Federated Few-shot Learning for Mobile NLP

Dongqi Cai\textsuperscript{1}, Shangguang Wang\textsuperscript{1}, Yaozong Wu\textsuperscript{1}, Felix Xiaozhu Lin\textsuperscript{2}, Mengwei Xu\textsuperscript{1}

1 Beiyou Shenzhen Institute
2 University of Virginia
FedNLP: focus of our work

Pre-training

Cloud

Public, centralized

BERT, DistilBERT, BART, GPT, etc

Fine-tuning

Fine-tuning

Fine-tuning

Text Classification

Seq. Tagging

Text Generation

Private, geo-distributed

Deployment

Cloud&Clients

Clients
Where is the training data coming from?

FedNLP: focus of our work

Pre-training

Public, centralized

Cloud

Cloud&Clients

Private, geo-distributed

Clients

Pre-training

BERT, DistilBERT, BART, GPT, etc

Fine-tuning

Text Classification.

Fine-tuning

Seq. Tagging

Fine-tuning

Text Generation.

Fine-tuning

Private, geo-distributed
Background: Federated Few-shot Learning (FedFSL)

(a) Classic FL: rely on abundant labels

(b) Our FedFSL Scenario

Clients

Gold labels (all)

Train

Local Model

Cloud Aggregator

Train

Infer
Background: Federated Few-shot Learning (FedFSL)

(a) Classic FL: rely on abundant labels

Well-curated labeled data is scarce on mobile devices

Users lack willingness.
Users lack expertise.
Diverse NLP tasks.
Mislabels are common.
Background: Federated Few-shot Learning (FedFSL)

Well-curated labeled data is scarce on mobile devices

(a) Classic FL: rely on abundant labels

Users lack expertise.

Users lack willingness.

Mislabels are common.

(b) Our FedFSL Scenario

Diverse NLP tasks.
The rational behind pseudo labeling:

“Training with pseudo labels encourages the model to learn a decision boundary that lies in a region where the example density is lower.”

For example,
“great”:0.9, “bad”:0.1 rather than “great”:0.6, “bad”:0.4
Low class overlap ➞ Low entropy
Background: Prompt learning

- T1 (label = +1): “Most delicious pizza I’ve ever had.”
- T2 (label = -1): “You can get better sushi for half the price.”
- T3 (label = ?): Pizza was good. Not worth the price.

"It was <MASK>. Pizza was good..."
System model

Labeled Data  Unlabeled Data

Clients

Input Text
Pizza was good...
Label
+1

Loss
+1: 0.8
-1: 0.2

It was <MASK>. Pizza was good..."

great: 0.8
terrible: 0.8

Pre-trained Model
Forward  Backward  Upgrade

Fine-tuned Model

Local Prompt Training

Pseudo-labeled Data

Next Iteration ...

Upgrade
Pre-trained Model
Fine-tuned Model
System model

Labeled Data  Unlabeled Data

Clients

Input Text  Label

Pizza was good...

+1: 0.8
-1: 0.2

great: 0.8
terrible: 0.8

Loss

Pre-trained Model

Fine-tuned Model

Local Prompt Training

Pseudo-labeled Data

Next Iteration ...
System model

Labeled Data  Unlabeled Data

Clients

Input Text

Label

Pizza was good... +1

It was <MASK>. Pizza was good...

Pre-trained Model

Fine-tuned Model

Loss

+1: 0.8
terrible: 0.2
great: 0.8

Forward  Backward  Upgrade

Local Prompt Training

Next Iteration...

Pseudo-labeled Data

Aggregated model
With only 64 data labels (0.005%–0.05% of the total dataset), accuracy of RoBERTa-large is high yet huge resource cost on clients. Early experiments highlight the two sides of a coin: a satisfactory model accuracy yet huge resource cost on clients. Based on the FedFSL work, 2.4 Experimental Observations:

• the two techniques atop FedNLP, shown in Fig. 2. This en-
ow by orchestrating training and inference runtime paces.
• Notably, such a design is compatible with prior FL lit-
61
ting is assisted with prompts, provided by the trainers ei-
58
eight at least one gold or pseudo label. The on-device train-
55
ing work to ne-tune a pre-trained language model.

