REDUCING COMMUNICATION OVERHEAD IN FEDERATED LEARNING FOR PRE-TRAINED LANGUAGE MODELS USING PARAMETER-EFFICIENT FINETUNING

Shubham Malaviya, Manish Shukla, Sachin Lodha
TCS Research
India
{shubham.malaviya,mani.shukla,sachin.lodha}@tcs.com

ABSTRACT

Pre-trained language models are shown to be effective in solving real-world natural language problems. Due to privacy reasons, data may not always be available for pre-training or finetuning of the model. Federated learning (FL) is a privacy-preserving technique for model training, but it suffers from communication overhead when the model size is large. We show that parameter-efficient finetuning (PEFT) reduces communication costs while achieving good model performance in both supervised and semi-supervised federated learning. Also, often in real life, data for the target downstream task is not available, but it is relatively easy to obtain the data for other related tasks. To this end, our results on the task-level transferability of PEFT methods in federated learning show that the model achieves good zero-shot performance on target data when source data is from a similar task. Parameter-efficient finetuning can aid federated learning in building efficient, privacy-preserving Natural Language Processing (NLP) applications.

1 INTRODUCTION

Language models have produced state-of-the-art performances on a series of NLP tasks such as machine translation, question answering, and sentiment analysis, among others. With the advent of models like ELMo Peters et al. (2018), BERT Devlin et al. (2018), and GPT Brown et al. (2020), the paradigm for developing NLP models has shifted to Transfer Learning: first, pre-train a language model and, then fine-tune it on the target dataset. To get acceptable performance on a downstream task, a large amount of labeled data is required, as finetuning the language models on a small labeled dataset could lead to a problem of overfitting and knowledge forgetting. However, getting a large amount of data may not always be possible due to data privacy concerns and geography-based data protection regulations (for example, GDPR Voigt (2017)) that impose strict rules on how data is stored, shared, and used. Also, there are practical risks of data abuse once the data is shared with third parties to receive personalized AI-based services. To address this issue, we explore two potential machine learning paradigms: Federated Learning (FL) and Transfer Learning.

Federated learning has emerged as a privacy-preserving learning technique that learns a model in a collaborative way across decentralized devices without sending the client's data to an external server. FL is continual learning which goes on till the convergence or pre-defined number of rounds. In each training round, all clients participating in a particular round send their local model to the server, and the server sends back the aggregated model to clients. A major drawback here is that language models incur a large communication overhead in FL due to their size, limiting their applicability in a distributed data setting. To address this challenge, we explore Parameter Efficient FineTuning (PEFT) methods such as Prefix-tuning Li & Liang (2021), Adapter Houlsby et al. (2019), BitFit Ravfogel et al. (2021), and Lora Hu et al. (2022) for language models in a federated learning environment. Parameter-efficient finetuning updates only a small number of parameters (inherently in the model or additionally introduced) while freezing the remaining parameters of the language model. Therefore, in federated learning, only a small number of updated parameters are communicated between clients and the server lowering the communication cost. PEFT also has the natural advantage of lower training computation on client machines compared to finetuning the entire pre-trained language model (PLM) on the client's limited computing resources. The reduced communication and training cost of PEFT makes it feasible to bring the power of language models to end users and to build privacy-preserving NLP applications.

Sometimes the data for some tasks is available, but it is scarce for other related tasks. For example, data suitable for text classification is more easily accessible than Named Entity Recognition (NER). Recent work shows the importance of transfer learning, wherein knowledge learned from one task gets transferred to another task. Knowledge transfer from

data-rich source tasks shown to improve target task performance for text classification Phang et al. (2018), sequence labeling Liu et al. (2019), and Question Answering (QA) tasks Talmor & Berant (2019). In recent work, Ding et al. (2022) have explored task-level transferability for the centralized setting, but it is not explored in a federated learning setup yet. Moreover, as non-independent and identically distributed (non-i.i.d.) data, that is, uneven distribution of data among clients is a significant challenge in FL (Kairouz et al. (2021)), it is important to check the transferability of PEFT in federated learning. Our contributions are as follows:

- The premise of readily accessible labeled data on client devices is not consistently valid in real-world scenarios, as clients often lack the necessary domain knowledge or incentives to label their data. Yet, obtaining expert-annotated labeled data on a server is possible. In light of this more realistic scenario, our research delves into an extensive experimental evaluation involving both supervised and semi-supervised methodologies within the federated learning framework (Section 5.4).
- We analyze the performance of four parameter-efficient finetuning (PEFT) methods under different non-i.i.d. settings in federated learning (Section 5.4).
- We analyze the effect of varying client fractions for a federated training round on model performance (Section 5.4.1). Additionally, we evaluate the task-level transferability of PEFT approaches in federated learning using six datasets of three different tasks from the GLUE Benchmark (Section 5.4.2).

2 Preliminaries

2.1 Federated Learning

The goal of federated learning is to collaboratively learn a global prediction model without exchanging client data. Let C be the number of clients collaborating in the learning process and R be the number of training rounds. In each round, the server first randomly selects m clients ($m \leq C$), and sends the global model W_g to them. Each participating client then trains a local model W_c on their local dataset $\mathcal{D}_c = \{x_1, \ldots, x_{N_c}\}$, where $N_c = |\mathcal{D}_c|$ is the total number of examples for the c^{th} client. The server then aggregates all the local models from the selected m clients to obtain a global model $W_g = 1/N * \sum_{c=1}^{m} W_c * N_c$. Here $N = \sum N_c$. The procedure is repeated for R number of rounds or until convergence. This procedure is generally referred to as FedAvg (McMahan et al. (2017)) in the literature.

Conventional federated learning assumes labeled data on the client, which may not be feasible in practice. Hence, we have also performed experiments in a semi-supervised setting with clients having unlabeled data and the server having few labeled data curated by experts. As clients only have unlabeled data, unsupervised representation learning is carried out during the local model training. Similar to federated learning, in FSSL, the server aggregates the local models W_c from all the participating clients' local models to obtain a global model W_g. Additionally, in FSSL, the server updates the global model parameters by training the model on the labeled dataset $\mathcal{D}_s$ and sends the updated model to clients in the next round.

2.2 Delta Tuning

The backbone of pre-trained models (PLMs) is generally a transformer model introduced by Vaswani et al. (2017). PLMs are composed of L identical stacked blocks, where each block consists of mainly two types of sub-layers: multi-head attention (MHA) and a feed-forward network (FFN). Each attention and feed-forward layer is followed by residual connection and layer normalization. Given a set of queries packed together in a query matrix $Q \in \mathbb{R}^{n \times d_k}$, keys and values packed together in $K \in \mathbb{R}^{m \times d_k}$ and $V \in \mathbb{R}^{m \times d_v}$, attention is computed as:

$$Attn(Q, K, V) = softmax\left(\frac{QK^\top}{\sqrt{d_k}}\right)V \tag{1}$$

where n and m are the number of queries and key-value pairs respectively. Since we are exploring the transferability aspect of PEFT methods, we put more focus on which parts of the PLM these methods affect and give a general idea about them.

