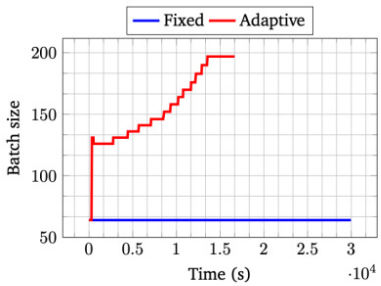
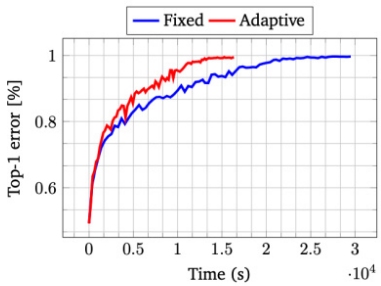
Hessisches Kompetenzzentrum
für Hochleistungsrechnen

Designing an Efficient Neural Architecture Search



(a) Batch size



(b) Accuracy

Project Manager
Arya Mazaheri

Researchers
Tim Beringer

Principal Investigator
Prof. Dr. Felix Wolf

Project Term
2020 - 2021

Clusters
Lichtenberg Cluster Darmstadt

Software
PyTorch

Institute
Parallel Programming

University
Technische Universität Darmstadt

Introduction

Within the last decade, deep neural networks have attracted much attention from academia and industry. Such a wide adoption is primarily due to recent advancements in big data, big compute, and efficient algorithms. Deep convolutional neural network (ConvNet) is a wellknown technique that has been successfully applied to many computer-vision tasks, such as object detection, semantic segmentation, pose estimation, and video classification. Such networks are first trained on a preferably large dataset to obtain a ConvNet model.
The model can then be deployed on the target hardware to perform inference. Both phases require extensive optimizations to become suitable for industrial applications.
Training ConvNets are often highly parallelizable and scalable. They heavily use computational resources, ranging from a high-end GPU to a large cluster of machines. Although such resources are readily available to the industry and product developers, not enough attention is being paid to the orchestration of AI jobs. Hence, AI projects have become the source of bottleneck in research and development. As a remedy, an AI-tailored job scheduler can immensely enhance the scalability of the deep-learning training jobs and give a more accurate completion time estimate. Furthermore, such a specialized job scheduler can free the user from defining scheduling parameters, resulting a higher resource utilization.

Methods

We introduce an elastic multi-tenant DL-aware scheduling platform based on the all-reduce architecture to deal with the increasing cost and complexity of DL training and neural architecture design. The multi- tenancy feature of our scheduler enables the efficient training of several deep-learning jobs concurrently. Furthermore, our proposed scheduler is equipped with a progress monitoring component for retrieving status updates, a cluster of docker containers orchestrated by Kubernetes to perform the actual training work, and a commandline client through which a user can interact with the

system. The user is only required to submit the deep-learning training script. Thus, no scheduling script (e.g., SLURM job script) is required. Our scheduler operates in two separate but interdependent levels to provide scalable and elastic job scheduling, namely cluster-level and job-level scheduling. The former deals with efficient resource allocation given the available nodes on the cluster, while the latter attempts to adjust the training hyper-parameters of the individual training jobs to enhance their throughput and efficiency. Altogether, we can decrease the training time and increase the overall resource utilization.

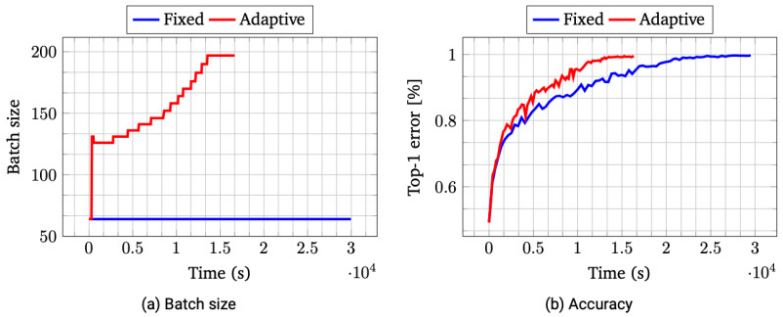
Results

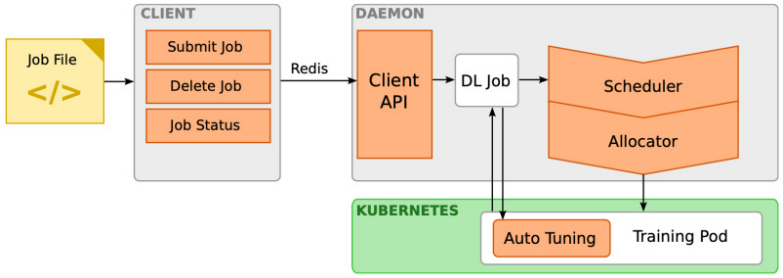
Unfortunately, we couldn’t install docker and Kubernetes on the Lichtenberg cluster for obvious reasons. Thus, our subproject related to automatic resource scaling has been put on hold. However, we obtained promising results related to hyperparameter tuning. Results show that adaptive parameter tuning can accelerate NAS jobs on the cluster and obtain a similar accuracy.

Discussion

Given the initial promising results that we obtained on 4 GPUs on Lichtenberg cluster, we are eager to continue our experiments with larger number of GPUs. If resources become available, we aim to use 16 GPUs with 4 nodes to validate our method. We are also considering switching to Podman to achieve the job scalability idea that we had envisioned for this project.

Figures





Last Update: 2022-04-27 15:53