DEEP LEARNING FRAMEWORK MINDSPORE AND PYTORCH COMPARISON

Xiangyu XIA and Shaoxiang ZHOU

Department of Information, Beijing City University, Beijing 100191, China

ABSTRACT

Deep learning has been well used in many fields. However, there is a large amount of data when training neural networks, which makes many deep learning frameworks appear to serve deep learning practitioners, providing services that are more convenient to use and perform better. MindSpore and PyTorch are both deep learning frameworks. MindSpore is owned by HUAWEI, while PyTorch is owned by Facebook. Some people think that HUAWEI's MindSpore has better performance than FaceBook's PyTorch, which makes deep learning practitioners confused about the choice between the two. In this paper, we perform analytical and experimental analysis to reveal the comparison of training speed of MIndSpore and PyTorch on a single GPU. To ensure that our survey is as comprehensive as possible, we carefully selected neural networks in 2 main domains, which cover computer vision and natural language processing (NLP). The contribution of this work is twofold. First, we conduct detailed benchmarking experiments on MindSpore and PyTorch to analyze the reasons for their performance differences. This work provides guidance for end users to choose between these two frameworks.

KEYWORDS

Deep Learning, Performance, Mindspore, Pytorch, Comparison.

1. INTRODUCTION

In recent years, machine learning has been used in various fields. Deep learning is a machine learning method that has many applications in computer vision, speech recognition, natural language processing and other fields. Deep learning relies on the number of neurons and layers of the neural network. With the continuous improvement of the depth of the network and the number of neurons, the relative accuracy will be higher. From traditional fully connected neural networks to convolutional neural networks and recurrent recurrent neural networks to graph neural networks and transformers, the depth and complexity of networks are getting higher and higher, and the types of networks are constantly changing. The more complex the network, the more parameters for training, which requires huge computing power, which requires a large number of high-performance computing cards to accelerate, such as GPU, TPU [1] and FPGA. Among these hardware options, GPU is the most popular choice. As neural networks become more and more complex, the continuous upgrade and iteration of computing cards has prompted developers of deep learning frameworks to provide users of deep learning with better usability and higher performance. Deep Learning Framework.

There are currently many deep learning frameworks, including Caffe [2] developed by UC Berkeley, TensorFlow [3] developed by Google, PyTorch[4] developed by Facebook, CNTK[5] developed by Microsoft and MindSpore by HUAWEI.

David C. Wyld et al. (Eds): EMSA, SEA, AIFU, NLCAI, BDML, BIoT, NCOM, CLOUD, CCSEA, SIPRO - 2022
pp. 191-196, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.120617

At present, there are two execution modes of mainstream deep learning frameworks, namely static graph mode and dynamic graph mode. Static graph mode has high training performance but is difficult to debug. Although the dynamic graph mode is easier to debug than the static graph mode, it is difficult to execute efficiently. Currently, PyTorch is the most popular framework in academia. PyTorch is popular for its ease of programming and debugging, using both static and dynamic computational graphs. Programming in PyTorch is more dynamic, with users defining and executing graph nodes during execution. PyTorch can suffer from Python interpreter overhead. MindSpore provides a unified coding method for dynamic graphs and static graphs, which greatly increases the compatibility of static graphs and dynamic graphs. Users do not need to develop multiple sets of codes, and can switch the dynamic graph/static graph mode only by changing one line of code.

There are many studies evaluating various deep learning frameworks on different hardware. However, they usually only focus on the output results of the frameworks to compare these deep learning frameworks, rather than systematically explaining the deep reasons for the output. The reason behind these two frameworks is rarely discussed because, due to their completely different software stacks (including the way computation graphs are constructed, runtime scheduling), it is difficult to compare the two frameworks.

In this article, our purpose is to deeply compare the performance differences between PyTorch and MindSpore under a single GPU. To make the work as comprehensive as possible, we selected 3 very classic neural networks, including CNN, RNN, and Transformer, which cover the fields of computer vision and natural language processing.

2. BACKGROUND

2.1. Neural Networks and Deep Learning

The emergence of neural networks has enabled the rapid development of deep learning and has received extensive attention in the field of artificial intelligence (AI). Beginning with AlexNet [6], various DNN architectures (GoogLeNet [7], ResNet [8]) have emerged one after another in a short period of time, providing better feature detection and accuracy. Typically, a DNN structure consists of an input layer, an output layer, and multiple hidden layers. These layers can be viewed as a set of operations. Some frameworks use layer abstraction, while others and TensorFlow use operator abstraction. There are different types of layers for different applications and purposes, such as convolutional, pooling and activation layers for feature extraction in image classification, attention layers for information filtering in NLP, and LSTM layer. Combinations of layers can be explored to meet the needs of an application, such as ResNet, even when existing layers are considered.

