Recall and Learn: Fine-tuning Deep Pretrained Language Models with Less Forgetting

Sanyuan Chen[1], **Yutai Hou**[1], **Yiming Cui**[1,2], **Wanxiang Che**[1], **Ting Liu**[1], **Xiangzhan Yu**[1]
[1]School of Computer Science and Technology, Harbin Institute of Technology, China
[2]Joint Laboratory of HIT and iFLYTEK Research (HFL), Beijing, China
{sychen, ythou, ymcui, car, tliu}@ir.hit.edu.cn, yxz@hit.edu.cn

Abstract

Deep pretrained language models have achieved great success in the way of pretraining first and then fine-tuning. But such a sequential transfer learning paradigm often confronts the catastrophic forgetting problem and leads to sub-optimal performance. To fine-tune with less forgetting, we propose a recall and learn mechanism, which adopts the idea of multi-task learning and jointly learns pretraining tasks and downstream tasks. Specifically, we introduce a Pretraining Simulation mechanism to recall the knowledge from pretraining tasks without data, and an Objective Shifting mechanism to focus the learning on downstream tasks gradually. Experiments show that our method achieves state-of-the-art performance on the GLUE benchmark. Our method also enables *BERT-base* to achieve better average performance than directly fine-tuning of *BERT-large*. Further, we provide the open-source RECADAM optimizer, which integrates the proposed mechanisms into Adam optimizer, to facility the NLP community.[1]

1 Introduction

Deep Pretrained Language Models (LMs), such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019), have significantly altered the landscape of Natural Language Processing (NLP), and a wide range of NLP tasks has been promoted by these pretrained language models. These successes are mainly achieved through *Sequential Transfer Learning* (Ruder, 2019): pretrain a language model on large-scale unlabeled data and then adapt it to downstream tasks. The adaptation step is usually conducted in two manners: fine-tuning or freezing pretrained weights. In practice, fine-tuning is adopted more widely due to its flexibility (Phang et al., 2018; Peters et al., 2019; Lan et al., 2020).

[1]https://github.com/Sanyuan-Chen/RecAdam

Despite the great success, sequential transfer learning of deep pretrained LMs is prone to catastrophic forgetting during the adaptation step. *Catastrophic forgetting* is a common problem for sequential transfer learning, and it happens when a model forgets previously learned knowledge and overfits to target domains (McCloskey and Cohen, 1989; Kirkpatrick et al., 2017). To remedy the catastrophic forgetting in transferring deep pretrained LMs, existing efforts mainly explore fine-tuning tricks to forget less. ULMFiT (Howard and Ruder, 2018) introduced discriminative fine-tuning, slanted triangular learning rates, and gradual unfreezing for LMs fine-tuning. Lee et al. (2020) reduced forgetting in BERT fine-tuning by randomly mixing pretrained parameters to a downstream model in a dropout-style.

Instead of learning pretraining tasks and downstream tasks in sequence, *Multi-task Learning* learns both of them simultaneously, thus can inherently avoid the catastrophic forgetting problem. Xue et al. (2019) tackled forgetting in automatic speech recognition by jointly training the model with previous and target tasks. Kirkpatrick et al. (2017) proposed Elastic Weight Consolidation (EWC) to overcome catastrophic forgetting when continuous learning multiple tasks by adopting the multi-task learning paradigm. EWC regularizes new task training by constraining the parameters which are important for previous tasks and adapt more aggressively on other parameters. Thanks to the appealing effects on catastrophic forgetting, EWC has been widely applied in various domains, such as game playing (Ribeiro et al., 2019), neural machine translation (Thompson et al., 2019) and reading comprehension (Xu et al., 2019).

However, these multi-task learning methods cannot be directly applied to the sequential transferring regime of deep pretrained LMs. Firstly, multi-task learning methods require to use the data of pre-

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 7870–7881,
November 16–20, 2020.

training tasks during adaptation, but the pretraining data of LMs is often inaccessible or too large for the adaptation. Secondly, we only care about the downstream task's performance, while multi-task learning also aims to promote performance on pretraining tasks.

In this paper, we propose a recall and learn mechanism to cope with the forgetting problem of fine-tuning the deep pretrained LMs. To achieve this, we take advantage of multi-task learning by adopting LMs pretraining as an auxiliary learning task during fine-tuning. Specifically, we introduce two mechanisms for the two challenges mentioned above, respectively. As for the challenge of data obstacles, we introduce the *Pretraining Simulation* to achieve multi-task learning without accessing to pretraining data. It helps the model recall previously learned knowledge by simulating the pretraining objective using only the pretrained parameters. As for the challenge of learning objective difference, we introduce the *Objective Shifting* to balance new task learning and pretrained knowledge recalling. It allows the model to focus gradually on the new task by shifting the multi-task learning objective to the new task learning.

We also provide Recall Adam (RECADAM) optimizer to integrate the proposed recall and learn mechanism into Adam optimizer (Kingma and Ba, 2015). We release the source code of the RECADAM optimizer implemented in PyTorch (Paszke et al., 2019). It is easy to use and can facilitate the NLP community for better fine-tuning of deep pretrained LMs. Experiments on the GLUE benchmark with the BERT-base model show that the proposed method can significantly outperform the vanilla fine-tuning method. Our method with the BERT-base model can even achieve better average results than directly fine-tuning the BERT-large model. In addition, thanks to the effectiveness of pretrained knowledge recalling, we can initialize the model with random parameters and gain better performance with larger parameter search space than the pretrained initialization. Finally, we achieve state-of-the-art performance on the GLUE benchmark with the ALBERT-xxlarge model.

