1. Project Description

1.1 Introduction

Deep learning has shown great power in many practical applications, ranging from speech recognition, visual object recognition, to text processing.

The recent observation has shown that the deep learning system must be able to handle very large number of training instances and model parameters for better accuracy. This trend builds a high barrier preventing more efforts to this area: researchers not only need to be expertise in learning algorithms, but also on implementing distributed system infrastructure.

In this work, we are building a deep learning test bed to enable DL researchers to run their algorithms in a distributed manner, without needs to implement distributed system stuffs. Researchers only need to describe the logic of their algorithms and our system auto-parallelizes the algorithm to a distributed environment and provide runtime environment with fault-tolerance, data synchronization, etc..

1.2 Research Goals

Our research goals contain two major parts:

1. A distributed system handling low-level distributed computation primitives, like concurrency, synchronization, and fault-tolerance, etc.
2. User-friendly model description language enabling researchers define their algorithms efficiently.

1.2.1 Distributed system infrastructure

Our work in this subject lies on two layers, distributed data store/management layer and computing layer.

(i) Distributed data store/management layer

In machine learning algorithms, the data should be stored in not only the distributed file systems, but also the memory of machines to avoid duplicate IOs on those intermediate results. We are going to develop multiple layer data management strategies, so that it will adjust the location of data (in disk or memory) according to the logic of computation.

(ii) Computing layer

The key component in computing layer is the scheduler of distributed tasks. In deep learning system, we have multiple design considerations for task scheduling:

1. Task scheduling within the same layer. Tasks within the same layer are vulnerable to problems like long tail, uneven load balancing in turn to blocks the execution of next layer. We are going to find the optimal scheduling for tasks in the same layer.
2. Scheduling across layers. The workload across multiple layers can be executed with pipeline for better performance. The challenge in this design is to arranging the tasks for more chances to pipeline and to keep the correctness of computation.

1.2.2 Model Description language

Recalling the goal of this project, low-level system details should be transparent to users. Users expect to use high-level description language to define the expected computational logic.

Our work in this part contains three components:

1. We will implement those common computing process in learning algorithms in distributed style and provide them to user as basic primitives. In this way, users can define the computation logic efficiently.
2. We will provide a high-level description language which encapsulates the details of languages like C++, Java to enable users focus on the logic instead of programming details.
3. We also provide auto-parallelization engine which takes the result in b as input, translate them into distributed tasks and run them on the system we mentioned in 1.2.1.

2. AWS Usage Plan

We plan to run our Deep Learning testbed on Amazon EC2. We will take EC2 as the development, test and experimental environment for our system. We request two-year free access of Amazon EC2 to cover the expenses for our system implementation and technique validation. We will publish our results on top conferences and journals and further apply external research funding.