FusionBench: Analyzing Kernel Fusion in Vision Models

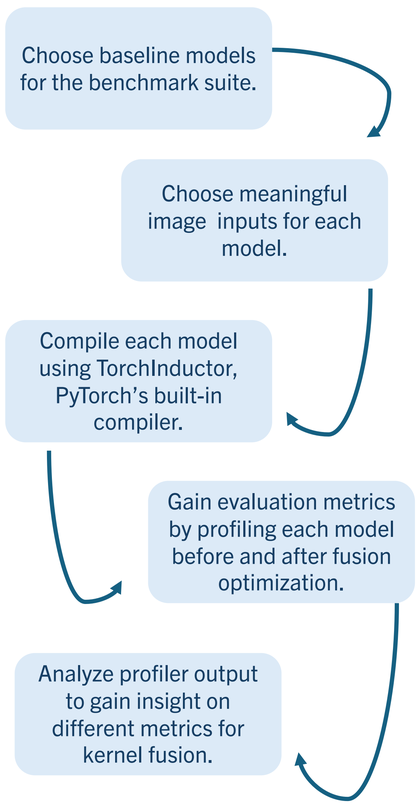
Kamaula Rowe[1], Angélica Aparecida Moreira[2], Tanvir Ahmed Khan[3]
[1]Princeton University, [2]Microsoft Research, [3]Columbia University