Our contributions can be summarized as follows: (1) We propose to tackle the catastrophic forgetting problem of fine-tuning the deep pretrained LMs by adopting the idea of multi-task learning and obtain state-of-the-art results on the GLUE benchmark. (2) We propose a recall and learn mechanism with Pretraining Simulation and Objective Shifting to achieve multi-task fine-tuning without data of pretraining tasks. (3) We provide the open-source RECADAM optimizer to facilitate deep pretrained LMs fine-tuning with less forgetting.

2 Background

In this section, we present two transfer learning settings: sequential transfer learning and multi-task learning. They both aim to improve the learning performance by transferring knowledge across multiple tasks, but apply to different scenarios.

2.1 Sequential Transfer Learning

Sequential transfer learning learns source tasks and target tasks in sequence, and transfers knowledge from source tasks to improve the models' performance on target tasks.

It typically consists of two stages: *pretraining* and *adaptation*. During pretraining, the model is trained on source tasks with the loss function Loss_S. During adaptation, the pretrained model is further trained on target tasks with the loss function Loss_T. The standard adaptation methods includes *fine-tuning* and *feature extraction*. Fine-tuning updates all the parameters of the pretrained model, while feature extraction regards the pretrained model as a feature extractor and keeps it fixed during the adaptation phase.

Sequential transfer learning has been widely used recently, and the released deep pretrained LMs have achieved great successes on various NLP tasks (Peters et al., 2018; Devlin et al., 2019; Lan et al., 2020). While the adaptation of the deep pretrained LMs is very efficient, it is prone to *catastrophic forgetting*, where the model forgets previously learned knowledge from source tasks when learning new knowledge from target tasks.

2.2 Multi-task Learning

Multi-task Learning learns multiple tasks simultaneously, and improves the models' performance on all of them by sharing knowledge across these tasks (Caruana, 1997; Ruder, 2017).

Under the multi-task learning paradigm, the model is trained on both source tasks and target tasks with the loss function:

$$\text{Loss}_\text{M} = \lambda \text{Loss}_\text{T} + (1 - \lambda)\text{Loss}_\text{S} \quad (1)$$

where $\lambda \in (0, 1)$ is a hyperparameter balancing these two tasks. It can inherently avoid catastrophic

forgetting because the loss on source tasks Loss_S is always part of the optimization objective.

To overcome catastrophic forgetting (discussed in § 2.1), can we apply the idea of multi-task learning to the adaptation of the deep pretrained LMs? There are two challenges in practice:

1) We cannot get access to the pretraining data to calculate Loss_S during adaptation.

2) The optimization objective of adaptation is Loss_T, while multi-task learning aims to optimize Loss_M, i.e., the weighted sum of Loss_T and Loss_S.

3 Methodology

In this section, we introduce Pretraining Simulation (§ 3.1) and Objective Shifting (§ 3.2) to overcome the two challenges (discussed in § 2.2) respectively. Pretraining Simulation allows the model to learn source tasks without pretraining data, and Objective Shifting allows the model to focus on target tasks gradually. We also introduce the RECADAM optimizer (§ 3.3) to integrate these two mechanisms into the common-used Adam optimizer.

3.1 Pretraining Simulation

As for the first challenge that pretraining data is unavailable, we introduce Pretraining Simulation to approximate the optimization objective of source tasks as a quadratic penalty, which keeps the model parameters close to the pretrained parameters.

Following Elastic Weight Consolidation (EWC; Kirkpatrick et al. 2017; Huszár 2017), we approximate the optimization objective of source tasks with Laplace's Method and assumption of independence among the model parameters. Since EWC requires pretraining data, we further introduce a stronger independence assumption and derive a quadratic penalty, which is independent of the pretraining data. We introduce the detailed derivation process as follows.

From the probabilistic perspective, the learning objective on the source tasks Loss_S would be optimizing the negative log posterior probability of the model parameters θ given data of source tasks D_S:

$$\text{Loss}_\text{S} = -\log p(\theta|D_S)$$

The pretrained parameters θ^* can be assumed as a local minimum of the parameter space, and it satisfies the equation:

$$\theta^* = \arg\min_\theta\{-\log p(\theta|D_S)\}$$

Due to the intractability, the optimization objective $-\log p(\theta|D_S)$ is locally approximated with the Laplace's Method (MacKay, 2003):

$$\begin{aligned}-\log p(\theta|D_S) \approx &-\log p(\theta^*|D_S) \\ &+\frac{1}{2}(\theta-\theta^*)^\top H(\theta^*)(\theta-\theta^*)\end{aligned}$$

where $H(\theta^*)$ is the Hessian matrix of the optimization objective w.r.t. θ and evaluated at θ^*. $-\log p(\theta^*|D_S)$ is a constant term w.r.t. θ, and it can be ignored during optimization.

Since the pretrained model convergences on the source tasks, $H(\theta^*)$ can be approximated with the empirical Fisher information matrix $F(\theta^*)$ (Martens, 2014):

$$F(\theta^*) = \mathbb{E}_{x\sim D_S}[\nabla_\theta \log p_\theta(x) \nabla_\theta \log p_\theta(x)^\top|_{\theta=\theta^*}]$$

$$H(\theta^*) \approx NF(\theta^*) + H_{prior}(\theta^*)$$

where N is the number of i. i. d. observations in D_S, $H_{prior}(\theta^*)$ is the Hessian matrix of the negative log prior probability $-\log p(\theta)$.

Because of the computational intractability, EWC approximate $H(\theta^*)$ by using the diagonal of $F(\theta^*)$ and ignoring the prior Hessian matrix $H_{prior}(\theta^*)$:

$$(\theta-\theta^*)^\top H(\theta^*)(\theta-\theta^*) \approx N\sum_i F_i(\theta_i-\theta_i^*)^2$$

where F_i is the corresponding diagonal Fisher information value of the model parameter θ_i.

