SUPPLEMENTARY MATERIAL

A PRE-TRAINING DETAILS

A.1 FACTUAL ADAPTER

The pre-trained model is fixed during training and the parameters of the factual adapter are trainable and initialized randomly. The model is trained with cross-entropy loss. To accelerate the training process, we set the max sequence length as 64 as the average sequence length of T-REx-rc is only 22.8. We train the model for 5 epochs using a batch size of 128. We use AdamW to optimize our models with the initial learning rate of 2e-5. We train the model with 4 16G NVIDIA V100 GPUs.

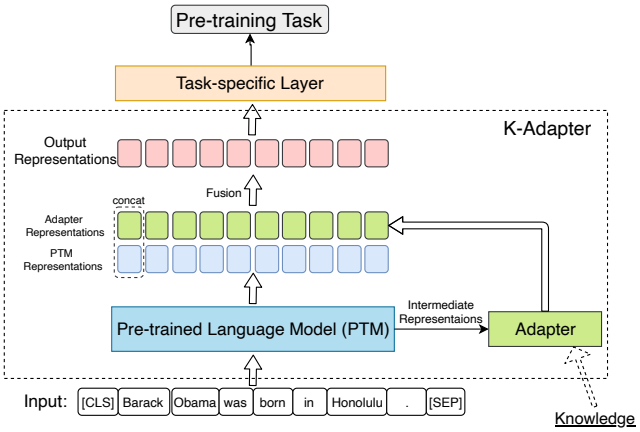

Figure 3: An overview of our K-ADAPTER to inject specific knowledge by training a knowledge-specific adapter on the pre-training task.

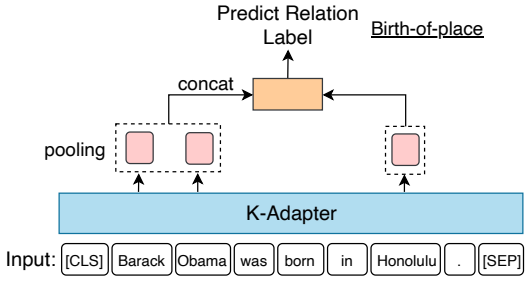

Figure 4: An example of using relation classification as a pre-training task to inject knowledge into K-ADAPTER: given *"Barack Obama was born in Honolulu"*, and then predicts the relationship between "Barack Obama" and "Honolulu" is "Birth-of-place".

A.2 LINGUISTIC ADAPTER

Same as the training process of the factual adapter, the pre-trained model is fixed during training and the parameters of the linguistic adapter are trainable and initialized randomly. The model is trained with BCEWithLogits loss. We set the max sequence length as 128. We train the model for 10 epochs using a batch size of 256. We use AdamW with the initial learning rate of 1e-5. We train the model with 4 16G NVIDIA V100 GPUs.

B APPLYING K-ADAPTER ON DOWNSTREAM TASKS

For the downstream tasks, the key point here is the combination of the pre-trained model's representations and adapter's representations, that is to say: leveraging the general information of the

pre-trained model on one hand, and the specific knowledge in the adapter on the other. To use K-ADAPTER for downstream tasks is very simple as shown in Figure 5. Usually, when we use pre-trained language models such as BERT and RoBERTa for downstream tasks, we feed the output features from the pre-trained model into the task-specific layer, and then do the corresponding downstream task. As for the K-ADAPTER, we fine-tune it just like what the orginal BERT or RoBERTa does. We concatenate the output features of the pre-trained model with the features of the adapter, and then feed them to the task-specific layer.

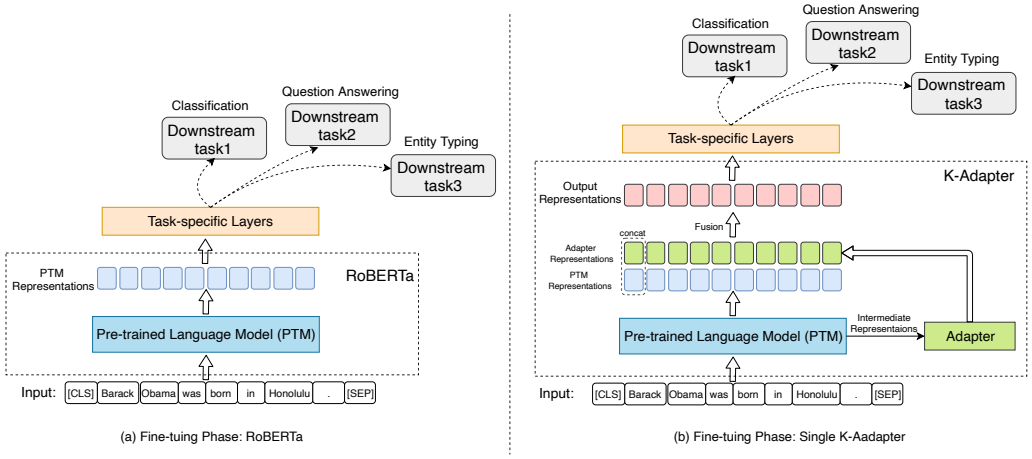

Figure 5: Fine-tuning K-ADAPTER just like what the original RoBERTa or BERT does.

C DATASET STATISTICS

In Table 8, we present the statistics of one relation classification dataset TACRED, and two entity typing datasets OpenEntity and FIGER. In Table 9, we present the statistics of one commonsense QA dataset CosmosQA and two open-domain QA datasets SearchQA and Quasar-T.

