PaddleFSL: A General Few-Shot Learning Toolbox in Python

Zhenqian Shen$^{1,2}$  
Yaqing Wang$^2$*  
Haoyi Xiong$^2$  
Xin Tian$^2$  
Zeyu Chen$^2$  
Quanming Yao$^1$  
Deijing Dou$^2$

$^1$Department of Electronic Engineering, Tsinghua University  
$^2$Baidu Research, Baidu Inc., Beijing, China  
*Correspondence to

Abstract

Few-Shot Learning (FSL), a type of machine learning problems which targets at generalizing from a few labeled data, is now an active area in both academia and industry. In this paper, we present a general few-shot learning toolbox in Python called PaddleFSL. PaddleFSL provides various FSL solutions which are applicable to diverse applications. It also contains detailed annotations and tutorial examples, such that users can easily develop and compare different FSL solutions. Furthermore, PaddleFSL is easy to be deployed on various training platforms. The project is available at https://github.com/tata1661/FSL-Mate/tree/master/PaddleFSL.

Keywords: Few-Shot Learning, One-Shot Learning, Meta Learning, Python Toolbox, PaddlePaddle

1. Introduction

Nowadays, over-parameterized deep models, of which parameter size is significantly larger than the sample size, has demonstrated high expressiveness and obtained state-of-the-art performance in many applications (He et al., 2016; Jumper et al., 2021). However, training these models require large scale and high quality labeled data which are costly and difficult to acquire. To handle this problem, Few-Shot Learning (FSL), which targets at generalizing from only a limited number of examples (Wang et al., 2020), rapidly becomes an active research area. Many FSL methods are proposed and obtained good performance, e.g., Siamese Network (Siamese) (Koch, 2015), Prototypical Network (ProtoNet) (Snell et al., 2017), Model-Agnostic Meta-Learning (MAML) (Finn et al., 2017), Relation Network (RelationNet) (Sung et al., 2018), and Almost No Inner Loop (ANIL) (Raghu et al., 2020). However, applying these methods to real-world applications faces many serious practical challenges. One has to first collect and prepare appropriate datasets, tries and selects among different models, tunes hyperparameters hard, and finally manages to evaluate the
To narrow down the gap between FSL research advances and their practical usage, we introduce a FSL toolbox in Python called PaddleFSL. It is built upon PaddlePaddle (Ma et al., 2019), which is an industrial deep learning platform with advanced technologies and rich features serving more than 4 million developers. Table 1 shows the comparison between PaddleFSL and relevant toolboxes, e.g. learn2learn (Arnold et al., 2020), Torchmeta (Deleu et al., 2019), keras-fsl (ker), mmfewshot (mmf), and Libfewshot (Li et al., 2021). Compared to these toolboxes, PaddleFSL covers broader application scenarios which include Computer Vision (CV), Natural Language Processing (NLP) and Bioinformatics (BIO). PaddleFSL can be used on various platforms including CPUs, GPUs and Clusters, while other toolboxes cannot be used on clusters. It also provides unit tests of different implementations and detailed documents. Up to now, PaddleFSL has 24441 downloads, 1247 stars and 240 forks. We continuously contribute to PaddleFSL with the vision of making it a general FSL toolbox which can help users to conquer few-shot problems in diverse applications.

2. Software Architecture

Figure 1 plots the software architecture of PaddleFSL, where we specially show the workflow of building a FSL solution using the components of PaddleFSL. **Dataset** provides various publicly accessible benchmark datasets of multiple applications. Note that PaddleFSL also
supports customized datasets of users. For example, users can create their own image or text dataset by inheriting CVDataset or NLPDataset in the Dataset. **Backbone** consists of multiple representation learning models to extract sample representation, which includes both models trained from scratch like convolution neural networks (CNN) (Krizhevsky et al., 2012; He et al., 2016) and graph neural networks (GNN) (Kipf and Welling, 2016), and pretrained models like BERT (Kenton and Toutanova, 2019) and ERNIE (Sun et al., 2019). **Model** provides classical FSL models which particularly deal with the lack of labeled data, e.g. Siamese and ProtoNet. **Algorithm** contains classical FSL algorithms such as MAML and ANIL to optimize the learning parameters efficiently. Relevant functions for performance evaluation are also provided. **Application** consists of our exemplar applications ranging from CV, NLP to BIO. **Platform** supports deployment of PaddleFSL on CPU, GPU and clusters. Users can freely select components of PaddleFSL to build their FSL solutions following the examples and documents.

