Journal of Machine Learning Research 1 (2000) 1-48 Submitted 4/00; Published 10/00

PaddleFSL: A General Few-Shot Learning Toolbox in Python

Zhenqian Shen[1,2] szg18@mails.tsinghua.edu.cn
Yaqing Wang[2*] wangyaqing01@baidu.com
Haoyi Xiong[2] xionghaoyi@baidu.com
Xin Tian[2] tianxin04@baidu.com
Zeyu Chen[2] chenzeyu01@baidu.com
Quanming Yao[1] qyaoaa@mail.tsinghua.edu.cn
Dejing Dou[2] doudejing@baidu.com
[1]*Department of Electronic Engineering, Tsinghua University*
[2]*Baidu Research, Baidu Inc., Beijing, China*
[*]*Correspondence to*

Editor:

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

			learn2learn	Torchmeta	keras-fsl	mmfewshot	Libfewshot	PaddleFSL
Application	CV	MAML	✔	✔	✗	✔	✔	✔
		ProtoNet	✔	✔	✗	✔	✔	✔
		RelationNet	✗	✗	✗	✔	✔	✔
	NLP	Siamese	✗	✗	✔	✗	✗	✔
		PET	✗	✗	✗	✗	✗	✔
		P-Tuning	✗	✗	✗	✗	✗	✔
	BIO	MAML	✗	✗	✗	✗	✗	✔
		PAR	✗	✗	✗	✗	✗	✔
Unit Test			✔	✔	✗	✔	✗	✔
Document			✔	✔	✔	✔	✔	✔
Platform		CPU	✔	✔	✔	✔	✔	✔
		GPU	✔	✔	✔	✔	✔	✔
		Cluster	✗	✗	✗	✗	✗	✔
Activity		Download	45182	131963	5484	913	N/A	24441
		Star	1725	1576	172	359	496	1247
		Fork	249	203	29	42	87	240

Table 1: Comparison of PaddleFSL with other popular FSL toolboxes. The activity statistics is collected from the respective GitHub pages on 2022/3/18.

performance fairly. All of these procedures involve different difficulties, which are particularly hard for beginners.

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 can not 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

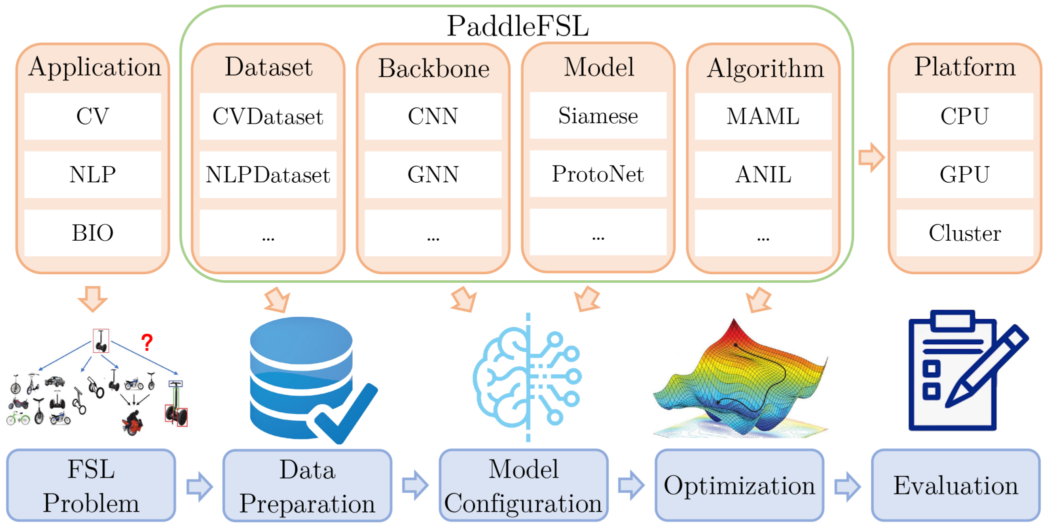

Figure 1: Software architecture of PaddleFSL.

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_τ 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 $Q_\tau = \{(x_{\tau,j}, y_{\tau,j})\}_{j=1}^{N_\tau^q}$ containing N_τ^q 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()`.

```python
import paddle
import paddlefsl
from paddlefsl.model_zoo import maml
# Platform selection
paddle.set_device('gpu:0')
# Dataset preparation
TRAIN_DATASET = paddlefsl.datasets.MiniImageNet(mode='train')
VALID_DATASET = paddlefsl.datasets.MiniImageNet(mode='valid')
TEST_DATASET = paddlefsl.datasets.MiniImageNet(mode='test')
# Model configuration
MODEL = paddlefsl.backbones.Conv(input_size=(3, 84, 84), output_size=5)
# Algorithm selection, meta training and meta testing
train_dir, MODEL = maml.meta_training(model=MODEL, train_dataset=
    TRAIN_DATASET, valid_dataset=VALID_DATASET, ways=5, shots=1)
maml.meta_testing(model=MODEL, test_dataset=TEST_DATASET, ways=5, shots=1)
```

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.

Application	Dataset	Algorithm	Backbone	Platform	Way	Shot	Result	ACC(%)
CV	*mini*-ImageNet	ProtoNet	CNN	GPU	5	1	reported	44.61 ± 0.78
							ours	49.75 ± 0.25
	Omniglot						reported	97.4
							ours	99.45 ± 0.04
NLP	FewRel	ProtoNet	CNN	GPU	5	1	ours	70.18 ± 0.35
					5	5		83.13 ± 0.42
		Siamese			5	1		72.07 ± 0.38
					5	5		80.05 ± 0.28
BIO	Tox21	MAML	GNN	CPU	2	10	ours	81.35 ± 0.73
				GPU				81.24 ± 0.58
				Cluster				81.57 ± 1.03

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

keras-fsl. https://github.com/few-shot-learning/Keras-FewShotLearning. Accessed: 2022-03-18.

OpenMMLab few shot learning toolbox and benchmark. https://github.com/open-mmlab/mmfewshot. Accessed: 2022-03-18.

Sébastien MR Arnold, Praateek Mahajan, Debajyoti Datta, Ian Bunner, and Konstantinos Saitas Zarkias. learn2learn: A library for meta-learning research. *arXiv preprint arXiv:2008.12284*, 2020.

Tristan Deleu, Tobias Würfl, Mandana Samiei, Joseph Paul Cohen, and Yoshua Bengio. Torchmeta: A meta-learning library for pytorch. *arXiv preprint arXiv:1909.06576*, 2019.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pages 1126–1135, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, 2021.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2016.

Gregory Koch. *Siamese neural networks for one-shot image recognition*. PhD thesis, University of Toronto, 2015.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

Wenbin Li, Chuanqi Dong, Pinzhuo Tian, Tiexin Qin, Xuesong Yang, Ziyi Wang, Jing Huo, Yinghuan Shi, Lei Wang, Yang Gao, et al. LibFewShot: A comprehensive library for few-shot learning. *arXiv preprint arXiv:2109.04898*, 2021.

Yanjun Ma, Dianhai Yu, Tian Wu, and Haifeng Wang. PaddlePaddle: An open-source deep learning platform from industrial practice. *Frontiers of Data and Domputing*, 1(1): 105–115, 2019.

Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? Towards understanding the effectiveness of MAML. In *International Conference on Learning Representations*, 2020.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087, 2017.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. ERNIE: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*, 2019.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1199–1208, 2018.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, pages 3630–3638, 2016.

Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys*, 53(3):1–34, 2020.