A study on communication optimization of distributed gradient descent algorithms based on large-scale machine learning

Hao WU
University of Science and Technology of China, Hefei, Anhui 230026

Abstract: In recent years, the rapid development of new generation information technology has resulted in an unprecedented expansion of information capacity. Machine learning algorithms are also increasingly used to compute information sets and build information systems to solve problems whose complexity makes algorithmic solutions infeasible. Examples include autonomous vehicles, speech recognition or user determination (recommendation systems). The complexity of the machine learning model, combined with the larger amount of data collected, makes it much more expensive to use the model on a single machine, or even impossible to train. Using the computing power of distributed systems is a straightforward, simple solution to the problem. Today, powerful computer clusters are used to train complex deep neural networks on large data sets. However, in large-scale clustered environments, the commonly used distributed synchronous stochastic gradient descent algorithms require frequent node communication to ensure consistency of the gradients (parameters). This has led to the communication bandwidth being a key constraint for distributed machine learning systems.

Keywords: distributed systems; machine learning; distributed optimization; communication efficiency

Introduction:

Gradient compression methods are the most direct way to reduce the communication between nodes, and are mainly divided into gradient quantization and gradient sparsification methods. While quantization methods suffer from limited compression ratios, sparsification methods often lead to serious degradation of model training accuracy. To address these problems, a hybrid gradient compression framework is proposed in this paper after analysing the bottlenecks of the algorithms. The hybrid gradient compression algorithm can combine the advantages of both gradient compression methods, i.e., higher communication compression ratio and lower loss of model training accuracy.

The gradient compression method operates on the full gradient value in each iteration, which introduces additional algorithmic overhead. From our research, we found that the complexity of most of the algorithms is above O(n + klogn). Secondly, although the existing algorithms are able to reduce the communication complexity to a certain extent on the original distributed synchronous stochastic gradient descent algorithm, they are still unable to achieve a constant level of communication overhead of O(1). To address these problems, we analyse the gradient distribution and variation pattern in model training and propose a Gaussian-averaged stochastic gradient descent algorithm. The algorithm can achieve O(n) algorithm complexity and O(1) communication complexity.

I. Research Status

With the rapid development of Internet technology, we have entered a brand new era - the era of big data. According to incomplete statistics from online sources, the total scale of Internet data in China has increased at least 50 times during the decade from 2005 to 2015. The speed of development in these areas is far ahead of Moore's Law for the growth of computing power and Nielsen's Law for the growth of network bandwidth in the field of computer hardware, which we are familiar with. However, due to the increasing volume of information, which is becoming easier to collect, research workers now often need to use collections of millions or even tens of millions of labelled image data to develop image classifiers for image recognition tasks (e.g., the ImageNet dataset, which contains 14 million images with more than 20,000 categories, has become an important dataset in the image domain), and also Thousands of hours of speech data sets are used to train speech recognition models (e.g., Baidu has developed and implemented Deep Speech 2, which uses tens of thousands of hours of audio data and over 2 million sentences of speech to train speech recognition models for English), and tens of millions of games of chess are used to train Go AI programs (e.g., A1, developed and implemented by Google's For example, the A1phaGo AI system developed and implemented by Google's Deep Mind lab used more than 30 million games of Go to learn and train, and was able to beat professional players in the end). The size of this training data was completely unimaginable at the beginning of the 21st century. Learning on such a large dataset requires a lot of computational resources and training time, and therefore places higher demands on the performance of computer hardware and software.

II. Gradient Compression

(i) Quantification

A class of gradient (or parameter) compression methods is quantization, that is, mapping continuous gradient (or parameter) information into a bucket of compressed value sets (usually ranges), the distribution of parameters and gradient values has a narrow dispersion, so this kind of methods can effectively use finite values to characterize the original gradient (or parameter), reducing the number of bits needed to represent each parameter. This approach has been successfully applied to deep learning training, both during training and in inference, where the values passed are quantized for training. Some papers have even quantized the gradient to binary and still achieved convergence, but only with a decrease in the accuracy of the model.

The gradients were compressed using lossless Hoffman coding and good results were obtained in training. In addition, the gradient is represented as 8 bi,. The elements of f lor}t32 are mapped to 8 bits: 1 bit for positive and negative elements, 3 bits for exponential bits and 4 bits for trailing bits. To further reduce the accuracy loss, a mechanism is proposed to dynamically adjust the exponent bits so that the exponent bits can be dynamically changed from 0 to 6 bits. The gradient element of the user-set Min value (Min is set to 0 by default in the text) is represented by "0" and the other elements are represented by "1". In the decompression stage, "0" and "1" are decoded as the average of the local negative gradient elements and non-negative gradient elements respectively. The work also proposes an error accumulation mechanism, where each compression error is added to the next gradient element, in order to reduce the negative impact of the error on the training.

Only the sign of the element is still passed during the transfer process, but it differs when the local node is updated. In addition, a specific error feedback mechanism is proposed to address the problem of non-convergence of the model training in some cases.

(ii) Sparsification

The main idea of the sparsification method is to select a subset of the elements of the original random gradient vector g through a certain strategy, so as to generate a sparse vector. The selection process can be formalized as follows: let b be a mask vector with the same number of elements as g. A "1" bit in the vector indicates that the corresponding gradient element g (i) has been selected. A Hadamard product is performed between the two vectors to generate a sparse vector of the original random gradient. The sparse vector can be represented in the implementation as two vectors: one containing the values of the selected elements of g, and the other containing the index of the corresponding "1" in b. Random-k algorithm. As the name suggests, let d be the number of elements contained in the original gradient vector, and the Random-k algorithm randomly selects a set of no-indexes from the forked set of possible indexes, and sets the no-index bit of b to "1". By its design, the Random-k algorithm is biased, but can be made bias-free by multiplying g by d/i.

