공유 GPU 클러스터에서의 딥러닝 작업 자원 경합 민감도 분석

유준열°, 전병곤

서울대학교 컴퓨터공학부
{gajagajago, bgchun}@snu.ac.kr

Investigating Contention Sensitivity of DL Training Workloads in Shared GPU Cluster

Junyeol Ryu°, Byung-Gon Chun
Department of Computer Science and Engineering, Seoul National University

요 약

Diverse deep learning(DL) jobs differ in resource usage characteristics due to distinct data, model structure, and training mechanism. Packing is a common technique used by GPU cluster schedulers to colocate multiple jobs to share GPUs, and is believed to improve cluster utilization as well as system throughput. However, a naive GPU packing decision blindly targeting higher utilization may incur contention, leading to an unintentional performance degradation. Thus, fine-grained profiling is critical to efficient packing. In this paper, we investigate the resource usage characteristics of DL jobs through fine-grained profiling and analyze contention sensitivity of each job.

1. Introduction

Deep learning (DL) has been adopted as breakthrough technology in almost every intelligent domain. We have witnessed the success of DL in computer vision [5, 6], language model [1], recommendation [4, 12], reinforcement learning [10], and speech recognition [2]. The wide spectrum of fields DL has been applied to introduced the development of diverse DNN model architecture, training data, and training algorithms. Due to this structural diversity, training of different DL jobs show highly divergent resource demands. Existing works have found DL jobs can be sensitive to the amount of GPU, as well as other auxiliary resources such as CPU and memory [11, 16].

Alongside the recent algorithmic innovations, it has been discovered that the performance of DL training can be improved with larger model parameters and training data. The growing scale of DL makes its training to be a long running task that needs multiple compute resources, mainly GPUs, to be gang-scheduled. It is now a commonplace for enterprises to build shared GPU clusters in order to accelerate DL training at scale [7, 16].

Today's shared GPU cluster for DL consists of multiple interconnected GPUs, i.e., nodes, and the nodes are connected to each other via high bandwidth Ethernet [14]. Cluster resources are shared among simultaneously running jobs. However, the heterogeneity of DL jobs' resource usage characteristics is becoming a main challenge to the shared cluster. A naive co-allocation of jobs with similar resource demands may incur contention, slowing down colocated jobs and damaging the system throughput.

Thus, a shared GPU cluster requires cluster scheduler make dynamic scheduling decisions to avoid contention while improving cluster utilization. The efficient allocation of resources critically depends on mitigating resource contention among resource sharing jobs. It is important to analyze the contention sensitivity of DL jobs and design scheduling algorithms to efficiently share the limited resources of the cluster. Exploiting the resource heterogeneity in resource usage of DL jobs can open a new opportunity to efficiently share the GPU cluster without compromising job level metrics.

2. Background

2.1 Shared GPU Clusters

DL training is a long-running and resource intensive task [11]. The fast growing scale of both data and the DNN model, and the demand for more resources has encouraged DL training on shared GPU clusters. GPU clusters employ scheduler [9, 17] to optimize job-level and system-level metrics. Job-level metrics include job throughput and job completion time. System-level metrics include system throughput and cluster utilization. To achieve successful training in both metrics, efficient allocation of cluster resources by the scheduler is critical.

2.2 Contention Sensitivity

DL jobs are heterogeneous in resource demands. Image [5, 6] and language models [1] are known to be GPU-intensive, recommendation [4, 12], GNN, reinforcement learning [10] to be CPU-intensive [16]. If a job shows an intense demand for a certain resource, we can deduce that the contention on the resource will be severely detrimental to job throughput. Therefore, we redefine the resource usage characteristics of a job as its contention sensitivity.

