Efficient Distributed Image Recognition Algorithm of Deep Learning Framework TensorFlow

Wencai Xu*
Jiangxi Teachers College, Yingtan 335000, Jiangxi, China,
*Corresponding author e-mail: xuwencai@jxtc.edu.cn

Abstract. Deep learning requires training on massive data to get the ability to deal with unfamiliar data in the future, but it is not as easy to get a good model from training on massive data. Because of the requirements of deep learning tasks, a deep learning framework has also emerged. This article mainly studies the efficient distributed image recognition algorithm of the deep learning framework TensorFlow. This paper studies the deep learning framework TensorFlow itself and the related theoretical knowledge of its parallel execution, which lays a theoretical foundation for the design and implementation of the TensorFlow distributed parallel optimization algorithm. This paper designs and implements a more efficient TensorFlow distributed parallel algorithm, and designs and implements different optimization algorithms from TensorFlow data parallelism and model parallelism. Through multiple sets of comparative experiments, this paper verifies the effectiveness of the two optimization algorithms implemented in this paper for improving the speed of TensorFlow distributed parallel iteration. The results of research experiments show that the 12 sets of experiments finally achieved a stable model accuracy rate, and the accuracy rate of each set of experiments is above 97%. It can be seen that the distributed algorithm of using a suitable deep learning framework TensorFlow can be implemented in the goal of effectively reducing model training time without reducing the accuracy of the final model.

Keywords: Deep Learning, Tensorflow Framework, Distributed, Image Recognition

1. Introduction
TensorFlow is widely used in various fields due to its various advantages in the field of deep learning [1-2]. However, TensorFlow is highly restrictive for a single computing node, especially as the size of the data set increases, its limitation becomes more prominent [3-4]. How to remove this restriction, so that TensorFlow can perform efficient expansion on ultra-large-scale systems, increase the training speed of deep learning through parallelism, reduce training time, improve training and testing accuracy, and apply deep learning to more It has great and far-reaching significance in solving complex problems [5-6].

There are many researches on efficient distributed image recognition for deep learning, such as: Samuel J. Yang, Marc Berndl, and others proposed a deep neural network model that can predict the
absolute measure of image focus on a single image without any user-specified parameters; the model runs at the image patch level, and it also outputs a measure of predictive certainty to achieve interpretable predictions [7]. Yue Qi, Wang Jun and others provided new solutions for the analysis and processing of UAV aerial images. They focused on deep learning algorithms, studying the location and fault recognition of insulators in aerial photography, which is important for the automation and detection of UAVs. Intelligence plays an important role [8].

This paper studies the specific implementation of the existing deep learning algorithm framework TensorFlow, the parameter synchronization problem between each operation node in the TensorFlow distributed parallel process, the scheduling problem of data and resources between each operation node in the TensorFlow distributed parallel process, and the design and integration. Optimize the distributed parallel algorithm in the machine learning algorithm framework TensorFlow.

2. Research on Efficient Distributed Image Recognition Algorithm of Deep Learning Framework TensorFlow

2.1 Deep Learning Theory Research

(1) Machine learning

Machine learning was born from the theory of pattern recognition and the theory that computers can perform specific tasks without programming. Machine learning is a data analysis method that can automatically build an analysis model. At the same time, it is also an algorithm that enables software applications to predict results more accurately without clear programming [9-10]. Machine learning algorithms can quickly and automatically generate models even in a very large range, which can analyze large and complex data and quickly provide accurate results. Common algorithm models are: decision tree model, K-means clustering algorithm model, neural network algorithm model, and reinforcement learning model.

(2) Deep learning

The purpose of deep learning is to allow computers to learn from examples such as humans, and hope that machines can acquire skills through experiential learning without human involvement [11-12]. Most deep learning algorithms use neural network architecture, so deep learning models are often called deep neural networks. Deep learning not only requires a large amount of labeled data as a training basis, but also requires huge computing power as a running medium. Although deep learning is a subset of machine learning, it is a relatively unique form of machine learning. Taking image classification as an example, in the process of deep learning operation, manual intervention is not required, and the algorithm automatically extracts features from the image. Compared with traditional algorithms, the advantage of deep learning is that as the amount of data available for training increases, the results of the algorithm can be continuously improved.

(3) Big data and cloud computing

Big data requires the ability to complete the mining of valuable data from the massive data abstracted by end users, and provides a large amount of labeled data for deep learning. Cloud computing implements an on-demand distribution model through scalable and flexible self-service applications, providing powerful computing capabilities for deep learning. The development of big data and cloud computing technology has truly opened the door to deep learning.