The updated models are then aggregated (default FedAvg); >68CB
3.M
M
80
FeS
58
 AGNEWS (skewed)  93.0 64.8±3.1 68.4±2.4 67.5±1.3 90.2±0.5
MNL1 (skewed)  85.0 37.7±5.6 42.4±5.8 42.7±6.3 77.4±1.2
YAHOO (skewed)  78.0 24.4±10.3 41.8±4.3 31.0±2.0 66.9±1.1
YELP-F (skewed)  70.0 38.3±8.8 51.2±1.8 45.7±4.4 58.2±2.4
YELP-F (uniform)  70.0 54.0±0.1 58.1±1.5 57.0±2.2 61.9±0.7

You can nd a detailed description of the datasets in Table 1 shows the convergence accuracy to get enough, correct labels, for which prompt learning can help; in turn, prompt learning's ability is limited to the task: on only one of them, the relative convergence accuracy is 40%–60%. Neither pseudo labeling nor prompt labeling are labeled. The accuracy could be further boosted by involving more data labels. Observation-1: FedFSL achieves satisfactory accuracy with different models and datasets. Batch size: 4.

Observation-2: FedFSL incurs huge system cost. For example, training takes 3.3 hours to converge, 7.3 hours, and 1.5 hours, respectively.

Inference runtime 28 hours, 5.8 hours, and 1.3 hours, respectively.

Communication cost 10.4 GB peak memory. The cost is about 1.4 million Joules of energy, 68.4 GBs of network transmission, 61.9 million Joules of energy, 10.4 GBs of network transmission, respectively.

Preliminary: FedFSL performance

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Full-set (oracle)</th>
<th>Vanilla-FedFSL</th>
<th>Prompt-Only</th>
<th>Pseudo-Only</th>
<th>Both (Ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGNEWS (skewed)</td>
<td>93.0</td>
<td>64.8±3.1</td>
<td>68.4±2.4</td>
<td>67.5±1.3</td>
<td>90.2±0.5</td>
</tr>
<tr>
<td>MNLI (skewed)</td>
<td>85.0</td>
<td>37.7±5.6</td>
<td>42.4±5.8</td>
<td>42.7±6.3</td>
<td>77.4±1.2</td>
</tr>
<tr>
<td>YAHOO (skewed)</td>
<td>78.0</td>
<td>24.4±10.3</td>
<td>41.8±4.3</td>
<td>31.0±2.0</td>
<td>66.9±1.1</td>
</tr>
<tr>
<td>YELP-F (skewed)</td>
<td>70.0</td>
<td>38.3±8.8</td>
<td>51.2±1.8</td>
<td>45.7±4.4</td>
<td>58.2±2.4</td>
</tr>
<tr>
<td>YELP-F (uniform)</td>
<td>70.0</td>
<td>54.0±0.1</td>
<td>58.1±1.5</td>
<td>57.0±2.2</td>
<td>61.9±0.7</td>
</tr>
</tbody>
</table>
With only 64 data labels (0.005%–0.05% of the total dataset), learning are indispensable.

scarce data labels; for which both pseudo labeling and prompt accuracy yet huge resource cost on clients.

Based on the FedFSL work

2.4 Experimental Observations

• residing in a central server.

much larger set of unlabeled samples has a tiny training set with labels

ow is the algorithmic foundation of our future

ow presented above, we perform

The updated models are then aggregated (default FedAvg

Training runtime

cate how inference runtime paces.

Notably, such a design is compatible with prior FL lit-

erature on client/data sampling [6, 88, 91].

enhancements [61, 47].

The on-device train-

me has a tiny training set with labels

A mechanism to orches-

raneous round will also be re-labeled to avoid forgetting

pseudo label

can help; in turn, prompt learning's ability is limited to the

racy to get enough, correct labels, for which prompt learning

other: pseudo labeling heavily relies on the initial model accu-

Table 1 shows the convergence

 Johannesburg, South Africa. 1

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Full-set (oracle)</th>
<th>Vanilla-FedFSL</th>
<th>Prompt-Only</th>
<th>Pseudo-Only</th>
<th>Both (Ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGNEWS (skewed)</td>
<td>93.0</td>
<td>64.8±3.1</td>
<td>68.4±2.4</td>
<td>67.5±1.3</td>
<td>90.2±0.5</td>
</tr>
<tr>
<td>MNLI (skewed)</td>
<td>85.0</td>
<td>37.7±5.6</td>
<td>42.4±5.8</td>
<td>42.7±6.3</td>
<td>77.4±1.2</td>
</tr>
<tr>
<td>YAHOO (skewed)</td>
<td>78.0</td>
<td>24.4±10.3</td>
<td>41.8±4.3</td>
<td>31.0±2.0</td>
<td>66.9±1.1</td>
</tr>
<tr>
<td>YELP-F (skewed)</td>
<td>70.0</td>
<td>38.3±8.8</td>
<td>51.2±1.8</td>
<td>45.7±4.4</td>
<td>58.2±2.4</td>
</tr>
<tr>
<td>YELP-F (uniform)</td>
<td>70.0</td>
<td>54.0±0.1</td>
<td>58.1±1.5</td>
<td>57.0±2.2</td>
<td>61.9±0.7</td>
</tr>
</tbody>
</table>

How about the system cost?