Adapter-based tuning inserts bottleneck adapter layers consisting of down and up projection matrices, W_{down} and W_{up}, between layers of PLMs. Parameter addition by this method can be described as

$$LayerNorm(X + H(X)) \longrightarrow LayerNorm(X + Delta(H(X))).$$
$$where\ Delta(X) = X + \sigma(XW_{down})W_{up},\ \sigma(\cdot) \text{ is an activation function}$$

Prefix tuning prepends tunable prefix vectors P_k and P_v to the keys and values of each head in the multi-head attention.

$$Attn(QW_q, KW_k, VW_v) \longrightarrow Attn(QW_q, Concat(P_k, KW_k), Concat(P_v, VWv))$$

BitFit is a specification-based method that specifies the part of the parameters of PLMs to be updated and does not add any new parameters to the model. BitFit only fine-tunes the bias vectors of the PLMs.

LoRA injects trainable low-rank matrices into attention layers for approximating the change of original weight matrices. The attention module for LoRA with rank r is defined as follows:

$$Attn(QW_q, KW_k, VW_v) \longrightarrow Attn(QW_q, Delta_k(K) + KW_k, Delta_v(V) + VW_v).$$
$$where\ Delta(X) = XW_{d_k \times d_r} W_{d_r \times d_k}$$

3 Related Work

Parameter-Efficient Fine-Tuning
With growing interest in pre-trained models, various efforts have been made to utilize language models for federated learning. From a model pre-training perspective, Tian et al. (2022) proposed a method to pre-train a model on data distributed among clients in a federated learning setup. The work of Chen et al. (2023) and Qu et al. (2022) aim to fill the gap between centralized and federated training with parameter initialization and model architecture designing, respectively. In their work on multilinguality in FL, Weller et al. (2022) show that the pre-trained model reduces the effect of non-i.i.d data in FL. Different from previous work, Tan et al. (2022); Chen et al. (2022) explore finetuning methods of pre-trained models in FL. Tan et al. (2022) utilize contrastive loss and prototypical networks to learn data representation. To reduce communication costs, only learned prototypes are transferred between clients and the server. Three different types of finetuning, namely, input modification, adding extra modules, and adjusting the backbone are explored by Chen et al. (2022) for communication-efficient federated learning. Shysheya et al. (2022) proposed transfer learning based data and parameter efficient system, FiT, for personalized federated learning. However, these studies are evaluated only in the vision domain. Although Zhang et al. (2022) investigated the performance of PEFT specifically for language models in a federated supervised learning setting, they did not perform the task-level transferability analysis on NLP datasets. Ding et al. (2022) categorizes parameter-efficient tuning methods into three groups: addition-based (adapter, prefix), specification-based (BitFit) and reparameterization-based (LoRA) methods. They have performed a detailed study of these PEFT methods and transferability analysis on text datasets in a centralized setting wherein data is stored at a centralized location. Contrary to them, we have evaluated task-level transferability in both supervised and semi-supervised settings in FL.

Federated Semi-supervised Learning (FSSL)
Recent work tackles the problem of labeled data scarcity in FL with semi-supervised learning techniques that use unlabeled data with a small amount of labeled data. Although consistency regularization-based methods such as FedMatch (Jeong et al. (2021)), FedCon (Long et al. (2021)) show good performance in FSSL for the vision domain, it requires data augmentation to be well defined. However, in the text domain, data augmentation is not straightforward, and changing only a few words can destroy the meaning of the sentence. For example, "he had his car *cleaned*" and "he had *cleaned* his car" have two different semantic meanings. To solve the problem of data augmentation, Malaviya et al. (2023) proposed a data augmentation-free framework, FedFAME, based on model contrastive learning for FSSL.

4 Method

Let $W_p = \{w_1, w_2, ..., w_M\}$ original PLM parameters, $W'_p = \{w'_1, w'_2, ..., w'_{M'}\}$ parameters of adapted PLM model for finetuning, and W_D trainable parameters. In vanilla fine-tuning, $M = M'$ and $|W_D| = |W_p|$, that is, the number of parameters updated during finetuning is equal to the number of original PLM parameters. Whereas in PEFT, few parameters are added to PLM while keeping the original PLM parameters frozen, which results in $M' \geq M$ and $|W_D| \ll |W_p|$. For PEFT, W_D is the added parameters or a part of the PLM parameters to be updated by PEFT methods.

We have discussed the federated learning procedure for a supervised setup in Section 2.1, generally referred to as FedAvg in the literature. As discussed earlier, in federated learning, the assumption of labeled data on the client device does not always hold for real-life applications. The lack of domain expertise and the lack of incentives lead to only unlabeled data on the client side. The issue of model training in this setup is partially addressed by semi-supervised learning in FL, wherein some data is annotated by experts on the server. Existing work in Federated

semi-supervised learning (FSSL) such as FedMatch and FedCon is based on consistency regularization which requires data augmentation to be well-defined to get better results. To solve the problem of data augmentation for text data, Malaviya et al. (2023) proposed a data augmentation-free framework for FSSL. The training procedure for federated supervised learning (FedAvg) and semi-supervised learning (FedFAME) is detailed in Algorithm 1. The details related to FedFAME training is provided in Appendix A.1.

Algorithm 1: Federated Parameter-Efficient Finetuning

Input: Client dataset $\mathcal{D}_k$, Labeled dataset $\mathcal{D}_s$ for server, PLM parameters W_p (freezed), Trainable parameters W_D, $|W_D| \ll |W_p|$, local and global learning rates η_u and η_g, local and global epochs for model training E_u and E_g, number of clients C, training rounds R, temperature τ.

Server Training:
Initialize W_p and trainable parameters for global model $W^1_{D_g}$
$r \leftarrow 1$
while $r \leq R$ **do**
  Randomly select a set of $\mathcal{M}$ clients from C
  $S \leftarrow \{\}$ **for** $c \in \mathcal{M}$ **do**
    // Send only global model trainable parameters $W^r_{D_g}$ to c^{th} client
    $W^r_{D_c}$ = **LocalTraining**($W^r_{D_g}$)
    $S = S \cup W^r_{D_c}$
  end
  $W^{r+1}_{D_g} \leftarrow 1/N * \sum_{c=1}^{|S|} W^r_{D_c} * N_c$ // $|S| = \mathcal{M}$
  // Additional traninig performed in FedFAME
  $itr \leftarrow 0$
  while $itr < E_g$ **do** // train $W^{r+1}_{D_g}$ on $\mathcal{D}_s$
    for *batch* $\{x, y\} \in \mathcal{D}_s$ **do**
      L_s = CrossEntropy($f_{[W_p:W^{r+1}_{D_g}]}(x), y$)
      $W^{r+1}_{D_g} \leftarrow W^{r+1}_{D_g} - \eta_g \nabla L_s$ // Update only trainable global parameters
    end
    $itr = itr + 1$
  end
  $r = r + 1$
end
LocalTraining(W^r):
  Initialize $W^r_{D_c} \leftarrow W^r$
  $itr \leftarrow 0$
  while $itr < E_u$ **do**
    for *batch* $\{x\} \in \mathcal{D}_c$ **do**
      if *FedAvg* **then** // Supervised Setup (Section 2.1)
        L_c = CrossEntropy($f_{[W_p:W^r_{D_c}]}(x), y$)
      end
      if *FedFAME* **then** // Semi-supervised setup. Labels not used here (Appendix A.1)
        $L_c = L_{con} + L_{distil}$ // Refer Equation 4
      end
      $W^r_{D_c} \leftarrow W^r_{D_c} - \eta_u \nabla L_c$ // Update only trainable local parameters
    end
    $itr = itr + 1$
  end
  return $W^r_{D_c}$ // send only trainable local parameters to the server

5 Experiments

5.1 Datasets

The General Language Understanding Evaluation benchmark (GLUE) benchmark proposed by Wang et al. (2018) has seen wide adoption in evaluating PLM's performance for parameter-efficient finetuning (PEFT) methods Li & Liang (2021); Ravfogel et al. (2021); Hu et al. (2022). We performed experiments on six GLUE datasets: SST-2, RTE,

Dataset	Description	Task	Train	Dev	Test	Metric
SST-2	Is the movie review positive, negative or neutral?	Sentiment Analysis	67349	436	436	Accuracy
RTE	Does sentence A entail sentence B?	NLI	2490	138	139	Accuracy
QNLI	Does sentence B contains the answer to the question in sentence A?	Question-Answering NLI	104743	2731	2732	Accuracy
MNLI	Does sentence A entail or contradict sentence B?	Multi-Genre NLI	353403	4907	4907	Accuracy
MRPC	Is the sentence B a araphrase of sentence A?	Paraphrase Identification	3668	204	204	F1 Score
QQP	Are two questions similar?	Paraphrase Identification	363846	20215	20215	Accuracy

Table 1: Dataset Summary. Here, NLI stands for Natural Language Inference.