2.2 Distributed Deep Learning

(1) Data parallel

Data parallelism refers to the segmentation of training data and the use of multiple instances of the same model to train multiple segments in parallel. Data parallelism is suitable for training medium-sized models with massive training data. The parallel data structure uses the SGD parallel algorithm, and the SGD parallel algorithm is divided into a synchronous mode and an asynchronous mode. In the SGD synchronization mode, all training programs train a large amount of training data at the same time. After the parameter exchange is completed, all training programs start with the new general
model parameters, and then proceed to the next batch of training. In the asynchronous SGD mode, the training program will fill a large amount of training data and immediately change the parameters through the parameter server, regardless of the status of other training programs.

(2) Model parallel

Model parallelism is to divide a single model into several parts in parallel, which are held by multiple training units and complete the training together. For large models that cannot be placed in the memory of a single computer, model parallelism is a good choice. In most cases, the overall communication consumption and synchronization caused by model parallelism exceed data parallelism, so the acceleration is not as good as data parallelism. In addition, in terms of implementation difficulty, defect tolerance and group utilization, data parallelism is better than model parallelism. Therefore, data parallelism is the preferred solution for deep learning spread in practical applications.

2.3 Tensorflow Deep Learning Framework Research

TensorFlow is an open source software library that uses data flow graphs for numerical calculations. Its flexible architecture allows developers to perform calculations on multiple platforms. TensorFlow has the characteristics of high flexibility, strong portability, support for multiple languages, and optimized performance. The entire system architecture of TensorFlow consists of two subsystems, front-end and back-end, with CAPI as the link between them. Its system design has a very good layered architecture. When the system is running, the front-end and the back-end are responsible for different tasks: the front-end system provides programming interfaces for developers using various languages, and is mainly responsible for the construction of the calculation graph; the back-end system provides environmental support for system operation, mainly responsible for the execution of the calculation map.

2.4 Tensorflow Distributed Architecture Analysis

The so-called distributed, the full name is called distributed system, and the popular definition refers to the establishment of a software system on the network. With the aid of the network system, the independent computer hardware is combined into a unified whole and delivered to users. For the user, the interaction between the various hardware in the distributed system is transparent, and the user uses the system as if using a stand-alone system.

(1) Tensorflow remote call

The remote call used in TensorFlow distributed is the gRPC framework, which is essentially an RPC framework, with high performance and universal characteristics. Remote calls are two servers A and B. The application deployed on server A wants to call the function or method provided by the application deployed on server B. However, since the two applications are not in the same memory space, they can only use the existing ones on the network. Protocol (such as TCP, UDP) to complete the representation of the call semantics and the transfer of the call data.

(2) Existing Tensorflow distributed model

For the distributed parallelism of TensorFlow, the two widely recognized parallel modes are data parallelism and model parallelism. Some neural network models are very large and cannot adapt to the memory of a single device. These models need to be stored separately on many devices and trained concurrently. This is the model parallelism in TensorFlow parallelism. In the data parallel framework, each device uses the same model, but different training samples are used for training in each device. Currently widely used and representative models are: parameter server, model partition, Ring Allreduce, Horovod.

2.5 Principles of Convolutional Neural Networks

The principle of recognition and classification based on convolutional neural network is: In the first step, after normalizing the input size of the picture, input it to CNN, and perform convolution operation on the input layer; the second step is to perform sequential convolution operations on the
input as a whole, and then perform non-linear processing on the convolved pictures to increase network sparsity; The third step is to pool, and use the overall statistical characteristics of the adjacent output of a certain position to replace the output of the network at that position, and get a smaller image than the original image, but with more specific feature information; the fourth step is to continue processing such pictures to obtain clear depth information and higher-order feature information, which is convenient for feature comparison; in the fifth step, the entire CNN neural network finally outputs the probability of the category of the input picture through the Sofmax layer, and completes the task of classifying the input picture.

2.6 Challenges Faced by Practical Applications and Solutions
Image recognition based on deep learning faces many challenges in practical applications: translation sensitivity of coherent types, position sensitivity and noise sensitivity, and occlusion problems caused by intensity sensitivity. For the above-mentioned problems that may exist in practice cannot be completely simulated, sample expansion of the data is required. The purpose of sample expansion is mainly twofold: one is to expand the test sample to study the durability algorithm of these problems to amplify the problems of translation sensitivity, position sensitivity, coherent point noise sensitivity and target occlusion sensitivity in practical applications; the other is to compensate the insufficient state in the training set by expanding the training samples, so that the trained algorithm model can also have good distinguishing performance under special conditions.