The purpose of deep learning training is to find a suitable set of model parameters to minimize the loss function, which reflects the error between the predicted result of the sample and the ground truth label. The training process usually consists of millions of iterations, each of which involves two computationally intensive stages, forward and backward propagation. In forward propagation, the training samples are input to the input layer, and the weights and biases are added to calculate the output feature map as the input of the next layer. Finally, the loss is calculated by comparing the output to the ground truth labels of the output layer, ending the forward pass. Backpropagation starts from the output layer, traverses each layer in reverse, calculates the gradient of the parameters of each layer through the chain rule according to the loss value, and optimizes the parameters. There are many optimizers for backpropagation such as Stochastic Gradient Descent (SGD), Momentum and Adam. In general, the loss value gets

smaller and smaller as the number of iterations increases. Training ends when certain conditions are met, such as the loss value is less than a threshold, or the validation accuracy is above a threshold.

2.2. Deep Learning Framework

There are currently two mainstream deep learning frameworks: one is to construct a static graph before execution to define all operations and network structures, typically TensorFlow, which improves training at the expense of ease of use performance during the period; the other is the immediate execution of dynamic graph calculations, typically represented by PyTorch. By comparison, it can be found that dynamic graphs are more flexible and easier to debug, but at the expense of performance. Therefore, it is difficult for existing deep learning frameworks to meet the requirements of easy development and efficient execution at the same time. This article uses the new deep learning framework MindSpore, which provides a unified coding method for dynamic graphs and static graphs, which greatly increases the compatibility of static graphs and dynamic graphs. Users do not need to develop multiple sets of codes, just change one line of code to switch dynamic graphs. Graph/Static Graph mode. The framework aims to achieve three goals: easy development, efficient execution, and full scene coverage. MindSpore provides users with a Python programming paradigm. With automatic differentiation based on source code transformation, users can use native Python control syntax and other advanced APIs such as Tuple, List, and Lambda expressions. The work in this paper mainly studies the performance comparison between PyTorch and MindSpore.

3. EXPERIMENTAL METHODS

3.1. Workloads Selection

This article tests the main areas of deep learning in order to be as comprehensive as possible. The test work selected two deep learning fields of computer vision and natural language processing in deep learning, and also included the current mainstream neural network architecture.

3.1.1. Computer Vision

Computer vision is a field of artificial intelligence. The image is fed into a neural network, which is trained through a series of mathematical calculations. A trained neural network can classify and detect objects in pictures or videos. In recent years, neural networks have developed rapidly in the field of computer vision. In this paper, we have chosen GoogleNet.

3.1.2. Natural Language Processing

Natural language processing is a field of artificial intelligence that enables computers to read and correctly understand the meaning of human language. The human natural language is input into the neural network, and the neural network is trained through a series of mathematical operations. The trained neural network can understand human language. RNN is a good model for natural language processing. In this paper, we choose the LSTM[9] and BERT[10] models for comparison.

3.2. Unify the Implementation between PyTorch and MindSpore

The implementation of the same neural network between MindSpore and PyTorch may differ in some aspects, which affects training performance and fair comparison. Therefore, we try to unify

the implementations of MindSpore and PyTorch in order to provide a fair comparison. We give implementation methods from two aspects of model structure and hyperparameters. The model settings are shown in Table 1.

Table 1. The settings in MindSpore and PyTorch

Domain	Model	Key Layer	Batch size	Dataset	Framework
CV	GoogleNet	Conv	128	CIFAR-10	MindSpore
CV	GoogleNet	Conv	128	CIFAR-10	PyTorch
NLP	LSTM	LSTM	64	Aclimdb_v1	MindSpore
NLP	LSTM	LSTM	64	Aclimdb_v1	PyTorch
NLP	BERT	Embedding Full-connect	8*256	Cn-wiki-128	MindSpore
NLP	BERT	Embedding Full-connect	8*256	Cn-wiki-128	PyTorch

3.3. Get Accurate Training Speed

Training a neural network can take anywhere from weeks to months. Due to the iterative nature of deep learning training, we only sample a small segment of the entire training, effectively collecting training performance. However, the sampling period may vary from input to input. For models with fixed input lengths, such as CNNs, training can stabilize quickly, which means that the difference between iterations is very small. Therefore, we can collect accurate training scores in a short time. For models with variable input length, such as RNN, the training speed is different for each iteration due to the different input size. In this case, training epochs (traversing the entire dataset) are required for stable performance.

In addition, there is usually a construction phase at the beginning of training to construct the computational graph, allocate memory, and modify some parameters (i.e., the workspace size of different convolutional layers).

Only after this does the computation at each step show repetitive behavior, which can be used to represent precise performance. Next, we describe the method to obtain accurate training speed in these 3 models.

4. EVALUATION

4.1. Experimental Setup

In order to ensure that the hardware is as unified as possible during training, we chose Alibaba Cloud GPU server ecs.gn6e-c12g1.3xlarge, 12-core Intel CPU, 92G memory, NVIDIA v100, and ubuntu18.04 system.