Since the pretraining data is unavailable, we further approximate $H(\theta^*)$ with a stronger assumption that each diagonal Fisher information value F_i is independent of the corresponding parameter θ_i:

$$(\theta-\theta^*)^\top H(\theta^*)(\theta-\theta^*) \approx NF\sum_i(\theta_i-\theta_i^*)^2$$

The final approximated optimization objective of the source tasks is the quadratic penalty between the model parameters and the pretrained parameters:

$$\begin{aligned}\text{Loss}_\text{S} &= -\log p(\theta|D_S) \\ &\approx \frac{1}{2}(\theta-\theta^*)^\top H(\theta^*)(\theta-\theta^*) \\ &\approx \frac{1}{2}(\theta-\theta^*)^\top (NF(\theta^*)+H_{prior}(\theta^*))(\theta-\theta^*) \\ &\approx \frac{1}{2}N\sum_i F_i(\theta_i-\theta_i^*)^2 \\ &\approx \frac{1}{2}NF\sum_i(\theta_i-\theta_i^*)^2 \\ &= \frac{1}{2}\gamma\sum_i(\theta_i-\theta_i^*)^2\end{aligned}$$

where $\frac{1}{2}\gamma$ is the coefficient of the quadratic penalty.

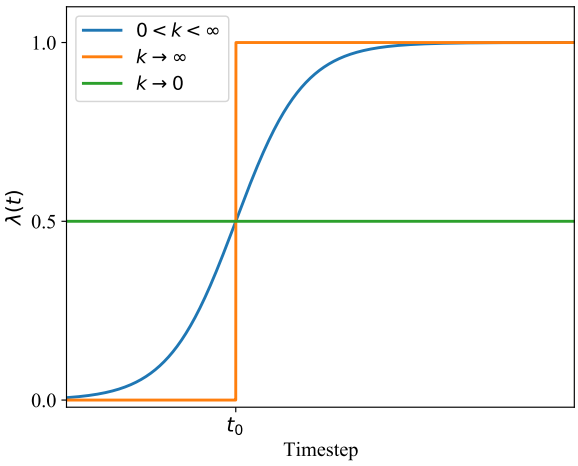


Figure 1: Objective Shifting: we replace the coefficient λ with the annealing function $\lambda(t)$. Fine-tuning and multi-task learning can be regarded as the special cases ($k \to \infty$ and $k \to 0$) of our method.

3.2 Objective Shifting

As for the second challenge that the optimization objective of multi-task learning is inconsistent with adaptation, we introduce Objective Shifting to allow the objective function to gradually shift to Loss_T with the annealing coefficient.

We replace the coefficient λ in the optimization objective of multi-task learning (as shown in Eq. 1) with the annealing function $\lambda(t)$, where t refers to the update timesteps during fine-tuning. The loss function of our method is set to multi-task learning with annealing coefficient:

$$\text{Loss} = \lambda(t)\text{Loss}_\text{T} + (1 - \lambda(t))\text{Loss}_\text{S}$$

Specifically, to better balance the multi-task learning and fine-tuning, $\lambda(t)$ is calculated as the sigmoid annealing function (Bowman et al., 2016; Kiperwasser and Ballesteros, 2018):

$$\lambda(t) = \frac{1}{1 + \exp(-k \cdot (t - t_0))}$$

where k and t_0 are the hyperparameters controlling the annealing rate and timesteps.

As shown in Figure 1, at the beginning of the training process, the model mainly learns general knowledge by focusing more on pretraining tasks. As training progress, the model gradually focuses on target tasks and learns more target-specific knowledge while recalling the knowledge of pretraining tasks. At the end of the training process, the model completely focuses on target tasks, and the final optimization objective is Loss_T.

Fine-tuning and multi-task learning can be regarded as special cases of our method. When $k \to \infty$, our method can be regarded as fine-tuning. The model firstly gets pretrained on source tasks with the Loss_S, then learns the target tasks with the Loss_T. When $k \to 0$, $\lambda(t)$ is a constant function, then our method can be regarded as the multi-task learning. The model learns source tasks and target tasks simultaneously with the loss function $\frac{1}{2}(\text{Loss}_\text{T} + \text{Loss}_\text{S})$.

3.3 RecAdam Optimizer

Adam optimizer (Kingma and Ba, 2015) is commonly used for fine-tuning the deep pretrained LMs. We introduce Recall Adam (RecAdam) optimizer to integrate the quadratic penalty and the annealing coefficient, which are the core factors of the Pretraining Simulation (§ 3.1) and Objective Shifting (§ 3.2) mechanisms respectively, by decoupling them from the gradient updates in Adam optimizer.

Loshchilov and Hutter (2019) observed that L2 regularization and weight decay are not identical for adaptive gradient algorithms such as Adam, and confirmed the proposed AdamW optimizer based on decoupled weight decay could substantially improve Adam's performance in both theoretical and empirical way.

Similarly, it is necessary to decouple the quadratic penalty and the annealing coefficient when fine-tuning the pretrained LMs with Adam optimizer. Otherwise, both the quadratic penalty and annealing coefficient would be adapted by the gradient update rules, resulting in different magnitudes of the quadratic penalty among the model's weights.

The comparison between Adam and RecAdam are shown in Algorithm 1, where SetScheduleMultiplier(t) (Line 11) refers to the procedure (e.g. warm-up technique) to get the scaling factor of the step size.