MNLI, QNLI, MRPC, and QQP. Since the test set is not present in GLUE datasets, following the work of Ding et al. (2022), we evenly divide the original development set into two halves representing the new development and test sets. A summary of the datasets is given in Table 1.

5.2 Dataset Creation for Federated Learning

The original GLUE dataset is distributed among clients to simulate distributed data setup in real-life. Since the training set of RTE and MRPC datasets is small, it is distributed among 10 clients, whereas the training set of other datasets is distributed among 100 clients. The fraction of clients sampled for each training round is set to 0.3 and 0.05 for federated learning with 10 and 100 clients, respectively. In federated supervised learning, clients have labeled data, whereas, in FSSL, clients only have unlabeled data, and the server has labeled data. To create unlabeled data on clients in FSSL, labels are removed from the data. In their work, Malaviya et al. (2023) (FedFAME) create labeled data on the server by sampling a few labeled instances per class from the training set without discarding them from the training set. This data sampling strategy is helpful to analyze the model performance for a varying number of labeled instances on the server.

We utilize the Dirichlet distribution (α) to simulate non-independent and identically distributed (non-i.i.d.) data following the work of Malaviya et al. (2023); Zhang et al. (2022). Note that the data distribution will be more imbalanced for the smaller values of α. We sampled 10% of original training data as labeled data per class for RTE and MRPC, which is 120 and 175. For other datasets, the number of labeled instances is set to 300 as 10% of the original dataset becomes very large in size.

5.3 Implementation Details

We have used the Flower framework for federated learning and non-i.i.d. data partition Beutel et al. (2020). For the reproducibility of experiments, a random seed for data partition is set to 10. Unless otherwise stated explicitly, we set the communication round to 100 and the local and server training epoch to one for federated training. For the Pre-trained language model, we utilize the BERT-Base architecture released by Huggingface[1]. We use the Trainer API from the Huggingface for model training and initialize the model parameters using three random seeds {17,25 and 42}. We perform experiments for learning rates 1e-3 & 5e-5, and $\alpha = 1.0$ & 0.1.

For the implementation of parameter-efficient finetuning (PEFT) methods, we have utilized the architecture proposed in the original work of Prefix tuning Li & Liang (2021), Adapter-based tuning Houlsby et al. (2019), BitFit Ravfogel et al. (2021), and LoRA Hu et al. (2022). We denote federated learning implementation of these methods as FedPrefix, FedAdapter, FedBitFit, and FedLoRA, respectively. We utilized the OpenDelta[2] library developed by Ding et al. (2022) to implement the PEFT methods. All experiments are performed on a server with a single Nvidia Tesla V100 GPU with 60 GiB RAM and 32 GiB GPU Memory.

[1]https://github.com/huggingface/transformers
[2]https://github.com/thunlp/OpenDelta

Communication cost for transferring model parameters between clients and the server for PEFT methods with BERT-Base as a PLM is given in Table 2. As we can see, the number of trainable parameters for PEFT methods is significantly smaller compared to the original pre-trained model. Reduction in communication cost from 420 MB (original PLM) to $<$ 50 MB for PEFT methods makes it feasible to utilize the efficiency of pre-trained models in federated learning.

Method	**FedAvg**		**FedFAME**	
	$\mathcal{B}$ **(MB)**	**Trainable Parameters (%)**	$\mathcal{B}$ **(MB)**	**Trainable Parameters (%)**
Prefix	37.1	8.14	37.9	8.16
Adapter	3.5	0.81	4.3	0.82
BitFit	0.447	0.092	1.2	0.094
LoRA	0.186	0.035	0.952	0.037

Table 2: Communication cost ($\mathcal{B}$) of transferring model parameters between clients and the server and, percentage of PLM's parameters to be trained.

Dataset	**Prefix**	**Adapter**	**BitFit**	**LoRA**	**Full**
SST-2	**92.66**	92.2	**92.66**	**92.66**	**92.66**
RTE	62.59	59.71	56.12	53.24	**66.19**
QNLI	91	90.12	87.12	86.93	**91.5**
MNLI	81.97	82.8	76.43	76.3	**83.39**
MRPC	88.36	89.7	88.82	86.62	**92.20**
QQP	89.09	89.39	85.06	84.54	**91.14**

Table 3: Performance of PEFT methods and Full model training in a centralized setting. The model trained for 20 epochs with learning rate = 5e-5.

			Dirichlet ($\alpha = 1.0$)			
Method	**SST-2**	**RTE**	**QNLI**	**MNLI**	**MRPC**	**QQP**
FedPrefix	**88.23** (0.54) 88.07 (1.53)	56.35 (8.44) 56.59 (0.9)	71.27 (8.21) **77.81** (0.45)	51.53 (3.28) **54.86** (3.66)	81.57 (0.25) **83.24** (1.61)	65.17 (1.4) **72.36** (0.14)
FedAdapter	84.79 (4.04) 86.77 (1.13)	52.28 (5.46) 56.35 (0.9)	67.42 (10.41) 73.38 (1.07)	43.47 (3.59) 46.52 (0.7)	82.9 (1.08) 81.28 (1.11)	64.66 (1.71) 68.05 (2.96)
FedBitFit	61.31 (15.4) 83.18 (2.0)	53.48 (6.6) **57.07** (0.9)	51.2 (1.41) 72.5 (1.17)	40.06 (4.07) 45.81 (1.93)	81.93 (0.88) 81.14 (0.64)	61.25 (4.85) 66.33 (1.07)
FedLoRA	51.15 (2.92) 82.87 (4.07)	48.2 (5.09) 56.35 (0.34)	50.32 (2.25) 71.86 (1.72)	34.65 (0.1) 44.68 (1.73)	52.27 (38.38) 81.15 (0.45)	45.54 (11.35) 65.23 (1.81)
			Dirichlet ($\alpha = 0.1$)			
FedPrefix	55.43 (1.7) **87.77** (0.66)	**59.47** (5.77) 57.79 (1.79)	61.42 (8.13) **77.43** (1.25)	36.09 (1.77) **51.88** (3.99)	76.71 (9.41) **83.54** (1.74)	65.21 (1.49) **72.09** (0.41)
FedAdapter	60.02 (5.93) 85.7 (0.71)	58.51 (2.65) 56.59 (1.7)	62.02 (8.25) 73.6 (0.65)	37.37 (3.5) 47.22 (2.08)	77.89 (7.31) 81.97 (0.41)	63.41 (1.55) 68.62 (3.04)
FedBitFit	55.81 (3.68) 83.64 (2.58)	53.72 (2.9) 55.4 (1.17)	56.19 (9.31) 72.5 (0.77)	36.76 (1.77) 45.01 (1.77)	54.97 (38.89) 81.79 (0.35)	65.19 (2.59) 67.34 (0.99)
FedLoRA	51.15 (2.92) 81.73 (4.97)	48.2 (5.09) 56.35 (0.9)	50.32 (2.25) 71.61 (1.14)	34.65 (0.1) 44.06 (1.78)	52.27 (38.38) 81.72 (0.38)	45.54 (11.35) 66.16 (1.75)

Table 4: Performance analysis of PEFT in FL for learning rate 5e-5 for FedAvg and FedFAME. In each cell, the upper and lower halves represent the performance of FedAvg and FedFAME, respectively. We run experiments with three random seeds 17,25,42 and report the mean performance. The value in parenthesis denotes the standard deviation. We present results for two values of α (1.0 and 0.1) indicating uneven data distribution among clients.