Satisfactory accuracy

Pseudo labeling

Prompt learning
Challenge: FedFSL system cost

Training efficiency

AdaFL: Efficient FL
Challenge: FedFSL system cost

Training efficiency
- Adapter
- LoRA
- BitFit
- ...

Labeling redundancy
- >87%

Labeling-training pace
- Number?
- Time?
- Frequency?

AdaFL: Efficient FL
- 2021.12
- 2022.08

FeS: Train without labels
- 2023.03
Design 1: Representational Filtering
Design: Representational Filtering

Client Data → Pseudo labeling (Inference) → Client Data

Confidence > threshold?

In vain

87%

Client Data → Subsequent training
Design: Representational Filtering

Client Data → Pseudo labeling (Inference) → Client Data

Transformer
Add & Norm
Feed Forward
Add & Norm
Attention

Client Data

87%
In vain
Confidence > threshold?

Client Data

Subsequent training

Client Data

Subsequent training

Client Data

Proxy model
Diversity penalty
Repr. score
Offline filtering
Design: Curriculum Pacing

Inference: Clients with prompts and a few/zero labeled data are used for inference. The model infers the labels of the unlabeled data based on the prompts and the few/zero labels. Updated attributes are used to train a new classification layer.

Training: For each iteration, the trainer updates the model with the updated labels from the clients. The updated model is then used for inference on the next iteration.

Updated model for the next iteration:

- $i_{th}$ iteration: Model inferred labels for unlabeled data.
- $i_{th}+1$ iteration: Model predicts labels for new unlabeled data using the updated model.
Design: Curriculum Pacing

**Inference**
- Clients
- # of labels

**Training**
- Updated model
- Updated labels

**Design: Curriculum Pacing**

- **f**: frequency of updating pseudo labels
- **n**: number of clients selected to perform pseudo labeling
- **k**: ratio of selected pseudo labels for the subsequent training

Pacing configuration \(<f,n,k>\)
Design: Curriculum Pacing

Inference

- $i_{th}$ iteration
- Updated model
- Clients
- # of labels

Training

- $i_{th}+1$ iteration
- Updated labels
- $j_{th}+f$ iteration
- $j_{th}$ iteration
- Updated model
- Clients
- # of labels

- $f$: frequency of updating pseudo labels
- $n$: number of clients selected to perform pseudo labeling
- $k$: ratio of selected pseudo labels for the subsequent training

Pacing configuration $<f,n,k>$
Design: Curriculum Pacing

Inference

\[ i_{th} + 1 \text{ iteration} \]

\[ i_{th} \text{ iteration} \]

Training

\[ j_{th} + f \text{ iteration} \]

\[ j_{th} \text{ iteration} \]

- **f**: frequency of updating pseudo labels
- **n**: number of clients selected to perform pseudo labeling
- **k**: ratio of selected pseudo labels for the subsequent training

Pacing configuration \(<f, n, k>\)
• Progressively speed up the pseudo labeling speed, i.e., adding more pseudo labels at a higher frequency.
• Progressively speed up the pseudo labeling speed, i.e., adding more pseudo labels at a higher frequency.

• Progressive upgrading is only a coarse-grained plan, how to control the pace more concisely?
Design: Curriculum Pacing

Augment efficiency (AUG-E):
measure the gradient of the time-to-accuracy curve to search for an effective configuration with low cost

\[ AUG - E(f, n, k) \leftarrow \frac{\eta \Delta(acc)}{C_{\text{infer}}(f, n) + \theta \cdot C_{\text{train}}(k)} \]

Our system selects a configuration with best AUG-E from a candidate list (hand-picked through extensive offline experiments) for future pseudo labeling.
**Design: Curriculum Pacing**

**Augment efficiency (AUG-E):**
measure the gradient of the time-to-accuracy curve to search for an effective configuration with low cost

\[
AUG - E(f, n, k) \rightarrow \frac{\eta \Delta(acc)}{C_{\text{infer}}(f, n) + \theta \cdot C_{\text{train}}(k)}
\]

- **Our system** selects a configuration with **best AUG-E** from a candidate list (hand-picked through extensive offline experiments) for future pseudo labeling.
Evaluation: Setup

- **Implementation**
  - FedNLP\(^1\)
  - PET\(^2\)