Line 6 of Algorithm 1 shows how we implement the quadratic penalty and annealing coefficient with the vanilla Adam optimizer. The weighted sum of the gradient of target task objective function $\nabla f(\theta)$ and the gradient of the quadratic penalty $\gamma(\theta - \theta^*)$ get adapted by the gradient update rules, which derives to inequivalent magnitudes of the quadratic penalty among the model's weights, e.g. the weights that tend to have larger gradients $\nabla f(\theta)$ would have the larger second moment v and be

Algorithm 1 Adam and RecAdam

given initial learning rate $\alpha \in \mathbb{R}$, momentum factors $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 10^{-8}$, pretrained parameter vector $\theta^* \in \mathbb{R}^n$, coefficient of quadratic penalty $\gamma \in \mathbb{R}$, annealing coefficient in objective function $\lambda(t) = 1/(1 + \exp(-k \cdot (t - t_0)), k \in \mathbb{R}, t_0 \in \mathbb{N}$

initialize timestep $t \leftarrow 0$, parameter vector $\theta_{t=0} \in \mathbb{R}^n$, first moment vector $m_{t=0} \leftarrow 0$, second moment vector $v_{t=0} \leftarrow 0$, schedule multiplier $\eta_{t=0} \in \mathbb{R}$

repeat

  $t \leftarrow t + 1$

  $\nabla f_t(\theta_{t-1}) \leftarrow$ SelectBatch(θ_{t-1}) ▷ select batch and return the corresponding gradient

  $g_t \leftarrow \lambda(t)\, \nabla f_t(\theta_{t-1}) + (1 - \lambda(t))\gamma(\theta_{t-1} - \theta^*)$

  $m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1) g_t$ ▷ here and below all operations are element-wise

  $v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$

  $\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$ ▷ β_1 is taken to the power of t

  $\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$ ▷ β_2 is taken to the power of t

  $\eta_t \leftarrow$ SetScheduleMultiplier(t) ▷ can be fixed, decay, or also be used for warm restarts

  $\theta_t \leftarrow \theta_{t-1} - \eta_t \left(\lambda(t)\, \alpha \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon) + (1 - \lambda(t))\gamma(\theta_{t-1} - \theta^*) \right)$

until *stopping criterion is met*

return optimized parameters θ_t

penalized by the relatively smaller amount than other weights.

With RECADAM optimizer, we decouple the gradient of the quadratic penalty $\gamma(\theta - \theta^*)$ and the annealing coefficient $\lambda(t)$ in Line 12 of Algorithm 1. In this way, only the gradient of target task objective function $\nabla f(\theta)$ get adapted during the optimization steps, and all the weights of the training model would be more effectively penalized with the same rate $(1 - \lambda(t))\gamma$.

Since the RECADAM optimizer is only one line modification from Adam optimizer, it can be easily used by feeding the additional parameters, including the pretrained parameters and a few hyperparameters of the Pretraining Simulation and Objective Shifting mechanisms.

4 Experiments

4.1 Setup

Model: We conduct the experiments with the deep pretrained language model BERT-base (Devlin et al., 2019) and ALBERT-xxlarge (Lan et al., 2020).

BERT is a deep bi-directional pretrained model based on multi-layer Transformer encoders. It is pretrained on the large-scale corpus with two unsupervised tasks: Masked LM and Next Sentence Prediction, and has achieved significant improvements on a wide range of NLP tasks. We use the BERT-base model with 12 layers, 12 attention heads and 768 hidden dimensions (total 108M parameters).

ALBERT is the latest deep pretrained LM that achieves state-of-the-art performance on several benchmarks. It improves BERT by the parameter reduction techniques and self-supervised loss for sentence-order prediction (SOP). The ALBERT-xxlarge model with 12 layers, 64 attention heads, 128 embedding dimensions and 4,096 hidden dimensions (total 235M parameters) is the current state-of-the-art model released by Lan et al. (2020).

Data: We evaluate our methods on the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2019).

GLUE is a well-known benchmark evaluating model capabilities for natural language understanding. It includes 9 tasks: Corpus of Linguistic Acceptability (CoLA; Warstadt et al. 2018), Stanford Sentiment Treebank (SST; Socher et al. 2013), Microsoft Research Paraphrase Corpus (MRPC; Dolan and Brockett 2005), Semantic Textual Similarity Benchmark (STS; Cer et al. 2017), Quora Question Pairs (QQP),[2] Multi-Genre NLI (MNLI; Williams et al. 2018), Question NLI (QNLI; Rajpurkar et al. 2016), Recognizing Textual Entailment (RTE; Dagan et al. 2006; Bar Haim et al. 2006; Giampiccolo et al. 2007; Bentivogli et al. 2009) and Winograd NLI (WNLI; Levesque et al. 2011).

Following previous works (Yang et al., 2019; Liu et al., 2019; Lan et al., 2020), we report our single-task single-model results on the dev set of 8 GLUE tasks, excluding the problematic WNLI dataset.[3] We report Pearson correlations for STS, Matthew's correlations for CoLA, the "match" condition (MNLI-m) for MNLI, and accuracy scores

[2] https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs

[3] https://gluebenchmark.com/faq

for other tasks.

Implementation: As discussed in § 3.3, we implement the Pretraining Simulation and Objective Shifting techniques with the proposed RECADAM optimizer. We fine-tune the additional output layer with the vanilla Adam optimizer, since it is excluded in the parameters of pretrained LMs. Our methods use random initialization because of the pretrained knowledge recalling implementation, while vanilla fine-tuning initializes the fine-tuning model with the pretrained parameters.

We use the data processing and evaluation script implemented by HuggingFace Transformers library.[4] We fine-tune BERT-base and ALBERT-xxlarge model with the same hyperparameters following Devlin et al. (2019) and Lan et al. (2020), except for the maximum sequence length which we set to 128 rather than 512. For the BERT-base model, we set the learning rate to 2e-5, use the gradient bias correction and select the training step (61,360 on MNLI, 56,855 on QQP, 33,890 on QNLI, 21,050 on SST, 13,400 on CoLA, 9,000 on STS, 11,500 on MRPC, 7,800 on RTE) to improve the fine-tuning stability on each task (Mosbach et al., 2020; Zhang et al., 2020b). We note that we fine-tune on RTE, STS, and MRPC directly using the pretrained LM while the previous works are using an MNLI checkpoint for further performance improvement. As for the hyperparameters of our methods, we set γ in the quadratic penalty to 5,000, and select the best t_0 and k in $\{100, 250, 500, 1{,}000\}$ and $\{0.05, 0.1, 0.2, 0.5, 1\}$ respectively for the annealing coefficient $\lambda(t)$ on each dev set. Following previous works (Liu et al., 2019; Lan et al., 2020), we report the score of 5 differently-seeded runs for each result.

