ECE 428 P4 Report
Group 25
NetIDs: tvitkin2, zhuxuan2
Names: Timothy Vitkin, Zhuxuan Liu

Design of system

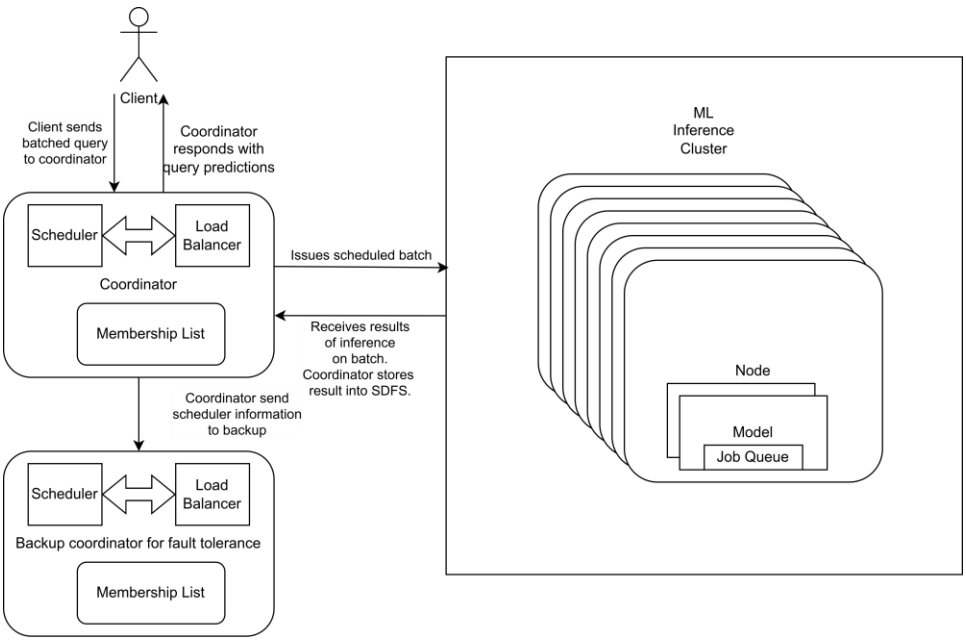
The system builds upon the membership protocol of MP2 and the SDFS of MP3. The system supports distribution of inference work across a cluster of VMs.

The client sends a message to do work to the coordinator. The coordinator enqueues the request. It then handles the message by dequeuing it when available. The coordinator will distribute work to all the nodes to execute the instructions. When sending out work, it will send a batch of a specified size. The batch size can be different across different neural nets, but it should not change over time for the same neural net. When nodes are done with a batch, they write their inference results to SDFS. After this write succeeds, they inform the coordinator that the batch is done, and the coordinator schedules the next batch. Each node will only have 1 batch being inferred rather than splitting across multiple batches.

The system ensures fair time inference by checking the query rate for each job. The job with the lowest query rate is prioritized for scheduling.

The system is designed to handle faults. If a non-coordinator node fails, its work will fail to commit to SDFS. The coordinator keeps track of all scheduled batches on a node and will reschedule batches that were on the failed node. The coordinator backs up the jobs and complete batches in each job using SDFS. This ensures that there will still be progress upon coordinator failure. When the coordinator is restored, it will refer to the newest backup of job progress and resume from there. There could be duplicate work, but there will not be any lost batches. When jobs are scheduled to a node, they will fetch the dataset to infer from SDFS. Even with leader failure, the dataset to perform scheduled inferences is available, ensuring inference can continue. There is also a write buffer if SDFS goes down.

The design is implemented in Python.



ECE 428 P4 Report
Group 25
NetIDs: tvitkin2, zhuxuan2
Names: Timothy Vitkin, Zhuxuan Liu

ECE 428 P4 Report
Group 25
NetIDs: tvitkin2, zhuxuan2
Names: Timothy Vitkin, Zhuxuan
Liu

Data analysis

1. Fair time inference

a) Ratio of batches to meet fair time inference with 2 jobs

The scheduler picks the “best” model, as in, the model with the lowest query count within the last 10 seconds. We found that ResNet jobs take 7 VMs while AlexNet jobs take 3VMs. This is because ResNet is a heavier model than AlexNet. This results in ResNet requiring more VMs. For this example, we had the same batch size for each. However, unless the batch size affects the query rate calculation (for example the batch takes more than 10s to infer), it will not change the ratio of VMs for fair time inference.

b) Time taken for the second job to begin executing in the cluster

A realistic estimate is to take into consideration how long it takes to load up all 8 inference nodes. Since each batch job takes approximately 1 second to execute, there will be up to 1 second prior to the second job beginning to do work. Based on our observations, it tends to be around 0.2 seconds because the nodes do not complete a batch simultaneously so there will quickly be a node available.

2. Failure of 1 non-coordinator VM

Time taken to go back to normal: An average case to detect a node failure is 2.5 seconds (worst-case = 5 seconds) in our system. Then, to acquiesce to “normal” differences in fair time, the system requires two intervals of the scheduler (to schedule two new batches on a different model) to complete, so an additional 2*0.5=1 second is required.

3. Failure of coordinator VM:

Time taken to go back to normal: This task is network-bound and is determined by the time that it takes for the backup coordinator to setup its filesystem representation of the network. Over the course of 5 runs on localhost, we found that running an LS on 10 VM’s with the entirety of the Oxford Pets Dataset loaded into SDFS took 0.217 seconds, with a max at 0.253 seconds. Coupled with the time that it takes for the failure detection to work, which is upper-bounded at 5 seconds, we are looking at an experimental upper bound of 5.26 seconds.