3. Experimental Research on Efficient Distributed Image Recognition Algorithm of Deep Learning Framework Tensorflow

3.1 Experimental Environment Configuration
(1) Hardware environment
In order to test the previous design and implementation of TensorFlow's efficient distributed image recognition algorithm, when building the TensorFlow cluster, a total of 7 hosts were selected, and all computing devices were under the same local area network. In order to distinguish between the processing capabilities of the equipment in the algorithm, the configuration of the selected hosts is different, so the final cluster can be either a homogeneous cluster or a heterogeneous cluster. The configuration of the selected host is shown in Table 1:

| Serial Number | CPU Core | Memory Size | GPU Type     | IP in LAN           |
|---------------|----------|-------------|--------------|---------------------|
| 1             | 6 Core   | 32G         | GTX1080Ti(11G) | 193.168.152.151     |
| 2             | 6 Core   | 16G         | GTX1060(6G)  | 193.168.152.151     |
| 3             | 4 Core   | 32G         | GTX1080Ti(11G) | 193.168.152.151     |
| 4             | 4 Core   | 16G         | No           | 193.168.152.151     |
| 5             | 4 Core   | 16G         | GTX1080Ti(11G) | 193.168.152.151     |
| 6             | 4 Core   | 16G         | GTX1060(6G)  | 193.168.152.151     |
| 7             | 4 Core   | 16G         | GTX1060(6G)  | 193.168.152.151     |

(2) Software environment
In the experiments involved in this paper, the software configuration of all hosts in the cluster is the same. The operating system uses CentOS 6.4, CUDA uses release 9.0, V9.0.176, TensorFlow uses 1.11.0, C/C++ The development environment uses Visual Studio Code with C/C++ extension, and the python development environment uses Visual Studio Code with python extension. It also installs the Secure Shell (SSH) remote login management protocol, and configures a secret login mode between each host.

(3) Subject
This project is mainly to process image input, so the final model used in the project is a multi-layer
convolutional neural network (CNN). This project takes real-time video of the driver's driving process as input, and classifies whether it is safe or not through a multi-layer deep neural network. Including convolutional layer, pooling layer, fully connected layer, residual network and other deep neural network structures, a total of 32 layers.

(4) Experimental data

The training data is composed of 331 real driving videos provided by a local company. The length of a single video is 30 minutes, the video resolution is 720*480, and the frame rate is 30FPS. In the data preprocessing stage, the aforementioned video is split according to the rule of taking one frame per second, and a total of about 595,800 video pictures are obtained; then, for this batch of original pictures, face detection is performed, and the undetected ones are filtered out Picture of the driver; then, using the knowledge of image coding (formula (1)), the picture is converted from an RGB three-channel image to a single-channel gray image; finally, a total of 42W gray images of the real driving process are obtained.

\[
\text{Gray} = R \cdot 0.299 + G \cdot 0.587 + B \cdot 0.114
\]  

(1)

3.2 Construction of Convolutional Neural Network Based on Tensorflow

(1) Simplified network parameters

In the convolutional neural network, methods such as local connection, weight sharing and pooling are introduced to simplify the network parameters. Taking one-dimensional data as an example, suppose the size of the data is W, the depth of the convolution kernel is F, the step size is S, and the number of zero padding is P. Then for each package core, the input data can be warped, and the feature map can be obtained. Calculated by formula (2), the number of N neurons in the feature map can be obtained. For high-dimensional data, formula (2) can be used to calculate the data of each dimension.

\[
N = \left\lceil \frac{(W - F + 2P)}{S} \right\rceil + 1
\]  

(2)

(2) Determination of network structure and parameters

The convolutional neural network in this article is a multi-scale feature convolutional neural network built on the basis of the classic structure of AlexNet. The training process of multi-scale feature convolutional neural network includes four parts: network parameter initialization, forward propagation calculation, back propagation calculation, weight and bias parameter update. The forward propagation calculation takes the output obtained by the activation function as the input of the next layer and feeds it forward. This article selects the RELU activation function, as shown in formula (3).

\[
f(x) = \max(0, x)
\]  

(3)

4. Experimental Research and Analysis of the Efficient Distributed Image Recognition Algorithm of the Deep Learning Framework Tensorflow

4.1 Performance Evaluation and Analysis of Test Results

In the experiment, the 42W driving pictures in the previous chapter were used to train the driver's unsafe behavior detection model. Different equipment and distributed algorithms were used. Finally, 12 sets of experiments were carried out and each set of experiments was iterating the data by 1. Completed in the case of an Epoch. Table 2 is a comparison of the task completion time and final accuracy of each group:

| Accurac y | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|-----------|---|---|---|---|---|---|---|---|---|----|----|----|
| Accuracy  | 97.1 | 97.5 | 98.4 | 98.2 | 98.6 | 97.5 | 97.9 | 97.2 | 98  | 98.1 | 97.7 | 97.3 |
It can be seen from Figure 1 that the 12 sets of experiments finally achieved a stable model accuracy rate, and the accuracy rate of each set of experiments was above 97%. It can be seen that the high-efficiency distributed image recognition algorithm of the deep learning framework TensorFlow finally achieved the final model training. The results have no effect, but the use of a suitable deep learning framework TensorFlow distributed algorithm can achieve the goal of effectively reducing model training time without reducing the accuracy of the final model.