5.4 Results

In section A.3 of their work, Devlin et al. (2018) have shown that finetuning BERT with learning rates (lr) 5e-5, 3e-5, and 2e-5 generally work well across all tasks. For our experiments, we have used the learning rates 5e-5 and 1e-3, respectively. The performance of the model trained for 20 epochs and lr = 5e-5 in a centralized setting is shown in Table 3. It can be observed that Prefix and Adapter tuning achieves better accuracies than BitFit and LoRA. A performance comparison of FedAvg (supervised setup) and FedFAME (semi-supervised setup) is presented in Table 4. Here, we have stated accuracies in the following manner: the upper half in each cell represents the performance of FedAvg, whereas the lower half represents the performance of FedFAME. Note that for all our experiments, we report

the f1-score for the MRPC dataset as in the GLUE benchmark and accuracy for other datasets. We have presented the model performance for different values of α in the Dirichlet distribution. A smaller value of α means more uneven data distribution among clients. For $\alpha = 1.0$, PEFT methods achieved the highest performance for datasets belonging to sentiment analysis and paraphrase identification tasks, SST-2, and MRPC. FedFAME outperforms FedAvg on all datasets except for SST-2 dataset. We observe a higher performance difference between PEFT methods for FedAvg, whereas these methods achieve nearly identical performance for FedFAME.

Considering a more real-life scenario, we evaluate PEFT methods on uneven data distribution by setting a value of α to 0.1 for Dirichlet distribution. We can see in the table that the performance of FedAvg decreases drastically for half of the datasets. For example, we can see an accuracy drop of around 30% for SST-2, 10% for QNLI, and 17% for MNLI datasets. On the other hand, FedFAME's performance is identical to $\alpha = 1.0$. The results show that even with a small amount of labeled data, FedFAME is more robust than FedAvg for non-i.i.d. data. FedPrefix consistently outperforms other methods on the majority of datasets when the lr is 5e-5. In comparison to a learning rate of 5e-5, we notice an overall enhancement in the performance of FedAvg when the learning rate is set to 1e-3, with substantial increases observed for FedAdapter and FedBitFit (Table 5). Moreover, FedAdapter outperforms other PEFT methods when the learning rate is 1e-3. In contrast, a considerable decline in accuracies for FedFAME is observed when compared to a learning rate of 5e-5. The standard deviation is also elevated for both FedAvg and FedFAME when the model is trained with a learning rate of 1e-3. The results shows that the learning rate plays an important role in a model's performance depending on the algorithm used for training. We perform our experiments using three random seeds {17, 25 and 42} for the training procedure and report the mean and standard deviation of the model performance.

			Dirichlet ($\alpha = 1.0$)			
Method	**SST-2**	**RTE**	**QNLI**	**MNLI**	**MRPC**	**QQP**
FedPrefix	89.37 (1.32) 77.37 (17.13)	56.35 (0.9) 56.35 (0.9)	72.67 (3.15) 64.98 (10.84)	52.7 (1.13) 46.53 (8.84)	77.96 (4.86) 81.47 (0.11)	68.33 (3.62) **76.19** (4.01)
FedAdapter	**91.21** (0.11) 91.13 (0.11)	**60.43** (4.24) 58.03 (2.96)	**76.43** (3.97) 76.33 (3.9)	58.63 (1.41) **59.68** (2.01)	81.33 (4.34) **83.72** (2.17)	74.01 (1.19) 74.57 (0.49)
FedBitFit	85.86 (1.42) 86.16 (1.85)	56.12 (5.6) 55.16 (5.13)	61.2 (10.53) 61.44 (10.86)	44.0 (2.45) 48.64 (7.69)	79.55 (3.98) 82.01 (0.86)	67.25 (3.18) 68.81 (3.87)
FedLoRA	51.15 (2.92) 62.84 (18.18)	48.2 (5.09) 53.24 (6.35)	50.32 (2.25) 58.47 (11.57)	34.65 (0.1) 41.42 (9.64)	50.01 (35.74) 54.83 (38.78)	45.54 (11.35) 57.13 (13.88)
			Dirichlet ($\alpha = 0.1$)			
FedPrefix	53.21 (0.0) 53.21 (0.0)	58.99 (5.09) 58.99 (5.09)	51.16 (1.23) 51.39 (1.1)	38.86 (3.05) 38.24 (3.7)	50.01 (35.74) 54.34 (38.43)	65.44 (2.59) 65.44 (2.59)
FedAdapter	85.47 (4.16) **85.63** (4.27)	**62.59** (5.38) 59.71 (6.1)	61.22 (11.0) **70.25** (9.89)	**50.89** (10.15) 48.3 (6.57)	81.45 (3.66) **83.62** (1.07)	67.71 (3.23) **71.45** (2.49)
FedBitFit	61.54 (10.97) 65.44 (16.48)	56.83 (1.55) 56.12 (0.59)	53.92 (3.3) 62.49 (9.79)	40.2 (5.95) 43.79 (10.96)	51.89 (36.83) 54.55 (38.57)	66.93 (2.64) 67.99 (3.85)
FedLoRA	51.15 (2.92) 62.46 (17.64)	48.2 (5.09) 52.76 (5.88)	50.32 (2.25) 58.3 (11.34)	34.65 (0.1) 40.91 (8.92)	50.01 (35.74) 55.35 (39.16)	45.54 (11.35) 57.04 (13.78)