2.3 GPU Packing

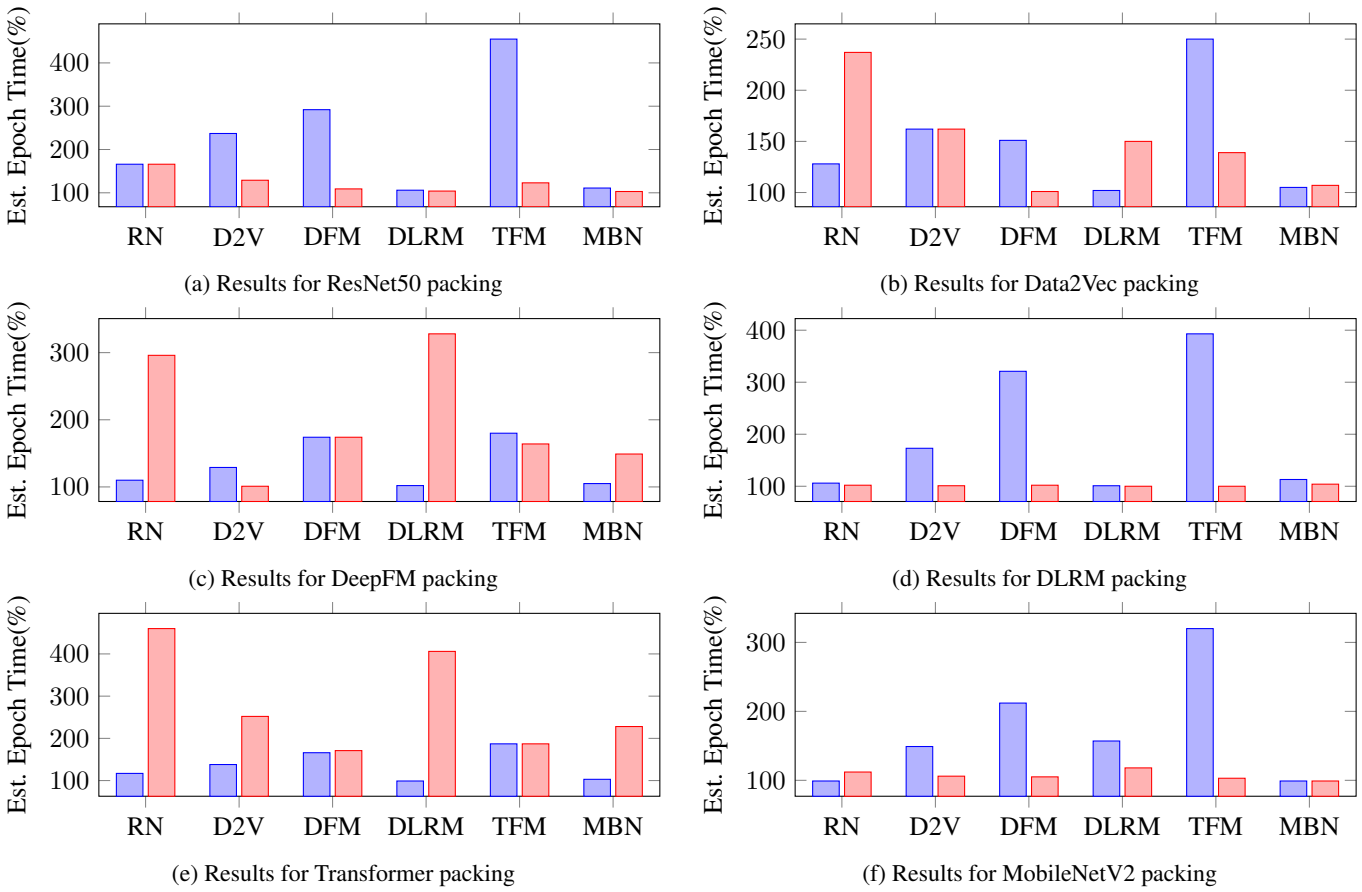
GPU packing defines spatial sharing of GPU, i.e., more than two jobs running simultaneously on a single GPU. On top of current NVIDIA GPUs' support of parallel execution of multiple CUDA kernels, techniques such as MIG or MPS are actively used. With MPS, processes share a shared context to mitigate the context switch overhead. MIG partitions GPU into small isolated slices to provide hardware isolation. MIG benefits processes with error isolation, typically from out-of-memory error, but has limitations in inter-GPU peer-to-peer communication. With memory size constantly growing(e.g., NVIDIA A100 80GB GPU), GPU packing is being actively utilized by recent cluster schedulers [8, 15, 18].