- **Setups**
  - 2 devices (TX2, RPI 4B)
  - 2 models (RoBERTa-base & large)
  - 4 datasets

- **Baselines**
  1. Vanilla Fine-Tuning (FedCLS)
  2. Vanilla Few-shot Tuning (FedFSL)
  3. Vanilla Few-shot Tuning + Bias-tuning (FedFSL-BIAS)

<table>
<thead>
<tr>
<th>Dataset</th>
<th>AGNEWS (^{[108]})</th>
<th>MNLI (^{[89]})</th>
<th>YAHOO (^{[108]})</th>
<th>YELP-F (^{[108]})</th>
</tr>
</thead>
<tbody>
<tr>
<td># Training</td>
<td>120k</td>
<td>392.7k</td>
<td>1.4M</td>
<td>650k</td>
</tr>
<tr>
<td># Test</td>
<td>7.6k</td>
<td>9.8k</td>
<td>60k</td>
<td>50k</td>
</tr>
<tr>
<td># Clients</td>
<td>100</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>Distribution</td>
<td>Skewed</td>
<td>Uniform</td>
<td>Skewed</td>
<td>Skewed</td>
</tr>
<tr>
<td>Labels</td>
<td>64</td>
<td>64</td>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>Prompt</td>
<td>a ____ b</td>
<td>a ?____, b</td>
<td>Category: a ____ b</td>
<td>It was _____. a</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Setup</th>
<th>Labeling</th>
<th>Training</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pacing</td>
<td>Optimization</td>
</tr>
<tr>
<td>FedCLS</td>
<td>/</td>
<td>/</td>
</tr>
<tr>
<td>FedFSL</td>
<td>Static</td>
<td>/</td>
</tr>
<tr>
<td>FedFSL-BIAS</td>
<td>Static</td>
<td>/</td>
</tr>
<tr>
<td>FeS (Ours)</td>
<td>Curriculum ((§3.1))</td>
<td>Filtering ((§3.2))</td>
</tr>
</tbody>
</table>

---


64 labels in total instead of per client
Evaluation: End-to-end Performance

- Our system significantly speeds up model convergence at high accuracy.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>AGNEWS</th>
<th>MNLI</th>
<th>YAHOO</th>
<th>YELP-F</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>TX2</td>
<td>RPI</td>
<td>TX2</td>
</tr>
<tr>
<td>FedCSL</td>
<td>27.9%</td>
<td>X</td>
<td>X</td>
<td>37.3%</td>
</tr>
<tr>
<td>FedFSL</td>
<td>92.5%</td>
<td>3.3</td>
<td>3.3</td>
<td>50.0</td>
</tr>
<tr>
<td>FedFSL-BIAS</td>
<td>92.5%</td>
<td>1.7</td>
<td>1.7</td>
<td>25.0</td>
</tr>
<tr>
<td>Ours</td>
<td>95.9%</td>
<td>0.4</td>
<td>0.4</td>
<td>5.5</td>
</tr>
</tbody>
</table>

Table 1: The final convergence accuracy (“Conv. Acc.”) and the elapsed training time (“Time-to-acc”) to reach different relative accuracy. “acc1”/“acc2” are the final convergence accuracy of FedFSL/FedFSL-BIAS, respectively. “X” means the accuracy cannot be achieved.
Evaluation: Key Design

- Our key designs contribute to the results significantly.

Fig. 1: Model convergence delays with and without Our system’s key designs, showing their significance. DC: training depth/capacity co-planning; RF: representative filtering; CP: curriculum pacing.
Evaluation: System Cost

Our system is resource-efficient.
- It saves up to 3000.0× network traffic. (Fig. 1)
- It reduces up to 41.2× energy consumption. (Fig. 2)
- It reduces the memory usage by 4.5×. (Fig. 3)

Fig. 1: The total network traffic of all clients.

Fig. 2: The total energy consumption of all clients, normalized to that of ours
Conclusion

• Our system is a FedFSL framework that enables practical few-shot NLP fine-tuning on federated mobile devices.
Conclusion

• Our system is a FedFSL framework that enables practical few-shot NLP fine-tuning on federated mobile devices.
• It incorporates pseudo labeling and prompt learning to achieve usable accuracy with only tens of data labels.

Code: https://github.com/UbiquitousLearning/FeS
Conclusion

- Our system is a FedFSL framework that enables practical few-shot NLP fine-tuning on federated mobile devices.
- It incorporates pseudo labeling and prompt learning to achieve usable accuracy with only tens of data labels.
- At system aspect, it proposes three novel techniques, i.e., early filtering unlabeled data, reducing the tuning depth/capacity, and curriculum orchestrate them to address the unique challenge of huge resource cost raised by its algorithmic.