4.2 Results on GLUE

Table 1 shows the single-task single-model results of our RECADAM fine-tuning method comparing to the vanilla fine-tuning method with BERT-base and ALBERT-xxlarge model on the dev set of the GLUE benchmark. We also present the single-task single-model results with the BERT-base model on the test set of the GLUE benchmark in Appendix A.1, where we achieve 1.0% improvement on average.

Results with BERT-base: With the BERT-base model, we outperform the vanilla fine-tuning method on 7 out of 8 tasks of the GLUE benchmark and achieve 1.0% improvement on the average median performance.

Especially for the tasks with smaller training data (<10k), our method can achieve significant improvements (+1.7% on average) compared to the vanilla fine-tuning method. Because of the data scarcity, vanilla fine-tuning on these tasks is potentially brittle and prone to overfitting and catastrophic forgetting problems (Phang et al., 2018; Jiang et al., 2019). With the proposed RECADAM method, we successfully achieve better fine-tuning by learning target tasks while recalling the knowledge of pretraining tasks.

It is interesting to find that compared to the median results with the BERT-large model, we can also achieve better results on more than half of the tasks (e.g., +4.0% on RTE, +0.4% on STS, +1.8% on CoLA, +0.4% on SST, +0.1% on QQP) and better average results (+0.5%) of all the GLUE tasks. Thanks to the less catastrophic forgetting realized by RECADAM, we can get better overall performance with much fewer parameters of the pretrained model.

Results with ALBERT-xxlarge: With the state-of-the-art model ALBERT-xxlarge, we outperform the vanilla fine-tuning method on 5 out of 8 tasks of the GLUE benchmark and achieve the state-of-the-art single-task single-model average median result 90.2% on the dev set of the GLUE benchmark.

Similar to the results with the BERT-base model, We find that our improvements mostly come from the tasks with smaller training data (<10k), and we can improve the ALBERT-xxlarge model's median performance on these tasks by +1.5% on average. Also, compared to the reported results by Lan et al. (2020), we can achieve similar or better median results on RTE (+0.1%), STS (-0.1%), and MRPC (+1.0%) tasks without pretraining on the MNLI task.

Overall, we outperform the average median results of the baseline with the ALBERT-xxlarge model by 0.7%, which is lower than the improvement we gain with the BERT-base model (+1.0%). With advanced model design and pretraining techniques, ALBERT-xxlarge achieves significantly better performance on the GLUE benchmark, which would be harder to be further improved.

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

Model	MNLI 392k	QQP 363k	QNLI 108k	SST 67k	Avg >10k	CoLA 8.5k	STS 5.7k	MRPC 3.5k	RTE 2.5k	Avg <10k	Avg
BERT-base (Devlin et al., 2019)	84.4	-	88.4	92.7	-	-	-	86.7	-	-	-
BERT-base (rerun) $_{Median}$	84.8	**91.4**	91.6	93.0	90.2	60.6	89.8	86.5	71.1	77.0	83.6
BERT-base + RecAdam $_{Median}$	**85.0**	**91.4**	**91.9**	**93.6**	**90.5**	**62.4**	**90.4**	**87.7**	**74.4**	**78.7**	**84.6**
BERT-base (rerun) $_{Max}$	84.9	91.4	92.0	93.3	90.4	61.6	89.9	**88.7**	71.5	77.9	84.2
BERT-base + RecAdam $_{Max}$	**85.3**	**91.6**	**92.1**	**94.0**	**90.8**	**62.6**	**90.6**	**88.7**	**77.3**	**79.8**	**85.3**
BERT-large (Devlin et al., 2019)	86.6	91.3	92.3	93.2	90.9	60.6	90.0	88.0	70.4	77.3	84.1
XLNet-large (Yang et al., 2019)	89.8	91.8	93.9	95.6	92.8	63.6	91.8	89.2	83.8	82.1	87.4
RoBERTa-large (Liu et al., 2019)	90.2	92.2	94.7	96.4	93.4	68.0	92.4	90.9	86.6	84.5	88.9
ALBERT-xxlarge (Lan et al., 2020)	**90.8**	92.2	95.3	**96.9**	**93.8**	71.4	**93.0**	90.9	89.2	86.1	90.0
ALBERT-xxlarge (rerun) $_{Median}$	90.6	92.2	**95.4**	96.7	93.7	69.5	**93.0**	91.2	87.4	85.3	89.5
ALBERT-xxlarge + RecAdam $_{Median}$	90.5	**92.3**	95.3	96.8	93.7	**72.9**	92.9	**91.9**	**89.3**	**86.8**	**90.2**
ALBERT-xxlarge (rerun) $_{Max}$	**90.7**	92.2	95.4	96.8	93.8	72.1	**93.2**	91.4	89.9	86.7	90.2
ALBERT-xxlarge + RecAdam $_{Max}$	90.6	**92.4**	**95.5**	**97.0**	**93.9**	**75.1**	93.0	**93.1**	**91.7**	**88.2**	**91.1**

Table 1: State-of-the-art single-task single-model results on the dev set of the GLUE benchmark. The number below each task refers to the number of training data. The average scores of the tasks with large training data (>10k), the tasks with small training data (<10k), and all the tasks are reported separately. We rerun the baseline of vanilla fine-tuning without further pretraining on MNLI. We report median and maximum over 5 runs.