3. System Overview

The primary objective of the cluster scheduler is "sharing well", i.e., improving cluster utilization while minimizing contention overhead. This can be achieved in conjugation of 1) efficient packing, i.e., configuring GPU sharing sets by pairing jobs that do not contend the same intra-GPU metric and 2) efficient collocation, i.e., co-allocating the jobs that do not contend the same host metric(CPU, memory) to the same node. Thus, the scheduler should acknowledge the contention sensitivity of jobs and operate by a scheduling policy that can achieve the "sharing well" objective.

Model	Est. Epoch Time	Host metrics		Device metrics				
		CPU	Memory	SM	Memory	L2	Bdw(read)	Bdw(write)
ResNet50	830	9%	26%	54%	46%	27%	27%	18%
Data2Vec	2484	10%	10%	29%	28%	20%	15%	12%
DeepFM	3505	9%	60%	5%	85%	27%	47%	38%
DLRM	2696	10%	92%	10%	4%	4%	4%	0%
Transformer	3497	8%	9%	36%	69%	37%	48%	20%
MobileNetV2	605	11%	42%	36%	63%	26%	32%	31%

Table 1: Profiled resource usage of diverse DL models



(a) Results for ResNet50 packing

(b) Results for Data2Vec packing

(c) Results for DeepFM packing

(d) Results for DLRM packing

(e) Results for Transformer packing

(f) Results for MobileNetV2 packing

Figure 1: Results for packing. RN, D2V, DFM, DLRM, TFM, MBN stand for ResNet50, Data2Vec, DeepFM, DLRM, Transformer, MobileNetV2. Y-axis indicates the ratio of the estimated epoch time when packed compared to exclusive execution. Left bar(blue) is the value of the base model and the right bar(red) is the value of the model at the corresponding x coordinate.

3.1 Profiling

We implement a fine-grained kernel profiler using Nsight Compute [13]. Existing schedulers[17] typically rely on coarse-grained profiling which only tracks job throughput or high-level metrics such as GPU/CPU utilization, GPU/CPU memory, etc. However, fine-grained GPU metrics(SM, L2 cache, memory bandwidth utilization) may be the most relevant factors correlated to the contention[3]. Thus our profiler collects fine-grained GPU metrics along with the metrics of other coarse-grained profilers.

4. Evaluation

In this section, we profile contention sensitivity of various DL jobs and their packing performance. The estimated epoch time is calculated by multiplying batch size with the profiled job throughput.

4.1 Experiment Setup

Resources. We run experiments using a server with six NVIDIA TITAN RTX 24GB GPUs, 72-core Intel Xeon Gold 6254 CPU, and 252GB RAM. Each packing pair is launched inside an isolated Docker container of a single GPU, with 10 CPU cores and 40GB RAM allocated proportionately to resemble the server configuration described above. Packing inside each container is enabled with MPS.

Models. Our experiments consider six different DNNs as shown in Table 1. We pick the models from four different workloads. ResNet50 [5] and MobileNetV2 [6] are image tasks, Transformer [1] is a language model, DLRM [12] and DeepFM [4] are recommendation models, and Data2Vec [2] is a speech model.

4.2 Results

Table 1 depicts profiled resource usage of diverse DL models. Image tasks show highest SM utilization among the workloads. Language model workloads show high demand for GPU memory and GPU memory read bandwidth. Recommendation mod-

els show highest host memory intensity. Speech model workloads show moderate intensity for all metrics.

Figure 1 depicts the estimated epoch time of each DL job when packed with another job. The result indicates that an efficient packing pair has a complementary resource usage profile. For example, packing Data2Vec and MobileNetV2 did not cause contention, as the estimated epoch time of each job remained almost equivalent to an exclusive setting. The result also shows that an inefficient pair has a conflicting profile, leading to bottleneck overlap and resource contention. For example, packing DLRM and DeepFM from the same recommendation workload incurred the estimated epoch time of DLRM to more than triple. This advocates that finding optimal packing plans can improve cluster utilization while preventing contention, thereby achieving "sharing well".