In addition, based on the Top-K algorithm, the concept of momentum is introduced into the local gradient transfer process. Warm-up training is also used, i.e. a small compression ratio is used at the beginning of training and then linearly increased to 99.9% to reduce the loss of convergence accuracy. A momentum term is also introduced to correct for local gradients.

To address the difficulty of determining the appropriate compression ratio in practice for Top-K class algorithms, an unbiased sparse coding algorithm is proposed to maximise the sparsity and control the variance of the gradient to ensure convergence speed, after finding that the variance of the gradient affects the convergence speed. At the same time, Adacomp can automatically modify the compression rate according to the local gradient activity, thus achieving a compression ratio of 200 times for the fully connected layer and 40 times for the convolutional layer with a certain Top minus one accuracy for deep neural networks and ImageNet training.

III. Gaussian average-based distributed stochastic gradient descent algorithm

(i) Design ideas and details analysis of Gaussian-averaged stochastic gradient descent algorithm

Since all working nodes are required to exchange their locally computed gradient information, the communication required in the gradient synchronization phase poses a scalability challenge for data-parallel distributed stochastic gradient descent. Although gradient compression methods (sparsity as well as quantization methods) can reduce the communication complexity of gradient synchronization, the computational efficiency of gradient compression is still critical for scalability of gradient synchronization in the case of small-scale node parallelism. Although the Top-K algorithm can reduce the amount of communication per staff, its computational overhead can offset the reduction in communication, resulting in longer execution times per iteration. As we have observed in experimental evaluations of distributed systems with 100 Gbps bandwidth, the high computational cost of the Top-K algorithm can overshadow its advantages in terms of communication efficiency. The same problem occurs for quantization algorithms. The Top-K algorithm is designed to avoid computationally costly ranking as well as selection processes. It assumes that the distribution of the gradients computed in a single iteration is Gaussian and estimates the threshold value based on statistics to select the appropriate gradient value, eliminating the sorting step in the original method implementation and reducing the computational complexity to O(n). Recent work has also demonstrated the importance of low computational complexity in the scalability of compression algorithms. All of the above work reduces the cost of gradient synchronisation and demonstrates the convergence capability of the model. However, they all require the working nodes to exchange gradient information of a certain size.

Based on the existing research work, we propose a novel approach to optimise the communication process. We assume that the gradients computed in each iteration obey a Gaussian distribution, and we can estimate the expectation of the distribution as well as the standard deviation based on statistical probability at the local computation node. Instead of averaging the gradient values themselves, we choose to exchange all local standard deviations between all working nodes and obtain the average of these standard deviations. The global average of the standard deviations also allows for gradient synchronisation at each worker node to allow for global sharing of updates across all operational nodes. This effectively reduces the communication cost per iteration to a floating point value (32 bits), thus enabling communication complexity for the local working nodes can. To further preserve information, we also utilised the mean of the locally computed gradient vectors and used them in the decoding process to ensure the smallest possible loss of information.

(ii) Study of the gradient distribution during model training

In this paper, we have discussed the theoretical perspective to study the law of gradient distribution in the training process of deep learning models. Based on most of the experimental results and previous work, it is assumed that the gradient values obey Gaussian distribution during the training iterations, and a sparsification method is proposed based on this assumption. In order to further verify the law of gradient distribution, here, we train three common deep network models in the domain on the dataset and extract the computed gradient

values from them every 300 iterations and analyse them. At each iteration of the update, the majority of the gradient values were in the. The gradient values are similar to a Gaussian distribution in the vicinity of the values.

Furthermore, as the number of iterations increases, more of the gradient values move closer to the central value, i.e. the variance of the entire gradient decreases. This explains why the gradient values continue to decrease as the model is updated in successive iterations. In the field of theoretical analysis, the variance of the gradient value is often used as a criterion for reaching an optimal value. In our algorithm design, we also assume that the gradients calculated during the training of the network model follow a Gaussian distribution.

Conclusion:

In this paper, we have focused on communication efficient distributed machine learning optimization algorithms to further improve the training efficiency of distributed systems for machine learning and to increase the utilization of computing power. First, gradient compression is the most direct way to reduce the communication between nodes. To address the shortcomings of the current algorithms, this paper proposes a hybrid gradient compression framework after analyzing the bottlenecks of the algorithms, and implements and verifies the algorithms under this framework. The results show that the hybrid gradient compression algorithm can combine the advantages of both gradient compression methods, i.e., higher communication compression ratio and lower loss of model training accuracy. However, gradient compression is essentially a low precision representation of the elements, and most of the existing gradient compression algorithms do not take into account the machine learning training task in their computation process.

References:

[1] Zhang Yazhong, Li Yuxiao, Liu Yuxiang, He Wei. OAM alignment algorithm based on machine learning gradient descent method [J]. Communication Technology, 2019, 052(006):1316-1319.

[2] Zhang YZ, Li YX, Liu YX, et al. OAM alignment algorithm based on machine learning gradient descent method[J]. Communication Technology, 2019, 52(6):4.

[3] Xie Zaipeng, Li Bowen, Zhang Ji, et al. A distributed coding-based stochastic gradient descent optimization method:, CN111104215A [P]. 2020.

About the author:

Hao Wu (1999.10.8 -), M, Han, Ningbo, Zhejiang, China, M.S., Research interests: Distributed machine learning, machine learning systems