3. Usage Example

Typically, FSL solutions adopts episode training (Vinyals et al. 2016), which uses a collection of tasks to train a predictive model which can apply for target tasks with few labeled examples. In few-shot classification problem, each task $T_\tau$ is a $N$-way $K$-shot classification task with a support set $S_\tau = \{(x_{\tau,i}, y_{\tau,i})\}_{i=1}^{NK}$ containing $N \times K$ labeled examples and a query set $Q_\tau = \{(x_{\tau,j}, y_{\tau,j})\}_{j=1}^{N_q}$ containing $N_q^2$ unlabeled examples to be predicted. Listing 1 shows a usage example of PaddleFSL in few-shot image classification tasks. One first needs to select a dataset from `paddlefsl.datasets`, a model from `paddlefsl.backbone`, and an algorithm from `paddlefsl.model_zoo`. After setting up datasets, backbone and model, one then can call the functions `meta_training()` to determine the learning parameters, and obtain the trained model `MODEL` and its path `train_dir`. Finally, one can evaluate the model performance on target tasks by calling the function `meta_testing()`.
Listing 1: Usage example of PaddleFSL on few-shot image classification tasks.

4. Experimental Results

Table 2 shows some experimental results in CV, NLP and BIO. PaddleFSL provides comparable reproduced results and a fair comparison between different algorithms. When running on different platforms, PaddleFSL obtains similar results.

<table>
<thead>
<tr>
<th>Application</th>
<th>Dataset</th>
<th>Algorithm</th>
<th>Backbone</th>
<th>Platform</th>
<th>Way</th>
<th>Shot</th>
<th>Result</th>
<th>ACC(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CV</td>
<td>mini-ImageNet</td>
<td>ProtoNet</td>
<td>CNN</td>
<td>GPU</td>
<td>5</td>
<td>1</td>
<td>reported</td>
<td>44.61 ± 0.78</td>
</tr>
<tr>
<td>CV</td>
<td>Omniglot</td>
<td>ProtoNet</td>
<td>CNN</td>
<td>GPU</td>
<td>5</td>
<td>1</td>
<td>ours</td>
<td>49.75 ± 0.25</td>
</tr>
<tr>
<td>CV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>reported</td>
<td>97.4</td>
</tr>
<tr>
<td>CV</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ours</td>
<td>99.45 ± 0.04</td>
</tr>
<tr>
<td>NLP</td>
<td>FewRel</td>
<td>ProtoNet</td>
<td>CNN</td>
<td>GPU</td>
<td>5</td>
<td>5</td>
<td>ours</td>
<td>83.13 ± 0.42</td>
</tr>
<tr>
<td>NLP</td>
<td></td>
<td>Siamese</td>
<td>CNN</td>
<td>GPU</td>
<td>5</td>
<td>1</td>
<td>72.07 ± 0.38</td>
<td></td>
</tr>
<tr>
<td>NLP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>5</td>
<td>ours</td>
<td>80.05 ± 0.28</td>
</tr>
<tr>
<td>BIO</td>
<td>Tox21</td>
<td>MAML</td>
<td>GNN</td>
<td>CPU</td>
<td>2</td>
<td>10</td>
<td>ours</td>
<td>81.35 ± 0.73</td>
</tr>
<tr>
<td>BIO</td>
<td></td>
<td></td>
<td></td>
<td>Cluster</td>
<td></td>
<td></td>
<td></td>
<td>81.57 ± 1.03</td>
</tr>
</tbody>
</table>

Table 2: Experiment results obtained in different applications.

5. Conclusion and Future Plans

PaddleFSL is a popularly used general FSL toolbox based on Python and PaddlePaddle, which provides classical FSL algorithms, supports diverse applications, and offers detailed documents and examples. In the future, we will continuously update PaddleFSL, incorporating more FSL algorithms, and expanding its application scenarios. We hope that PaddleFSL can help users from both academia and industry to easily handle FSL problems, and contribute to the advances of FSL research.
References