5. Conclusion

We implement a fine-grained kernel profiler and profile contention sensitivity of jobs from diverse workloads. Experiments show that our approach can promote "sharing well" of the shared GPU cluster and improve cluster efficiency and job level metrics in conjugation. We hope our work will benefit existing DL clusters.

6. Acknowledgments

This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2015-0-00221, 다양한 분석을 고속 수행하는 단일화된 빅데이터 스택 개발).

References

[1] A. Baevski and M. Auli. Adaptive input representations for neural language modeling. *arXiv preprint arXiv:1809.10853*, 2018.

[2] A. Baevski, W.-N. Hsu, Q. Xu, A. Babu, J. Gu, and M. Auli. Data2vec: A general framework for self-supervised learning in speech, vision and language. *arXiv preprint arXiv:2202.03555*, 2022.

[3] S. Choi, S. Lee, Y. Kim, J. Park, Y. Kwon, and J. Huh. Serving heterogeneous machine learning models on {Multi-GPU} servers with {Spatio-Temporal} sharing. In *2022 USENIX Annual Technical Conference (USENIX ATC 22)*, pages 199–216, 2022.

[4] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He. Deepfm: a factorization-machine based neural network for ctr prediction. *arXiv preprint arXiv:1703.04247*, 2017.

[5] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[6] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019.

[7] M. Jeon, S. Venkataraman, A. Phanishayee, J. Qian, W. Xiao, and F. Yang. Analysis of large-scale multi-tenant gpu clusters for dnn training workloads. In *2019 USENIX Annual Technical Conference (USENIX ATC 19)*, pages 947–960, 2019.

[8] G. Lim, J. Ahn, W. Xiao, Y. Kwon, and M. Jeon. Zico: Efficient {GPU} memory sharing for concurrent {DNN} training. In *2021 USENIX Annual Technical Conference (USENIX ATC 21)*, pages 161–175, 2021.

[9] K. Mahajan, A. Balasubramanian, A. Singhvi, S. Venkataraman, A. Akella, A. Phanishayee, and S. Chawla. Themis: Fair and efficient {GPU} cluster scheduling. In *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)*, pages 289–304, 2020.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[11] J. Mohan, A. Phanishayee, J. Kulkarni, and V. Chidambaram. Looking beyond {GPUs} for {DNN} scheduling on {Multi-Tenant} clusters. In *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*, pages 579–596, 2022.

[12] M. Naumov, D. Mudigere, H.-J. M. Shi, J. Huang, N. Sundaraman, J. Park, X. Wang, U. Gupta, C.-J. Wu, A. G. Azzolini, et al. Deep learning recommendation model for personalization and recommendation systems. *arXiv preprint arXiv:1906.00091*, 2019.

[13] NVIDIA. Nsight compute. `https://docs.nvidia.com/nsight-compute/NsightCompute/index.html`.

[14] S. Shi, X. Zhou, S. Song, X. Wang, Z. Zhu, X. Huang, X. Jiang, F. Zhou, Z. Guo, L. Xie, et al. Towards scalable distributed training of deep learning on public cloud clusters. *Proceedings of Machine Learning and Systems*, 3:401–412, 2021.

[15] G. Wang, K. Wang, K. Jiang, X. Li, and I. Stoica. Wavelet: Efficient dnn training with tick-tock scheduling. *Proceedings of Machine Learning and Systems*, 3:696–710, 2021.

[16] Q. Weng, W. Xiao, Y. Yu, W. Wang, C. Wang, J. He, Y. Li, L. Zhang, W. Lin, and Y. Ding. {MLaaS} in the wild: Workload analysis and scheduling in {Large-Scale} heterogeneous {GPU} clusters. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*, pages 945–960, 2022.

[17] W. Xiao, R. Bhardwaj, R. Ramjee, M. Sivathanu, N. Kwatra, Z. Han, P. Patel, X. Peng, H. Zhao, Q. Zhang, et al. Gandiva: Introspective cluster scheduling for deep learning. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 595–610, 2018.

[18] P. Yu and M. Chowdhury. Salus: Fine-grained gpu sharing primitives for deep learning applications. *arXiv preprint arXiv:1902.04610*, 2019.