Appendix: Artifact Description/Artifact Evaluation

ARTIFACT DOI

10.5281/zenodo.8197186

ARTIFACT IDENTIFICATION

We propose FamilySeer, an auto-tuning framework for accelerating the deep learning inference, which leverages a new search method exploiting the subgraph similarity. FamilySeer focuses on improving the search efficiency (i.e., the tuning time to reach the given inference latency) and search quality (i.e., the final inference latency after tuning). And the artifact also focuses on these two parts.

We evaluate FamilySeer on eight representative deep learning models of Table 3 on two platforms. The GPU platform is equipped with two Nvidia V100 GPUs (CUDA 11.2), and the CPU platform is equipped with two Intel Xeon Silver 4210 CPUs. During experiments, we control the stochasticity of FamilySeer as follows: 1) the seeds of random number generators (for TVM and XGBoost) are fixed, therefore the family cost models can be initialized identically across multiple runs, and 2) the hyper-threading and the turbo boost on CPU are disabled with CPU frequency set at maximum, and the CPU power policy is set to performance, to avoid performance fluctuation on both platforms.

The computational artifacts can fully reproduce all unique contributions claimed in this paper.

REPRODUCIBILITY OF EXPERIMENTS

The following experiments take about three weeks totally. Therefore, we have also provided our logs to save the AE reviewers' time. With these logs, the repeoducibility time can be reduced to five hours.

Search efficiency. We compare the search efficiency of FamilySeer with that of AutoTVM and Ansor. We let Ansor run the search with its recommended time budgets, i.e., 900 for each subgraph on GPU and 800 on CPU and calculate the maximum decreased latency. Then we perform the auto-tuning using FamilySeer, Ansor, and AutoTVM, and report the time to get the tuned model codes with 80%, 90%, and 100% of the maximum decreased latency. The results indicate that FamilySeer costs less tuning time than Ansor and AutoTVM. Especially on GPU platform, the multi-GPU acceleration optimization can further reduce the tuning time.

Search quality. We compare the search efficiency of FamilySeer with that of Ansor, AutoTVM, and XLA. We give Ansor and AutoTVM sufficient time budgets as they recommend, and FamilySeer uses the same time as Ansor consumed. And XLA is a JIT compiler needing no searching. Then we perform the auto-tuning using FamilySeer, Ansor, and AutoTVM to generate the model codes, and report their end-to-end inference latency. The results indicate that FamilySeer can reach lower latency than others. Although the speedup may be slight, it is still crucial to meet the SLA and improve the user experience in model serving scenarios.

Combined with the pre-trained cost model. We leverage the pre-trained dataset of TenSet, and compare the cost model accuracy of FamilySeer (cost model on the family basis) and TenSet (monolithic cost model). In total, we have collected 4,410,134 training samples to train their cost models. We split the samples into training dataset and testing dataset by subgraphs, where the ratio of training dataset is set to 0.9. The results shows FamilySeer has high cost model accuracy on more than 80% of all subgraphs than TenSet, indicating FamilySeer can also be applied to pre-trained cost models.

ARTIFACT DEPENDENCIES AND REQUIREMENTS

Hardware resources: A machine equipped with a Nvidia GPU (we use the Nvidia V100 GPU)

Operating System: Ubuntu 20.04 (both the host OS and the OS in our docker image)

Software libraries: CUDA 11.2.2, docker, and nvidia-docker2. Others are installed in the docker image

Input datasets: We use eight deep learning models for evaluation, including: ResNet50_v1, ResNet152_v2, Mobilenet, Mobilenetv2, ViT-Huge, BERT-Large, RoBERTa-Large, GPT2-Small.

ARTIFACT INSTALLATION & DEPLOYMENT PROCESS

The following environment setup steps take about one hour totally.

- Download the codes and AE scripts
- Install docker and nvidia-docker
- Build the docker image for FamilySeer's compilation environment
- Run the corresponding familyseer-AE container
- Get into the familyseer-AE container and build FamilySeer binary
- Validate the installation & Reproduce experiments

The details of our experiments can be found in the README.md of https://doi.org/10.5281/zenodo.8197186.

All the experiments take about three weeks totally. Therefore, we have also provided our logs to save the AE reviewers' time. With these logs, the repeoducibility time can be reduced to five hours.

OTHER NOTES

None