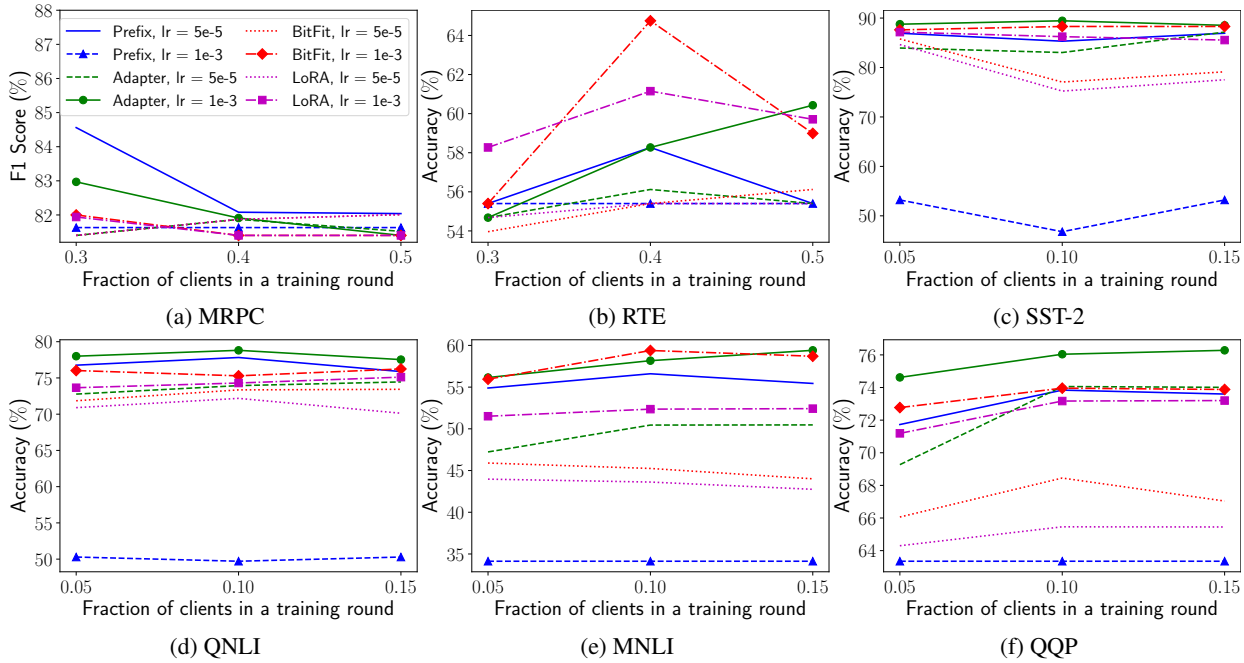
Table 5: Performance analysis of PEFT in federated learning for learning rate 1e-3 for FedAvg and FedFAME.

5.4.1 Analysis of varying client fractions for a federated training round

In federated learning, a subset of clients is randomly selected from the entire client pool for each training round. In this section, we examine how altering the client fraction impacts model performance. Figure 1 illustrates the results for models trained using various PEFT methods with learning rates of 1e-3 and 5e-5 for FedFAME. The corresponding results for FedAvg can be found in Figure 3. The model parameters for this experiment were initialized using a random seed of 42. For the RTE and MRPC datasets, we selected client fractions of 0.3, 0.4, and 0.5 from a pool of 10 clients, yielding participation of 3, 4, and 5 clients in each federated training round, respectively. For the remaining datasets, we chose client fractions of 0.05, 0.1, and 0.15 from a total of 100 clients for our experimentation.

As the client fraction increases, we observe minor improvements in model performance for the MNLI and QQP datasets. However, results for the remaining datasets indicate that increasing the client fraction does not necessarily lead to better model performance. As seen in Table 4 and Table 5, the performance of PEFT methods is heavily

influenced by the learning rate, which is also evident in Figure 1. Furthermore, the best PEFT method for a specific dataset varies depending on the learning rate. For instance, Adapter tuning is the best-performing method for the QQP dataset when the learning rate is set to 1e-3, while Prefix tuning surpasses other methods with a learning rate of 5e-5. These results demonstrate that there is no deterministic set of values for hyperparameters such as learning rate and client fraction, that universally yield optimal performance across all datasets.



(a) MRPC (b) RTE (c) SST-2

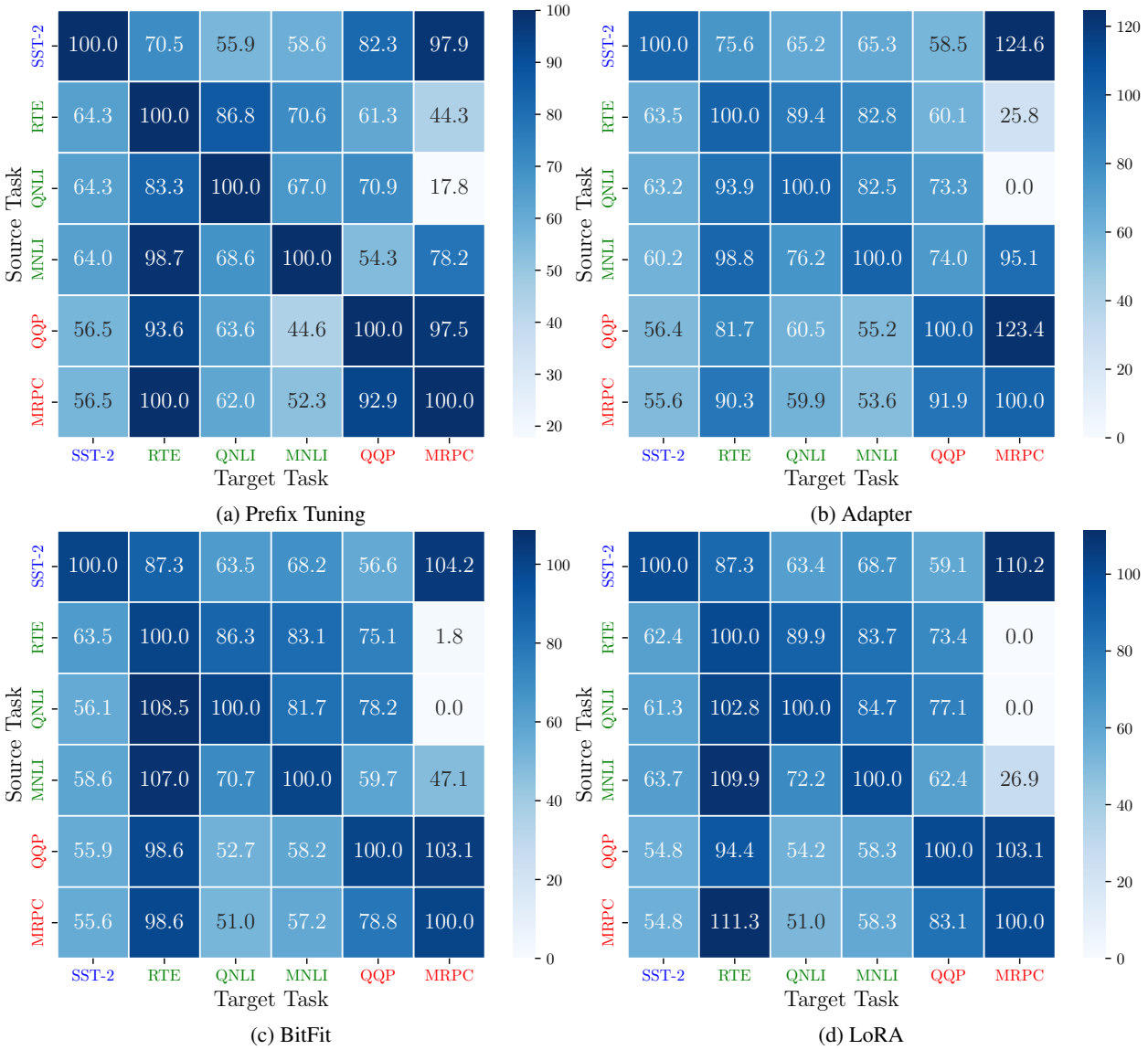
(d) QNLI (e) MNLI (f) QQP

Figure 1: Performance evaluation of varying client fractions sampled per FL round for FedFAME. The MRPC and RTE datasets are distributed among 10 clients, while the remaining datasets are distributed among 100 clients. Here, 'lr' denotes learning rate.