Method	CoLA	STS	MRPC	RTE	Avg
vanilla fine-tuning	60.6	89.8	86.5	71.1	77.0
RecAdam + PI	62.0	**90.4**	87.3	73.6	78.3
RecAdam + RI	**62.4**	**90.4**	**87.7**	**74.4**	**78.7**

Table 2: Comparison of different model initialization strategies: pretrained initialization (PI) and Random Initialization (RI). We report median over 5 runs.

4.3 Analysis

Model Initialization: With our RECADAM method, the model can be initialized with random values, and recall the knowledge of pretraining tasks while learning the new tasks.

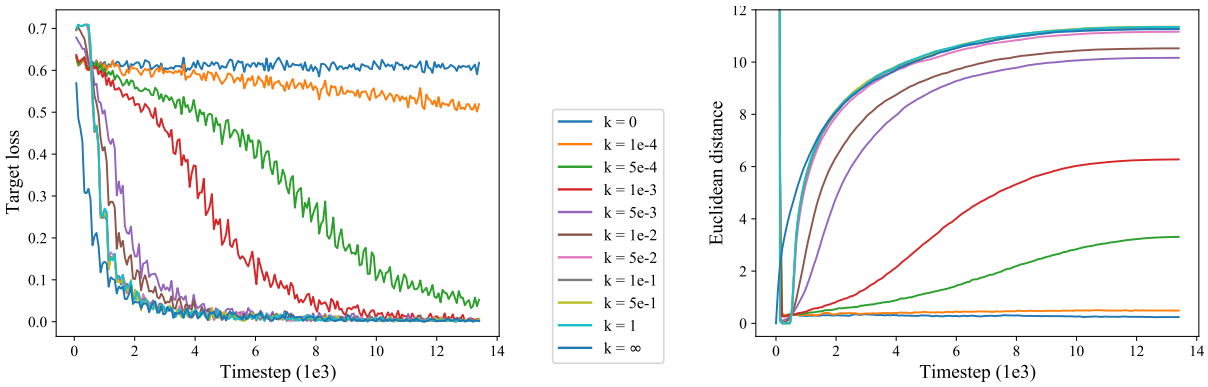
It is interesting to see whether the choice of initialization strategies would impact the performance of our RECADAM method. Table 2 shows the performance comparison of different initialization strategies for RECADAM obtained by the BERT-base model. It shows that RECADAM with both initialization strategies can outperform the vanilla fine-tuning method on all four tasks. For the target task STS, the model with pretrained initialization can achieve the same result as random initialization. For the other tasks (e.g., CoLA, MRPC, RTE), the models with random initialization can achieve better performance. It is because the randomly initialized model can benefit from a larger parameter search space. By contrast, with pretrained initialization, the search space would be limited to around the pretraining model, making it harder for the model to escape poor local minima and gain better performance on the new tasks.

Forgetting Analysis: As introduced in § 3.2, we realize multi-task fine-tuning with the Objective Shifting technique, which allows the model's learning objective to shift from the source tasks to the target tasks gradually. The hyperparameter k controls the rate of the objective shifting.

Figure 2 shows the learning curves of our fine-tuning methods with different k value obtained by BERT-base model trained on CoLA dataset. As discussed in § 3.2, Fine-tuning and multi-task learning can be regarded as the special cases ($k \to \infty$ and $k \to 0$) of our method.

As shown in Figure 2a, with the larger shifting rate k, the model can converge quickly on the target task. As k decreases, it takes a longer time for the model to converge on the target task because of the slower shifting from the pretrained knowledge recalling to target task learning.

Figure 2b shows the pretrained knowledge forgetting during the fine-tuning process. We measure the pretrained knowledge forgetting by the Euclidean distance between the weights of the fine-tuning model and the pretrained model. With vanilla fine-tuning ($k \to \infty$), the Euclidean distance begins at zero and increases as the model learns the target task. With a modest shifting rate k, at the very early timesteps, the Euclidean distance drops sharply because of the random initialization and pretrained knowledge recalling. Then the curve rises with the growth rate slowing down because of the target task learning. As k decreases, thanks to more iterations of pretrained knowledge recalling, the model can achieve less forgetting at the end of



(a) Training loss on the target task

(b) Knowledge forgetting from the source tasks

Figure 2: Learning curves obtained by BERT-base model trained with different objective shifting rate k on CoLA.

the fine-tuning.

Overall, our methods provide a bridge between fine-tuning and multi-task learning. With smaller k, the model achieves less knowledge forgetting from the source tasks but risks not converging completely on the target task. With a good balance between the pretrained knowledge recalling and new task learning, our methods can consistently outperform the vanilla fine-tuning by not only converging on target tasks but also less forgetting from source tasks.

5 Related Works

Catastrophic forgetting has been observed as a great challenge issue in sequential transfer learning, especially in the continuous learning paradigm (McCloskey and Cohen, 1989; French, 1999; Goodfellow et al., 2013; De Lange et al., 2019). Many methods have been proposed to avoid catastrophic forgetting (Kirkpatrick et al., 2017; Li and Hoiem, 2017; Rebuffi et al., 2017; Mallya and Lazebnik, 2018). We focus on regularization-based methods (Kirkpatrick et al., 2017; Li and Hoiem, 2017) which recall the previous knowledge with an regularization term, because they don't require the storage of the pretraining data, and are flexible on the new tasks. Regularization-based methods can be further divided into data-focused and prior-focused methods. Data-focused methods regularize the new task learning by knowledge distillation from the pretrained model (Hinton et al., 2015; Li and Hoiem, 2017; Zhang et al., 2020a). Prior-focused methods regard the distribution of the pretrained parameters as prior when learning the new task (Kirkpatrick et al., 2017; Zenke et al., 2017; Xuhong et al., 2018; Aljundi et al., 2018). We adopted the idea of prior-focused methods because they enable the model to learn more general knowledge from the pretrained parameters more efficiently. While the prior-focused methods, such as EWC (Kirkpatrick et al., 2017) and its variants (Schwarz et al., 2018; Liu et al., 2018), don't directly access to the pretraining data, they need some pretraining knowledge which is not available in our setting. Therefore, we further approximate to a quadratic penalty which is independent with the pretraining data given the pretrained parameters.