Table 8: The statistics of the relation classification dataset TACRED and entity typing datasets, i.e., Open Entity and FIGER.

Dataset	Train	Dev	Test	Relation/Type
TACRED	68,124	22,631	15,509	42
Open Entity	2,000	2,000	2,000	6
FIGER	2,000,000	10,000	563	113

Table 9: The statistics of the question answering datasets, i.e., CosmosQA, SearchQA and Quasar-T.

Dataset	Train	Dev	Test
CosmosQA	25,588	3,000	7,000
SearchQA	99,811	13,893	27,247
Quasar-T	28,496	3,000	3,000

D FINE-TUNING DETAILS AND HYPERPARAMETERS

We implement our experiments using Huggingface[3]. For all fine-tuning experiments, we use AdamW as the optimizer. The parameters of adapters are fixed during the fine-tuning process and the parameters of RoBERTa are trainable and initialized from Huggingface checkpoint. We select the best hyperparameters on the validation set. For all experiments, we set the random seed to be 42 for reproductibility.

D.1 ENTITY TYPING

For Open Entity dataset, we set the max sequence length to be 256 and select the hyperparameters from batch size: {4, 8}, learning rate: {2e-5, 1e-5, 5e-6} and warmup step: {0, 200, 500, 1000, 1200}. For K-ADAPTER (F), the best performance is achieved at batch size=4, lr=5e-6, warmup=500 (it takes about 2 hours to get the best result running on singe 16G P100). For K-ADAPTER (L), the best performance is achieved at batch size=4, lr=5e-6, warmup=1000 (it takes about 2 hours to get the best result running on singe 16G P100). For K-ADAPTER (F+L), the best performance is achieved at batch size=4, lr=5e-6, warmup=1000 (it takes about 3 hours to get the best result running on singe 16G P100). For FIGER dataset, we run experiments on 4 16G P100 for 3 epochs, set the max sequence length to be 256, and select the hyperparameters from batch size: {64, 512, 2048}, learning rate: {2e-5, 1e-5, 5e-6} and warmup step: {0, 200, 500, 1000, 1200}. For K-ADAPTER (F), the best performance is achieved at batch size=2048, lr=5e-6, warmup=500. For K-ADAPTER (L), the best performance is achieved at batch size=2048, lr=5e-6, warmup=200. For K-ADAPTER (F+L), the best performance is achieved at batch size=2048, lr=5e-6, warmup=1000.

D.2 QUESTION ANSWERING

For CosmosQA dataset, we run experiments on one single 16G P100 for 3 epochs, set the max sequence length to be 256, and select the hyperparameters from batch size: {16, 32, 64, 128}, learning rate: {2e-5, 1e-5, 5e-6} and warmup step: {0, 200, 500, 800, 1000}. For K-ADAPTER (F+L) and its ablated models, the best performance is achieved at batch size=64, lr=1e-5, warmup=0 (it takes about 8 hours to get the best result).

For SearchQA dataset, we run experiments on one single 16G P100 for 2 epochs, set the max sequence length to be 128, and select the hyperparameters from batch size: {2, 4, 8, 16}, learning rate: {5e-5, 2e-5, 1e-5, 5e-6} and warmup step: {0, 500, 1000}. For K-ADAPTER (F+L) and its ablated models, the best performance is achieved at batch size=8, lr=5e-6, warmup=0. For Quasar-T dataset, we run experiments on one single 16G P100 for 5 epochs, set the max sequence length to be 256, and select the hyperparameters from batch size: {2, 4, 8, 16}, learning rate: {5e-5, 2e-5, 1e-5, 5e-6} and warmup step: {0, 500, 1000}. For K-ADAPTER (F+L) and its ablated models, the best performance is achieved at batch size=16, lr=1e-5, warmup=0.

D.3 RELATION CLASSIFICATION

For TACRED dataset, we run experiments on 4 16G P100 for 5 epochs, set the max sequence length to be 184, and select the hyperparameters from batch size: {4, 8, 16, 32}, learning rate: {2e-5, 1e-5, 5e-6, 1e-6} and warmup step: {0, 200, 500, 800, 1000, 1200}. For K-ADAPTER (F), the best performance is achieved at batch size=32, lr=1e-5, warmup=500. For K-ADAPTER (L), the best performance is achieved at batch size=32, lr=1e-5, warmup=200. For K-ADAPTER (F+L), the best performance is achieved at batch size=32, lr=5e-6, warmup=1000.

E PROBING EXPERIMENTS

We implement our probing experiments using LAMA[4]. LAMA probe aims to answer cloze-style questions about relational facts, e.g., "Simon Bowman was born in [MASK]". This task requires the language model to predict a distribution over a limited vocabulary to replace [MASK]. When we

[3]https://github.com/huggingface/transformers

[4]https://github.com/facebookresearch/LAMA

infuse knowledge into knowledge-specific adapters, we do not change the original parameters of the pre-trained model and thus do not adopt the masked language model (MLM) as a pre-training task. Therefore, before we conduct probing experiments, we need to add and train a linear layer as the mlm layer for predicting the [MASK] entities. Specifically, we fix all the parameters of K-ADAPTER and only update the parameters of the mlm layer using a masked language modeling (MLM) loss. We adopt the raw WikiText-2 dataset (181M). We train the mlm layer with one single 16G P100 for 2 epochs. We set the max sequence length to be 512, batch size to be 1024 and warmup step to be 0.