5.4.2 Task-level Transferability

Recently, Vu et al. (2022) and Su et al. (2022) have shown cross-task transferability for prompt tuning. In their work, Ding et al. (2022) have further demonstrated the task-level transferability of PEFT methods for PLM finetuning in a centralized data setting. We present the results for the transferability of a model trained in a centralized setting in Appendix A.4. In this work, we are further interested in the transferability of PEFT methods in federated learning because of uneven data distribution between clients. Therefore, we have investigated task-level transferability on six datasets of three different tasks from GLUE Benchmark. Since the FedFAME framework performed better compared to FedAvg for learning rate 5e-5 and $\alpha = 0.1$, we present the zero-shot performance of PEFT methods for FedFAME in Figure 2. Results for FedAvg are provided in Appendix A.3. To check the transferability, the delta parameters of PLM trained on one dataset are transferred to all other datasets. Next, the relative zero-shot performance of the model on the target dataset is computed by zero-shot transferring performance divided by the original performance on the target dataset. Here, we show relative performance for better visualization of the change in accuracy.

In general, we can observe a good level of transferability among datasets. Specifically, performance is high between datasets belonging to the same task, and it becomes lower when datasets are from different source and target tasks. It can be seen in Figure 2 that the RTE dataset achieves the highest level of transferability for all PEFT methods. For prefix and LoRA tuning, RTE gets better zero-shot performance with the source data of different tasks than the data belonging to the same task. Similarly, task-level transferability is lowest for the MRPC dataset, and the performance is poor when data is from Natural Language Inference (NLI) task. Our observations for PEFT methods in federated learning are in line with the work of Ding et al. (2022) in a centralized setting. Although the model trained in a centralized setting achieves better accuracies, we observe that the transferability of the model parameters trained in federated learning (Figure 2) is better than the centralized setting (Figure 5)). One interesting thing to observe here is that transferring does not always hold symmetric properties between two datasets. i.e., if parameter transfer from dataset A to dataset B works well, it does not always guarantee that vice versa will work well too. For example, the



(a) Prefix Tuning (b) Adapter

(c) BitFit (d) LoRA

Figure 2: Examining zero-shot transfer performance of PEFT methods employing BERT-Base PLM with a learning rate of 5e-5 and Dirichlet ($\alpha = 0.1$). Relative performance (zero-shot transferring performance / original performance) (%) on the target tasks (columns) when delta parameters are transferred from the source tasks (rows) is presented here. Here, task types are represented with the following colors: Blue: sentiment analysis, Green: natural language inference, Red: paraphrase identification.

model achieves near original zero-shot accuracy for the RTE dataset when we utilize delta parameters from the MRPC dataset. However, the performance is poor for the MRPC dataset when the source dataset is RTE. Overall results show that it is promising to utilize model parameters trained on similar tasks using PEFT methods in federated learning setup and to build privacy-sensitive NLP applications when data for target tasks are scarce but available for source tasks.

6 Limitations

A significant assumption of our work is that the pre-trained model can fit in the client device's memory. However, with the progression of the field, models are getting larger and cannot fit in the device's memory. For example, the GPT-3 model consists of 178 billion parameters, requiring more than 700 GB of memory to store the model. Strategies like

knowledge distillation, pruning, and quantization reduce the model's size and make it deployable on edge devices. It is necessary to analyze the effect of these model downscaling techniques on parameter-efficient finetuning and task-level transferability of the downscaled model parameters in a federated learning setup.

7 Discussion & Conclusion

Data privacy and Data scarcity are two major obstacles in getting sufficient data for pre-training and finetuning language models. Federated and transfer learning together enable the use of pre-trained language models (PLMs) on the end-user device and the building of efficient privacy-preserving NLP applications.

Federated learning (FL) would be a viable option for collaborative learning from data silos created in presence of strict data protection regulations. However, the size of pre-trained language models (PLMs) causes large communication overhead in FL making their applicability in real-life infeasible. We show that parameter-efficient finetuning (PEFT) methods reduce the communication cost from 420 MB to $<$ 50 MB and for some methods, it is even $<$ 1 MB for the BERT-Base model (Table 2). Because of achieving good model performance even with small trainable parameters (Table 4 & 5), PEFT could become a defacto tool for utilizing pre-trained models in federated learning. Another general problem from the dataset perspective is that while data for one task is available, it is scarce for another related task. Different from existing work that explores the transferability of model parameters for PEFT methods in a centralized setting, we have investigated whether model parameter transfer is efficient in a federated learning setup. We show the task-level transferability of these methods on six datasets for three different tasks from the GLUE benchmark (Figure 2). We observed that model performance is high when trained on datasets belonging to the same task, but the performance decreases when source and target datasets are from different tasks. Further, our observations for the transferability of PEFT methods in FL are in line with the existing work in a centralized setting.

Our work is a preliminary study of the efficiency and transferability of parameter-efficient finetuning in a federated learning setup. We have evaluated these methods on datasets containing general information. It remains to be evaluated whether this work is applicable to datasets in specific domains such as healthcare, cybersecurity, finance, and others.

References

Daniel J Beutel, Taner Topal, Akhil Mathur, Xinchi Qiu, Titouan Parcollet, Pedro PB de Gusmão, and Nicholas D Lane. Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*, 2020.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Hong-You Chen, Cheng-Hao Tu, Ziwei Li, Han Wei Shen, and Wei-Lun Chao. On the importance and applicability of pre-training for federated learning. In *International Conference on Learning Representations*, 2023. URL https://review.net/forum?id=fWWFv--P0xP.

Jinyu Chen, Wenchao Xu, Song Guo, Junxiao Wang, Jie Zhang, and Haozhao Wang. Fedtune: A deep dive into efficient federated fine-tuning with pre-trained transformers. *arXiv preprint arXiv:2211.08025*, 2022.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models. *arXiv preprint arXiv:2203.06904*, 2022.

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

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Wonyong Jeong, Jaehong Yoon, Eunho Yang, and Sung Ju Hwang. Federated semi-supervised learning with inter-client consistency & disjoint learning. In *9th International Conference on Learning Representations, ICLR 2021*, 2021.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 4582–4597, 2021.

Nelson F Liu, Matt Gardner, Yonatan Belinkov, Matthew E Peters, and Noah A Smith. Linguistic knowledge and transferability of contextual representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 1073–1094, 2019.

Zewei Long, Jiaqi Wang, Yaqing Wang, Houping Xiao, and Fenglong Ma. Fedcon: A contrastive framework for federated semi-supervised learning. *arXiv preprint arXiv:2109.04533*, 2021.

Shubham Malaviya, Manish Shukla, Pratik Korat, and Sachin Lodha. Fedfame: A data augmentation free framework based on model contrastive learning for federated semi-supervised learning. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, SAC '23, pp. 1114–1121, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9781450395175. doi: 10.1145/3555776.3577613. URL https://doi.org/10.1145/3555776.3577613.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *CoRR*, abs/1802.05365, 2018. URL http://arxiv.org/abs/1802.05365.

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

Liangqiong Qu, Yuyin Zhou, Paul Pu Liang, Yingda Xia, Feifei Wang, Ehsan Adeli, Li Fei-Fei, and Daniel Rubin. Rethinking architecture design for tackling data heterogeneity in federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10061–10071, 2022.

Shauli Ravfogel, Elad Ben-Zaken, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked languagemodels. *arXiv preprint arXiv:2106.10199*, 2021.