Catastrophic forgetting in NLP has raised increased attention recently (Mou et al., 2016; Arora et al., 2019; Chronopoulou et al., 2019). Many approaches have been proposed to overcome the forgetting problem in various domains, such as machine translation (Miceli-Barone et al., 2017; Thompson et al., 2019) and reading comprehension (Xu et al., 2019). As sequential transfer learning widely used for NLP tasks (Howard and Ruder, 2018; Devlin et al., 2019; Liu et al., 2019; Lan et al., 2020), previous works explore many fine-tuning tricks to reduce catastrophic forgetting for adaptation of the deep pretrained LMs (Howard and Ruder, 2018; Sun et al., 2019; Zhang et al., 2019; Chen et al., 2019; Jiang et al., 2019; Lee et al., 2020). In this paper, we bring the idea of multi-task learning which can inherently avoid catastrophic forgetting, and achieve consistent improvement with the proposed RECADAM optimizer.

6 Conclusion

In this paper, we propose to tackle the catastrophic forgetting in transferring deep pretrained language models by bridging two transfer learning paradigms: sequential fine-tuning and multi-task

learning. To cope with the absence of pretraining data during the joint learning of the pretraining task, we introduce a Pretraining Simulation mechanism to learn the pretraining task without data. Then we introduce the Objective Shifting mechanism to better balance the learning of the pretraining and downstream tasks. Experiments demonstrate the superiority of our method in transferring deep pretrained language models, and we provide the open-source RECADAM optimizer by integrating the proposed mechanisms into Adam optimizer to facilitate better usage of deep pretrained language models.

Acknowledgements

We are grateful for the helpful comments and suggestions from the anonymous reviewers. This work was supported by the National Natural Science Foundation of China (NSFC) via grant 61976072, 61632011 and 61772153.

References

Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. 2018. Memory aware synapses: Learning what (not) to forget. In *Proceedings of the European Conference on Computer Vision*, pages 139–154.

Gaurav Arora, Afshin Rahimi, and Timothy Baldwin. 2019. Does an lstm forget more than a cnn? an empirical study of catastrophic forgetting in nlp. In *Proceedings of the Australasian Language Technology Association*, pages 77–86.

Roy Bar Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second PASCAL recognising textual entailment challenge. *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*.

Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini. 2009. The fifth PASCAL recognizing textual entailment challenge. In *Proceedings of Text Analysis Conference*.

Samuel Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The SIGNLL Conference on Computational Natural Language Learning*, pages 10–21.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the International Workshop on Semantic Evaluation*, pages 1–14. Association for Computational Linguistics.

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 *Proceedings of the Advances in Neural Information Processing Systems*, pages 1906–1916.

Alexandra Chronopoulou, Christos Baziotis, and Alexandros Potamianos. 2019. An embarrassingly simple approach for transfer learning from pretrained language models. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, pages 2089–2095.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The PASCAL recognising textual entailment challenge. In *Machine Learning Challenges. Evaluating Predictive Uncertainty, Visual Object Classification, and Recognising Tectual Entailment*, pages 177–190. Springer.

Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. 2019. Continual learning: A comparative study on how to defy forgetting in classification tasks. *arXiv preprint arXiv:1909.08383*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, pages 4171–4186. Association for Computational Linguistics.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the International Workshop on Paraphrasing*.

Robert M. French. 1999. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128 – 135.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9. Association for Computational Linguistics.

Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. 2013. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the Association for Computational Linguistics*, pages 328–339.

Ferenc Huszár. 2017. On quadratic penalties in elastic weight consolidation. *arXiv preprint arXiv:1712.03847*.

Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Tuo Zhao. 2019. Smart: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization. *arXiv preprint arXiv:1911.03437*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*.

Eliyahu Kiperwasser and Miguel Ballesteros. 2018. Scheduled multi-task learning: From syntax to translation. *Transactions of the Association for Computational Linguistics*, 6:225–240.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

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 *Proceedings of the International Conference on Learning Representations*.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. 2020. Mixout: Effective regularization to finetune large-scale pretrained language models. In *Proceedings of the International Conference on Learning Representations*. OpenReview.net.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2011. The Winograd schema challenge. In *Proceedings of the Association for the Advancement of Artificial Intelligence Spring Symposium*, page 47.

Zhizhong Li and Derek Hoiem. 2017. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947.

Xialei Liu, Marc Masana, Luis Herranz, Joost Van de Weijer, Antonio M Lopez, and Andrew D Bagdanov. 2018. Rotate your networks: Better weight consolidation and less catastrophic forgetting. In *Proceedings of the International Conference on Pattern Recognition*, pages 2262–2268. IEEE.

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. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *Proceedings of the International Conference on Learning Representations*. OpenReview.net.

David JC MacKay. 2003. *Information theory, inference and learning algorithms*. Cambridge university press.

Arun Mallya and Svetlana Lazebnik. 2018. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7765–7773.

James Martens. 2014. New insights and perspectives on the natural gradient method. *arXiv preprint arXiv:1412.1193*.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*, volume 24, pages 109–165. Elsevier.

Antonio Valerio Miceli-Barone, Barry Haddow, Ulrich Germann, and Rico Sennrich. 2017. Regularization techniques for fine-tuning in neural machine translation. In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 1489–1494.