4.2 Synchronous and Asynchronous SGD Algorithm Comparison Experiment
In the test of training acceleration performance, this paper found that the parallel training of distributed deep learning can greatly improve the training efficiency of the model. This paper designs and implements both synchronous and asynchronous SGD parallel algorithms. In order to prove that the asynchronous SGD parallel algorithm and the synchronous SGD algorithm improve the efficiency of distributed training, this paper conducts the following comparative experiments. The comparison results are shown in Table 3:

Table 3. The Comparison of Asynchronous SGD and Synchronous SGD Training

|        | 0  | 20 | 40 | 60 | 80 | 100 | 120 | 140 | 160 | 180 | 200 |
|--------|----|----|----|----|----|-----|-----|-----|-----|-----|-----|
| asynchronous SGD | 1  | 0.55 | 0.49 | 0.491 | 0.489 | 0.488 | 0.489 | 0.485 | 0.483 | 0.484 | 0.482 |
| Synchronization SGD | 1  | 0.61 | 0.539 | 0.493 | 0.486 | 0.482 | 0.469 | 0.459 | 0.454 | 0.455 | 0.454 |
Figure 2. The Comparison of Asynchronous SGD and Synchronous SGD Training
The faster the top-5 error rate drops, the faster the model is trained. It can be seen from Figure 2 that when the top-5 error rate drops to 55%, the asynchronous SGD parallel algorithm only took about 20 hours, while the synchronous SGD parallel algorithm took about 40 hours. It can be seen that the asynchronous SGD parallel algorithm is significantly faster than the synchronous SGD parallel algorithm, but the top-5 error rate of the classification model trained with the asynchronous SGD parallel algorithm stays at about 48%, which is higher than 45% of the model trained by the synchronous SGD algorithm.

5. Conclusions
This thesis starts from practical applications. Today, with the rapid development of information, big data is produced at a rapid pace. The use of deep models to solve practical problems has become an important part of the entire computer information technology environment. However, the training process of the deep model is often very time-consuming, and the cost increase directly caused by this is huge. This paper aims to solve this problem. Image recognition is a hot topic in Internet search. After the concept of deep learning was put forward, image recognition technology has been greatly improved. At present, the network model used for image recognition has developed from a shallow model to a deep model, which can extract and learn features from images, thereby effectively classifying images.

References
[1] Shujian, ZHANG, Chen, et al. Deep learning network for UAV person re-identification based on residual block. Science China (Information Sciences), 2020, v.63(07):248-250.
[2] Khan M J, Yousaf A, Abbas A, et al. Deep Learning for Automated Forgery Detection in Hyperspectral Document Images. Journal of Electronic Imaging, 2018, 27(PT.2):053001.
[3] Bekeneva Y A, Petukhov V D, Frantsisko O Y. Local image processing in distributed monitoring system. Journal of Physics: Conference Series, 2020, 1679(3):032048 (5pp).
[4] Andrade D, Trabasso L G. An OpenCL framework for high performance extraction of image features. Journal of Parallel & Distributed Computing, 2017, 109(NOV.):75-88.
[5] Zhang S, Ren X, Luo L, et al. Logging-based identification and evaluation of karst fractures in the eastern Right Bank of the Amu Darya River, Turkmenistan. Natural Gas Industry B, 2019, 6(1):58-63.

[6] Halder C, Das N, Roy K, et al. Word-Level Multi-Script Indic Document Image Dataset and Baseline Results on Script Identification. International journal of computer vision and image processing, 2017, 7(2):81-94.

[7] Yang S J, Berndl M, Ando D M, et al. Assessing microscope image focus quality with deep learning. Bmc Bioinformatics, 2018, 19(1):77.

[8] Qi Y, Mu S, Wang J, et al. Intelligent Recognition of Transmission Line Inspection Image Based on Deep Learning. Journal of Physics: Conference Series, 2021, 1757(1):012056 (6pp).

[9] Bai J, Fei J. Research and Implementation of Handwritten Numbers Recognition System Based on Neural Network and Tensorflow Framework. Journal of Physics: Conference Series, 2020, 1576(1):012029 (9pp).

[10] Manaswi N K. Basics of TensorFlow. 2018, 10.1007/978-1-4842-3516-4(Chapter 1):1-30.

[11] Wang L, Cao X, Li M. English Letter Recognition Based on TensorFlow Deep Learning. Journal of Physics: Conference Series, 2020, 1627(1):012012 (6pp).

[12] Qin J, Liang J, Chen T, et al. Simulating and Predicting of Hydrological Time Series Based on TensorFlow Deep Learning. Polish Journal of Environmental Studies, 2019, 28(2):795-802.