Aliaksandra Shysheya, John Bronskill, Massimiliano Patacchiola, Sebastian Nowozin, and Richard E Turner. Fit: Parameter efficient few-shot transfer learning for personalized and federated image classification. *arXiv preprint arXiv:2206.08671*, 2022.

Yusheng Su, Xiaozhi Wang, Yujia Qin, Chi-Min Chan, Yankai Lin, Huadong Wang, Kaiyue Wen, Zhiyuan Liu, Peng Li, Juanzi Li, Lei Hou, Maosong Sun, and Jie Zhou. On transferability of prompt tuning for natural language processing. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 3949–3969, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.290. URL https://aclanthology.org/2022.naacl-main.290.

Alon Talmor and Jonathan Berant. Multiqa: An empirical investigation of generalization and transfer in reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4911–4921, 2019.

Yue Tan, Guodong Long, Jie Ma, Lu Liu, Tianyi Zhou, and Jing Jiang. Federated learning from pre-trained models: A contrastive learning approach. *arXiv preprint arXiv:2209.10083*, 2022.

Yuanyishu Tian, Yao Wan, Lingjuan Lyu, Dezhong Yao, Hai Jin, and Lichao Sun. Fedbert: when federated learning meets pre-training. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 13(4):1–26, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Paul et al. Voigt. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 10(3152676):10–5555, 2017.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou', and Daniel Cer. SPoT: Better frozen model adaptation through soft prompt transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5039–5059, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.346. URL https://aclanthology.org/2022.acl-long.346.

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

Orion Weller, Marc Marone, Vladimir Braverman, Dawn Lawrie, and Benjamin Van Durme. Pretrained models for multilingual federated learning. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1413–1421, 2022.

Zhuo Zhang, Yuanhang Yang, Yong Dai, Lizhen Qu, and Zenglin Xu. When federated learning meets pre-trained language models' parameter-efficient tuning methods. *arXiv preprint arXiv:2212.10025*, 2022.

A Appendix

A.1 FedFAME

Local Training
As the client has only unlabeled data, the model contrastive loss is introduced in the local loss computation to learn a generalized data representation. A projection head of dimension 256 is added on top of a base encoder of a model to compare the representations of an instance in the projection space. For a given client c and an instance x, let $q^r = p^r_{W_c}(x)$ and $q^{r-1} = p^{r-1}_{W_c}(x)$ represent the output of the projection head of the local model at round r and $r-1$ respectively. Let q^r_g represent the output of the projection head of the global model at round r. Given this, the *model contrastive loss* is defined as:

$$L_{con} = -\log \frac{\exp(sim(q^r, q^r_g)/\tau)}{\exp(sim(q^r, q^r_g)/\tau) + \exp(sim(q^r, q^{r-1})/\tau)} \tag{2}$$

Where τ is a temperature hyperparameter. Since global model weights get updated on the labeled data on the server, global model's knowledge is distilled into client's local model using following distillation loss:

$$L_{distil} = CE(f_{W_g}(x), f_{W_c}(x)) \tag{3}$$

where $f(\cdot)$ is the output of the classification layer and CE is the Cross-Entropy loss. In round r, for the c^{th} client, the objective is to minimize the following cumulative loss function:

$$L_c = \min_{W^r_c} \mathbb{E}_{x \sim \mathcal{D}_c}[L_{con}(W^r_c; W^{r-1}_c; W^r_g; x) + L_{distil}(W^r_c; W^r_g; x)] \tag{4}$$

Note that, Equation 4 is minimized with respect to the local model parameters only. Global model parameters are not updated at any time during local training.

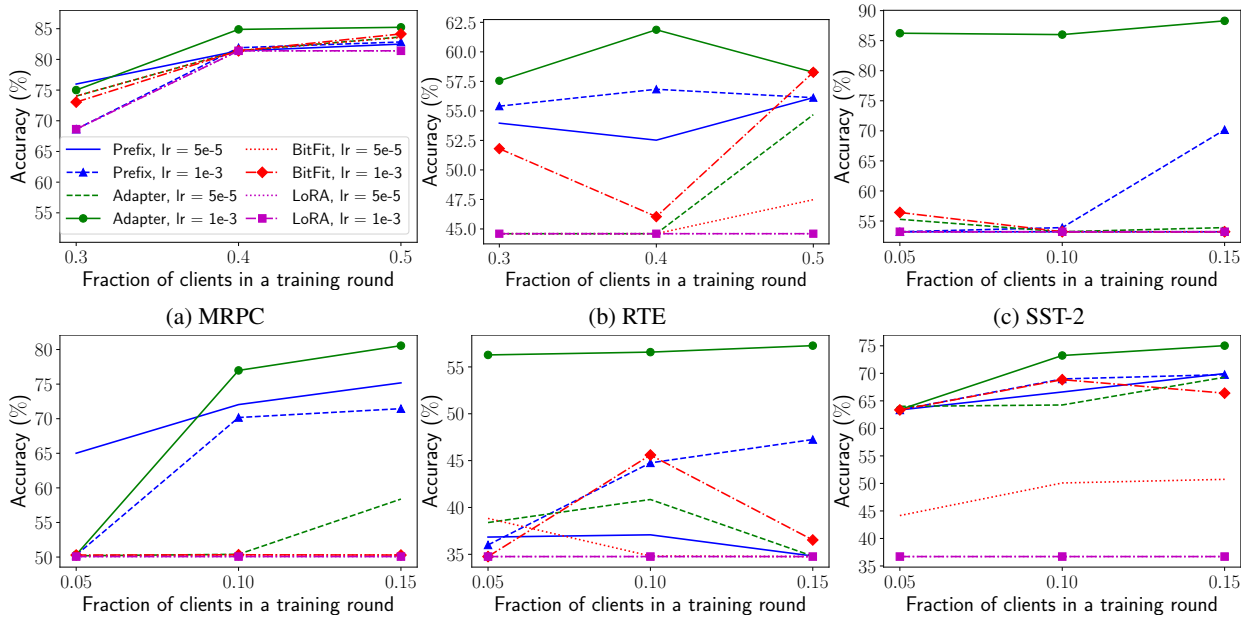
Server Training
In the general FL setting, the server only aggregates local models from clients and then sends them back. In FSSL, it is assumed that few labeled data can be collected on the server with help of domain experts. Utilizing this labeled data, the aggregated model (global model) is now trained by minimizing cross-entropy loss on labeled data $\mathcal{D}_s$. Formally, the weights W_g of the global model are updated by minimizing the following loss:

$$L_s = \min_{W_g} \mathbb{E}_{(x,y) \sim \mathcal{D}_s}[CE(W_g; (x, y))] \tag{5}$$

Note that, W_c and W_g are composed of pre-trained model parameters W_p and trainable paramters W_D introduced by PEFT methods. For FedFAME, $W_D = Concat(W_D, W_{proj})$ where W_{proj} denotes the parameters related to the projection head added for training. The training procedure for federated supervised learning (FedAvg) and semi-supervised learning (FedFAME) is detailed in Algorithm 1.