Marius Mosbach, Maksym Andriushchenko, and Dietrich Klakow. 2020. On the stability of fine-tuning bert: Misconceptions, explanations, and strong baselines. *arXiv preprint arXiv:2006.04884*.

Lili Mou, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin. 2016. How transferable are neural networks in nlp applications? In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 479–489.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 8024–8035. Curran Associates, Inc.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, pages 2227–2237.

Matthew E. Peters, Sebastian Ruder, and Noah A. Smith. 2019. To tune or not to tune? adapting pretrained representations to diverse tasks. In *Proceedings of the Workshop on Representation Learning for NLP*, pages 7–14. Association for Computational Linguistics.

Jason Phang, Thibault Févry, and Samuel R Bowman. 2018. Sentence encoders on stilts: Supplementary training on intermediate labeled-data tasks. *arXiv preprint arXiv:1811.01088*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 2383–2392. Association for Computational Linguistics.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. 2017. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2001–2010.

João Ribeiro, Francisco Melo, and João Dias. 2019. Multi-task learning and catastrophic forgetting in continual reinforcement learning. *EPiC Series in Computing*, 65:163–175.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Sebastian Ruder. 2019. *Neural transfer learning for natural language processing*. Ph.D. thesis, NUI Galway.

Jonathan Schwarz, Wojciech Czarnecki, Jelena Luketina, Agnieszka Grabska-Barwinska, Yee Whye Teh, Razvan Pascanu, and Raia Hadsell. 2018. Progress & compress: A scalable framework for continual learning. In *Proceedings of the International Conference on Machine Learning*, pages 4535–4544. PMLR.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Empirical Methods in Natural Language Processing*, pages 1631–1642.

Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2019. How to fine-tune bert for text classification? In *Proceedings of the China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer.

Brian Thompson, Jeremy Gwinnup, Huda Khayrallah, Kevin Duh, and Philipp Koehn. 2019. Overcoming catastrophic forgetting during domain adaptation of neural machine translation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, pages 2062–2068.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the International Conference on Learning Representations*. OpenReview.net.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. 2018. Neural network acceptability judgments. *arXiv preprint 1805.12471*.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*.

Ying Xu, Xu Zhong, Antonio Jose Jimeno Yepes, and Jey Han Lau. 2019. Forget me not: Reducing catastrophic forgetting for domain adaptation in reading comprehension. *arXiv preprint arXiv:1911.00202*.

Jiabin Xue, Jiqing Han, Tieran Zheng, Xiang Gao, and Jiaxing Guo. 2019. A multi-task learning framework for overcoming the catastrophic forgetting in automatic speech recognition. *arXiv preprint arXiv:1904.08039*.

LI Xuhong, Yves Grandvalet, and Franck Davoine. 2018. Explicit inductive bias for transfer learning with convolutional networks. In *Proceedings of the International Conference on Machine Learning*, pages 2825–2834.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Proceedings of the Advances in Neural Information Processing Systems*, pages 5754–5764.

Friedemann Zenke, Ben Poole, and Surya Ganguli. 2017. Continual learning through synaptic intelligence. In *Proceedings of the International Conference on Machine Learning*, pages 3987–3995. JMLR. org.

Jeffrey O Zhang, Alexander Sax, Amir Zamir, Leonidas Guibas, and Jitendra Malik. 2019. Side-tuning: Network adaptation via additive side networks. *arXiv preprint arXiv:1912.13503*.

Junting Zhang, Jie Zhang, Shalini Ghosh, Dawei Li, Serafettin Tasci, Larry Heck, Heming Zhang, and C-C Jay Kuo. 2020a. Class-incremental learning via deep model consolidation. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pages 1131–1140.

Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, and Yoav Artzi. 2020b. Revisiting few-sample bert fine-tuning. *arXiv preprint arXiv:2006.05987*.

Model	MNLI 392k	QQP 363k	QNLI 108k	SST 67k	Avg >10k	CoLA 8.5k	STS 5.7k	MRPC 3.5k	RTE 2.5k	Avg <10k	Avg
BERT-base (Devlin et al., 2019)	84.6	**71.2**	90.5	93.5	85.0	52.1	**85.8**	**88.9**	66.4	73.3	79.1
BERT-base + RecAdam	**85.0**	**71.2**	**91.0**	**94.0**	**85.3**	**55.4**	**85.8**	88.6	**70.0**	**75.0**	**80.1**

Table 3: Results on the test set of the GLUE benchmark, scored by the evaluation server.[5] The number below each task refers to the number of training data. The average scores of the tasks with large training data (>10k), the tasks with small training data (<10k), and all the tasks are reported separately. Following Devlin et al. (2019), we report F1 scores for QQP and MRPC, Spearman correlations for STS-B, Matthew's correlations for CoLA, and accuracy scores for the other tasks. We submitted the best model on each dev set.

A Appendices

A.1 Test Results on GLUE Tasks

As shown in § 4.2, we report both the median and the maximum scores over five runs for the vanilla fine-tuning method and our RECADAM fine-tuning method on the dev set of the GLUE benchmark. The results with the BERT-base model show that we outperform the baseline method by 1.0% on the average median performance and 1.1% on the average maximum performance.

To confirm our best model's generalization on the dev set, we present the single-task single-model results with the BERT-base model on the test set of the GLUE benchmark in Table 3. Similar to the performance on the dev set, we achieve the same or better results on 7 out of 8 tasks of the GLUE benchmark and achieves 1.0% improvement on average.

Compared to the results (+0.3% on average) on the tasks with larger training data (>10k), we obtain more significant improvement (+1.7% on average) on the tasks with smaller training data (<10k). It is consistent with our findings on the dev results (discussed in § 4.2), which shows the generalization and effectiveness of the proposed RECADAM method.

[5] https://gluebenchmark.com/leaderboard