Overview of Dr Nghiem's research

Machine learning problems are usually converted into an objective function to solve. The optimization algorithm is an important tool for solving parameters in the objective function. At present, the optimization algorithms commonly used in machine learning mainly include gradient decent (GD) algorithm, second-order algorithm, proximity gradient (PG) algorithm, coordinate reduction (CD) algorithm and alternating direction multiplier method (ADMM). However, faced with the challenges of big data, the single-thread optimization algorithm cannot meet the needs of large-scale machine learning applications. Parallel and distributed optimization algorithms based on different computing models have become research hotspots. In the multi-core environment, researchers have proposed parallel optimization algorithm based on sharing storage which updates the model parameters in the shared memory by multi-thread parallel training model parameters using dividing the sample data. Consequently, it requires a collaborative framework involving a group of agents coordinated by a central coordinator. The whole problem of optimization is divided into several sub-problems for the agents to solve. In the process of execution, the agents submit their optimization solutions of these sub-problems back to the central coordinator and the later one collect all the information together and figure out next optimization direction towards the global target. Then the coordinator broadcast different subtasks for the agents to optimize in the new direction. Consequently, there are a lot of communications between the agents and the

central coordinator. It will get even worse when the number of agents becomes larger or the communication cost is unaffordable. One possible solution for this extensive communication cost in distributed optimization environment is to predict the results of some agents' optimization, which reduces the unnecessary communication steps.

Dr Nghiem's research focus on learning agents' operations with Gaussian Processes, which is an adaptive mechanism. Firstly, he developed a model to predict the optimized gradient information from the historical data based on Gaussian Processes. According to the predictive variance, the coordinator decides whether to communicate with the agent or not. Furthermore, this predictive model was developed based on linearized Gaussian Process to deal with non-convex optimization problems which is rather typical. The efficiency and advantages of both predictive model are demonstrated using critical numerical examples.

Further questions

This model involving a central coordinator responsible for provisioning and storage and several agents responsible for calculations and updates is the current mainstream model. It is called the master-slave or server-client model. Such models usually exist in a fixed, star-shaped structure, most notably the parameter server. There are a few problems, such as decentralized optimization of the initial motivation: what happens when the server goes wrong? This year, AWS has had several network outages, causing multiple websites such as Piazza and Quora. If

the same thing happened to the DNN model you ran for half a month, how can the results be guaranteed even if the server is down, or even let the node continue to calculate. One direct idea is to add backups or to set up multiple servers to form a hierarchical network structure.

In recent years there is a new type of idea: simply remove the server to optimize the side communication between nodes, and finally achieve the effect of entire prosperity and intelligence of the whole network. There are quite a few implementations of this type of algorithm. For example, EXTRA [1] which is a consensus algorithm, DIGing[2], diffusion of diffusion algorithms [3], and some theoretical work [4]. Such algorithms have both pros and cons.

First, without a server, the system is more robust ensure the privacy of the data on larger issues. For example, we are considering allowing all hospitals to connect through such a network. Without disclosing patient information based on a central server, they can be consistent and equivalent for models to train. It will adapt to more local data. In the same cases, if you use a server with unified upload and unified distribution, everyone's model will stay the same even if new data comes. The hospital generally prefers that its model adapts to the local patient situation. For this case, the decentralization algorithm is used. Only the information of the neighbors within 50 kilometers can be collected and the information farther away is not so careful. This information can still pass by through neighbors step by step. Last but not least, this algorithm is very interesting. In reality, animals like ants, sardines and bees all gain community wisdom by communicating with neighbors

rather than establishing servers. Its disadvantages are also obvious. The total traffic of all nodes is even larger than the central algorithm. If the central algorithm has n connections in the case of n nodes, the decentralization algorithm will have nE, where E is the average degree of the nodes. Because the overall convergence speed is slower than the central algorithm, this traffic is even larger. On the other hand, there is no hardware support as well.

Reference

[1] W. Shi, Q. Ling, G. Wu, and W. Yin. Extra: An exact first-order algorithm for decentralized consensus optimization

[2] A. Nedich, A. Olshevsky, and W. Shi. Achieving geometric convergence for distributed optimization over time-varying graphs

[3] J. Chen and A. H. Sayed. Diffusion adaptation strategies for distributed optimization and learning over networks

[4] K. Yuan, Q. Ling, and W. Yin. On the convergence of decentralized gradient descent