A.2 Analysis of varying client fractions for a federated training round for FedAvg

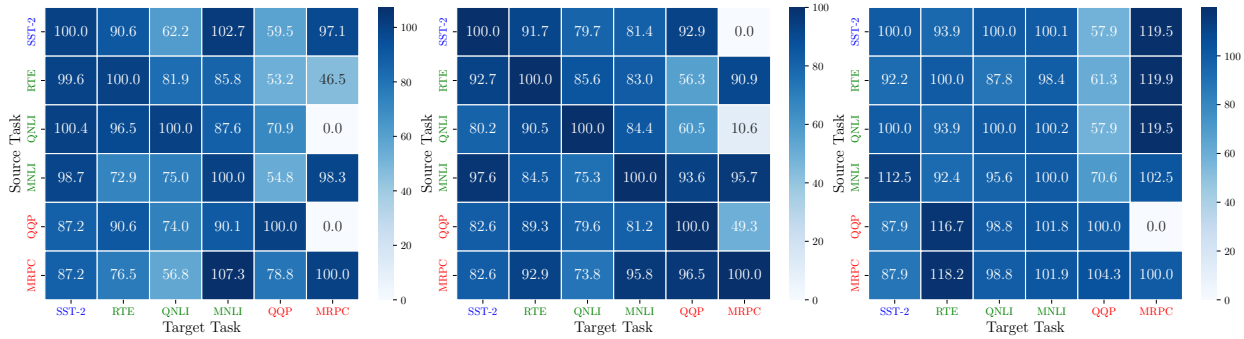
Figure 3 illustrates the results for models trained using various PEFT methods with learning rates of 1e-3 and 5e-5 for FedAvg. Notably, Adapter tuning surpasses other PEFT methods when the learning rate is set to 1e-3. Although the performance of PEFT methods generally appears to improve as the client fraction increases, as observed in Section 5.4.1, this is not always the case. Furthermore, a considerable gap in accuracy for FedAdapter is evident when comparing different learning rates, which aligns with the observation in Section 5.4.1 that the performance of PEFT methods is heavily influenced by the learning rate.



(a) MRPC (b) RTE (c) SST-2

(d) QNLI (e) MNLI (f) QQP

Figure 3: Performance evaluation of varying client fractions sampled per FL round for FedAvg. The MRPC and RTE datasets are distributed among 10 clients, while the remaining datasets are allocated to 100 clients. Here, lr denotes learning rate.



(a) Prefix Tuning (b) Adapter (c) BitFit

Figure 4: Zero-shot transferring performance of PEFT methods for FedAvg using BERT-Base PLM, learning rate = 5e-5 and Dirichlet ($\alpha = 0.1$). Here, task types are represented with the following colors: Blue: sentiment analysis, Green: natural language inference, Red: paraphrase identification.

A.3 Transferability Analysis for FedAvg

In Figure 4, we have shown the transferability analysis for FedAvg for $\alpha = 0.1$ and lr = 5e-5. Here the performance for LoRA is not shown because the model was not able to learn anything and produced the same performance for

all datasets resulting in 100% transferability for all cells in the heatmap. It can be observed in Figure 4 that the transferability is high for FedAvg. However, PEFT methods achieve lower accuracies for FedAvg when lr is 5e-5 (Table 4). Therefore, we should jointly analyze the accuracy and transferability of the model.

A.4 Transferability Analysis for Centralized Model Training

We present a transferability analysis for a model trained for 20 epochs with lr 5e-5 in a centralized setting in Figure 5. Similar to FedFAME (Figure 2), parameter-efficient finetuning in a centralized setting achieves high transferability for RTE and MRPC datasets. The model gets a more than 80% score when MRPC is a target dataset, and other datasets are source tasks. Lower accuracies for FedAvg and FedFAME and lower transferability performance on the MNLI dataset indicate that multi-genre in the dataset makes the model training difficult.

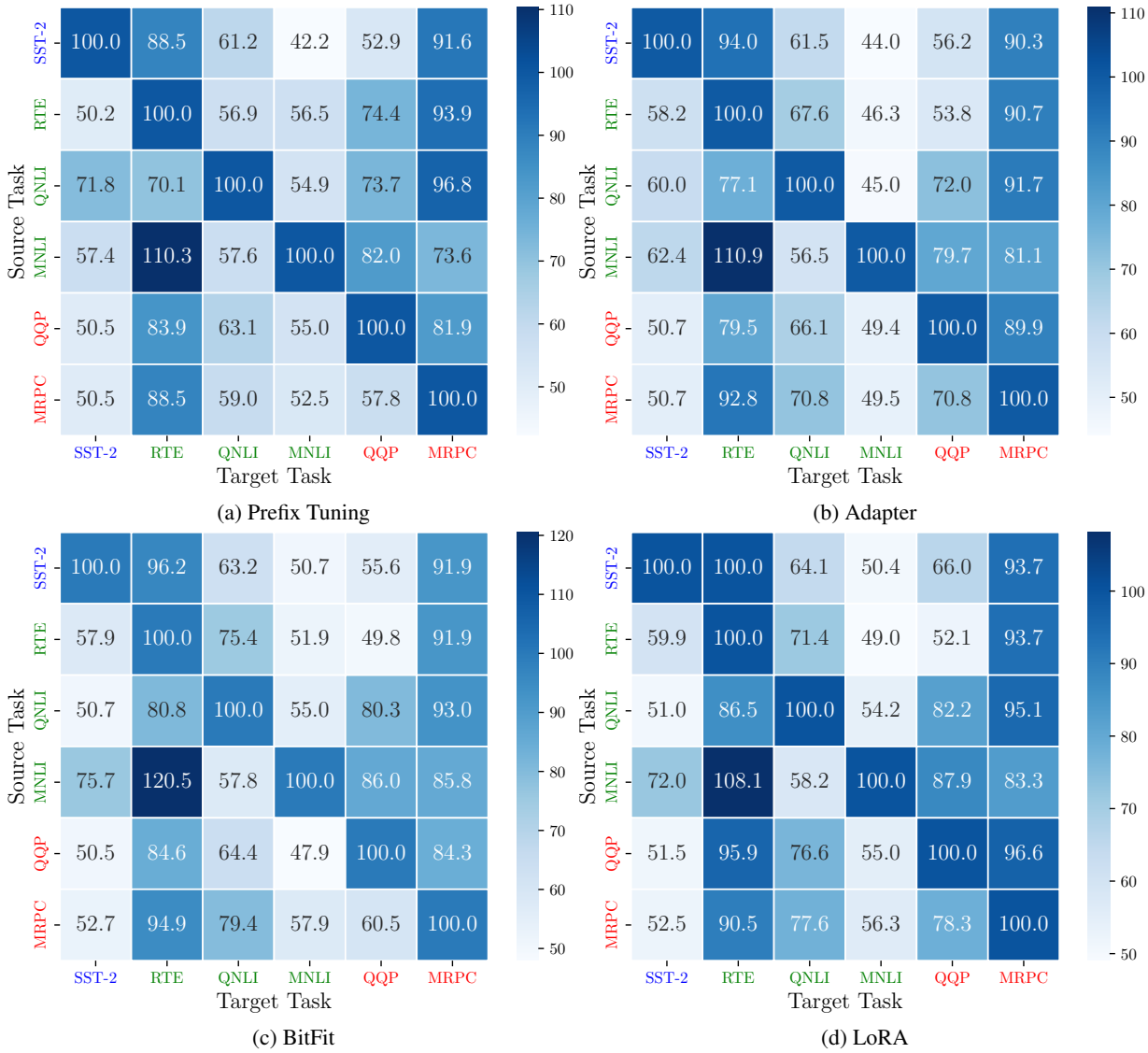


Figure 5: Zero-shot transferring performance of PEFT methods using BERT-Base PLM, for a centralized model training. Here, task types are represented with the following colors: Blue: sentiment analysis, Green: natural language inference, Red: paraphrase identification.