of 10 datasets. All claims in this paper are based on the experimental results. No demographic or identity characteristics information is used in this paper.

References

Edwin V Bonilla, Kian Chai, and Christopher Williams. 2007. Multi-task gaussian process prediction. *Advances in neural information processing systems*, 20.

Inigo Casanueva, Tadas Temcinas, Daniela Gerz, Matthew Henderson, and Ivan Vulic. 2020. Efficient intent detection with dual sentence encoders. *ACL 2020*, page 38.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Zhi Chen, Lu Chen, Bei Chen, Libo Qin, Yuncong Liu, Su Zhu, Jian-Guang Lou, and Kai Yu. 2022. UniDU: Towards a unified generative dialogue understanding framework. In *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 442–455, Edinburgh, UK. Association for Computational Linguistics.

Samuel Coope, Tyler Farghly, Daniela Gerz, Ivan Vulić, and Matthew Henderson. 2020. Span-convert: Few-shot span extraction for dialog with pretrained conversational representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 107–121.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. Samsum corpus: A human-annotated dialogue dataset for abstractive summarization. *EMNLP-IJCNLP 2019*, page 70.

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

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

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth international conference on the principles of knowledge representation and reasoning*.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. Adapterfusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. Adapterhub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020): Systems Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Jun Quan, Deyi Xiong, Bonnie Webber, and Changjian Hu. 2019. Gecor: An end-to-end generative ellipsis and co-reference resolution model for task-oriented dialogue. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4547–4557.

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. *Journal of Machine Learning Research*, 21:1–67.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters. *Advances in neural information processing systems*, 30.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Asa Cooper Stickland and Iain Murray. 2019. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning. In *International Conference on Machine Learning*, pages 5986–5995. PMLR.

Yixuan Su, Lei Shu, Elman Mansimov, Arshit Gupta, Deng Cai, Yi-An Lai, and Yi Zhang. 2022. Multi-task pre-training for plug-and-play task-oriented dialogue system. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4661–4676.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

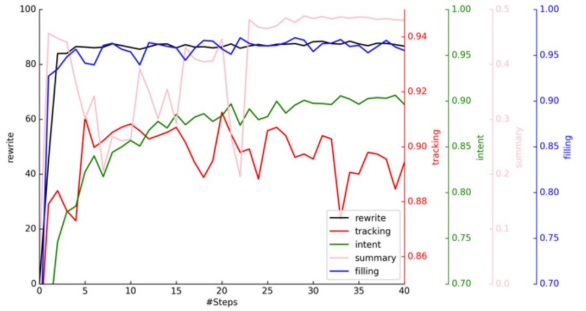
Qian Wang and Jiajun Zhang. 2021. Parameter differentiation based multilingual neural machine translation. *arXiv preprint arXiv:2112.13619*.

Alex Warstadt, Amanpreet Singh, and Samuel Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

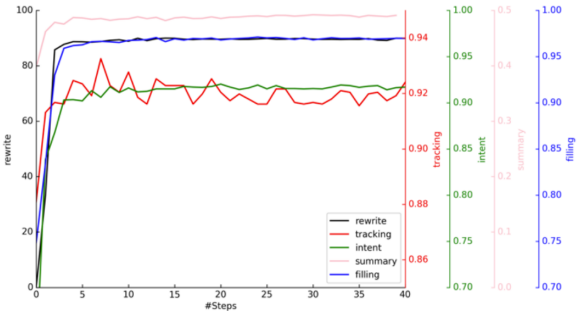
Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gasic, Lina M Rojas Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

A Ablation Study on DU Tasks



(a) Basic adapter differentiation.



(b) Robust adapter differentiation.

Figure 4: The performance curves on five dialogue understanding tasks with (a) basic adapter differentiation and (b) robust adapter differentiation methods.