4.2. Overall Training Performance Comparison

In this subsection, we first look into the comparison of overall training performance between MindSpore and PyTorch. The results are shown in Table 2.

Table 2. Overall training speed on MindSpore and PyTorch

Model	Time	Loss	Acc
GoogleNet_MS	126.87(m)	0.0016	93%
GoogleNet_PT	152(m)	0.0016	94.68%
LSTM_MS	1049(s)	0.12	84%
LSTM_PT	1154(s)	0.0057	83.95%
BERT_MS	610(h)	1.7	58.88%
BERT_PT	1147.5(h)	1.71	59.21%

First, the overall training performance of PyTorch and MindSpore under the NVIDIA platform is compared, and the results are shown in the table. It can be seen from the results that the overall performance gap between MindSpore and PyTorch is small. By analyzing the experimental data, it is found that MindSpore's training speed is fast, but its accuracy rate is lower than PyTorch, while PyTorch is just the opposite. PyTorch's training speed is slow, but its accuracy rate is high. In summary, the overall training performance of MindSpore and PyTorch on the NVIDIA platform is very similar.

MindSpore is a deep learning framework developed by HUAWEI. They have developed a matching deep learning computing card Ascend910 for MindSpore. This paper also uses Ascend910 to test the above deep learning model. The experimental data is shown in Table 3.

Table 3. Training speed on Ascend

Model	Time	Loss	Acc
GoogleNet	63.85(m)	0.0016	93.4%
LSTM	523(s)	0.12	85%
BERT	384(h)	1.7	58.90%

Through experiments, we found that the speed of training with Ascend910 is much faster than the speed of training the model with the NVIDIA platform, and the accuracy is similar to the accuracy of the model trained with the NVIDIA platform. To sum up, MindSpore's accuracy rate on Ascend910 is similar to that on NVIDIA platform, but the training speed is faster than NVIDIA platform.

Training performance is an important indicator of deep learning models, and inference performance is also an important indicator in the use of deep learning models, as shown in Table 4.

Table 4. Training speed on Ascend

Framework	Model	Hardware	images/sec
MindSpore	ResNet-50	V100	1490.2
PyTorch	ResNet-50	V100	856.5
MindSpore	ResNet-50	Ascend910	2115

Through the data, we found that during the inference process, the speed of MindSpore is faster than that of PyTorch when using the NVIDIA platform, and the speed of using Ascend910 is much faster than that of using the NVIDIA platform. To sum up, the use of Ascend910 prevails when the application of the model is the primary selection criterion.

5. CONCLUSION

The ultimate goal of this article is to help end users make an informed decision between how to choose two of the most popular deep learning frameworks: MindSpore and PyTorch, in single-GPU training. We systematically evaluate single-GPU training on MindSpore and PyTorch using 3 representative models. Through these comprehensive experiments, we provide insightful observations and recommendations for end users and system developers. First, we decompose the training process of a single GPU, showing that the training process is mainly consumed by GPU processing, which is mainly the execution time of the kernel. Therefore, the running speed of key layers plays a crucial role in single-GPU training. We then evaluate the performance of various models implemented with different key layers and present the trade-offs among them to provide reference for end users to choose various implementations in reality. Finally, we evaluate the performance impact of MindSpore and PyTorch in the dynamic graph case. The conclusion is that when deciding between MindSpore and PyTorch based on training speed, choose MindSpore, and when deciding between MindSpore and PyTorch based on accuracy, choose PyTorch. Choose MindSpore when the application of the model is the primary selection criterion.

ACKNOWLEDGEMENTS

Supported by Beijing City University in 2021 “the innovation and entrepreneurship training program for college students”

REFERENCES

[1] Jouppi N P, Young C, Patil N, et al. In-datacenter performance analysis of a tensor processing unit. In: Proceedings of the 44th Annual International Symposium on Computer Architecture, Toronto, 2017. 1–12

[2] Jia Y, Shelhamer E, Donahue J, et al. Caffe: convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM International Conference on Multimedia, 2014. 675–678

[3] Abadi M, Barham P, Chen J, et al. TensorFlow: a system for large-scale machine learning. In: Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation, 2016. 265–283

[4] Paszke A, Gross S, Chintala S, et al. Automatic differentiation in PyTorch. In: Proceedings of the Autodiff Workshop on NIPS, 2017

[5] Seide F, Agarwal A. CNTK: Microsoft’s open-source deep-learning toolkit. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016. 2135–2135

[6] Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks. In: Proceedings of Advances in Neural Information Processing Systems, 2012. 1097–1105

[7] Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015. 1–9

[8] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016. 770–778

[9] Yu, Yong, et al. "A review of recurrent neural networks: LSTM cells and network architectures." Neural computation 31.7 (2019): 1235-1270.

[10] Devlin J, Chang M W, Lee K, et al. BERT: pre-training of deep bidirectional transformers for language understanding. 2018. ArXiv:1810.04805