Code: https://github.com/UbiquitousLearning/FeS
Conclusion

• Our system is a FedFSL framework that enables practical few-shot NLP fine-tuning on federated mobile devices.

• It incorporates pseudo labeling and prompt learning to achieve usable accuracy with only tens of data labels.

• At system aspect, it proposes three novel techniques, i.e., early filtering unlabeled data, reducing the tuning depth/capacity, and curriculum orchestrate them to address the unique challenge of huge resource cost raised by its algorithmic.

• Compared to vanilla FedFSL, Our system reduces the training delay, client energy, and network traffic by up to 46.0×, 41.2× and 3000.0×, respectively.

Code: https://github.com/UbiquitousLearning/FeS
Concluding Remarks by Mengwei

- The recent AI wave (large, foundational, multimodal models) is going to make another Golden Era for mobile computing.
  - Think of Smartphones/IoTs as humans-level assistants

- Two key research directions
  - Making LLMs run fast and learn rapidly on devices (hw-sw-algo. codesign)
  - Building killer apps atop LLMs (agents, searching, AIGC, etc)

- Open to collaboration and debate!
  - Who are we: a junior faculty plus a group of passionate graduate students who believe in LLM as a game changer to mobile research
Appendix for Q&A
Different parameter-efficient methods

• Adapter is not only for "adapters".

• Parameter-efficient methods are unified (He, ICLR’22).

• Bias-tuning provides the best accuracy-efficiency tradeoff under few-shot learning scenarios (Logan, ACL’22).

Design 2: Training Depth/Capacity Co-planning

(a) Layer-freeze

(b) Bias-tuning

(c) Ours
With only 64 data labels (0.005%–0.05% of the total dataset), a set of early experiments on its performance. The results can help; in turn, prompt learning’s ability is limited to the tasks where it is assisted with prompts, provided by the trainers especially when the global model is still weak. A mechanism to orchestrate training and inference runtime paces.

Training runtime is assisted with prompts, provided by the trainers especially when the global model is still weak. A mechanism to orchestrate training and inference runtime paces. The updated models are then aggregated (default FedAvg). The above hyper-parameters are added as training samples. In subsequent rounds, pseudo labels are treated equally as the training labels.

**2.4 Experimental Observations**

- **Observation-1:** FedFSL achieves satisfactory accuracy with much larger set of unlabeled samples when the global model is still weak. A mechanism to orchestrate training and inference runtime paces.

### Table: Preliminary FedFSL performance and cost

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Full-set (oracle)</th>
<th>Vanilla-FedFSL</th>
<th>Prompt-Only</th>
<th>Pseudo-Only</th>
<th>Both (Ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGNEWS (skewed)</td>
<td>93.0</td>
<td>64.8±3.1</td>
<td>68.4±2.4</td>
<td>67.5±1.3</td>
<td>90.2±0.5</td>
</tr>
<tr>
<td>MNLI (skewed)</td>
<td>85.0</td>
<td>37.7±5.6</td>
<td>42.4±5.8</td>
<td>42.7±6.3</td>
<td>77.4±1.2</td>
</tr>
<tr>
<td>YAHOO (skewed)</td>
<td>78.0</td>
<td>24.4±10.3</td>
<td>41.8±4.3</td>
<td>31.0±2.0</td>
<td>66.9±1.1</td>
</tr>
<tr>
<td>YELP-F (skewed)</td>
<td>70.0</td>
<td>38.3±8.8</td>
<td>51.2±1.8</td>
<td>45.7±4.4</td>
<td>58.2±2.4</td>
</tr>
<tr>
<td>YELP-F (uniform)</td>
<td>70.0</td>
<td>54.0±0.1</td>
<td>58.1±1.5</td>
<td>57.0±2.2</td>
<td>61.9±0.7</td>
</tr>
</tbody>
</table>

Satisfactory accuracy and prompt learning are indispensable.

### Graph: Mem & Latency of MobiCom (Jetson)

- **OOM:** Excessive on-device **inference.**
- **Prompt learning** needs large NLP model.
- Sophisticated **orchestration** workflow.
Paths towards practical federated learning

Solution
- Activation cache
- Configurator

Key block
- Adapters

Challenge
- Network transmission

Challenges
- Data labels
- Curriculum pacing
- Tuning co-planner
- Representative filter
- Prompt learning
- Pseudo learning

AdaFL: Efficient FL
2021.12

FeS: Few-shot FL
2022.08 2023.03