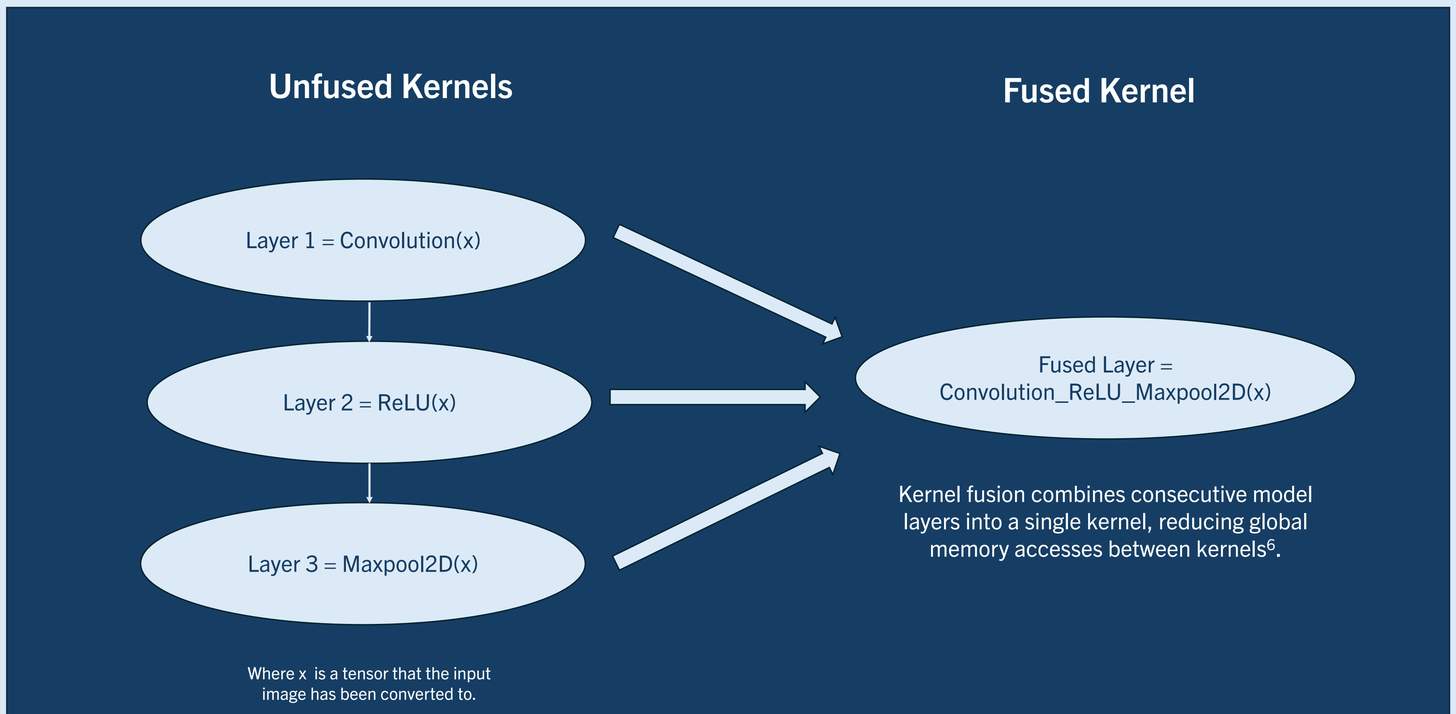
Background

- **PyTorch** is a renowned open-source machine learning (ML) framework used for developing deep learning models[1].
- **Benchmark suites** are collections of tests used to evaluate the performance of applications and systems on specific tasks. Many ML benchmark suites exist, with some of the most renowned being MLPerf[2], SPEC CPU2017[3], and TorchBench[4]. Communities such as HuggingFace[5] also contribute to ML research.
- **Kernel fusion** combines consecutive operators into a single kernel (operation) improving efficiency by minimizing memory access overhead and enhancing computational performance[6].
  - **There are two types of kernel fusion: tensor + element kernel fusion and tensor + element+ tensor kernel fusion[6].**
    - Tensor: data structure similar to an array or matrix
    - Element: Individual data
  - Tensor + Element kernel fusion involves fusing a tensor operation with a more computation-intensive layer such as convolution or attention layers with a memory-bound element operation such as the rectified linear unit(ReLU) or batch normalization. One example is Convolution + ReLU fusion[6].
  - Tensor + Element + Tensor kernel fusion involves fusing consecutive tensor operations, with potential intermediate element operations. One example is Convolution + ReLU + Convolution[6].
- Despite the prevalence of ML benchmarks suites, to our knowledge, **no current suite evaluating kernel fusion currently exists.**

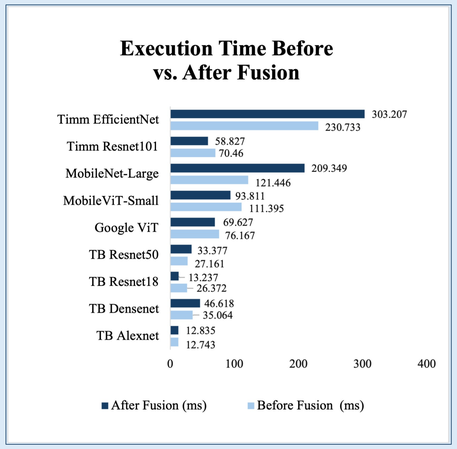
Methodology

Goal: Develop a benchmark suite to study the effectiveness of kernel fusion in improving performance of various ML workloads.

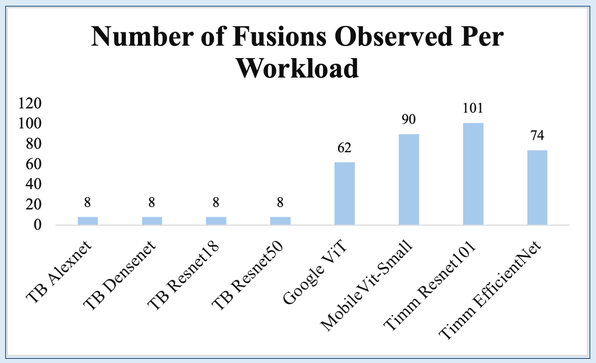




Kernel fusion optimizations **lower execution times for some** models, while increasing them for others.



As shown, kernel fusion optimizations performed by the compiler led to lower execution times for some models and increased times for others, a contrast to expected speedups for all models.



The number of fusions per model varied within the range of 8-101, serving as an example of how the fused output kernels from the compiler depend on the model architecture. Models with lower fusion counts had less layers than those with higher fusion counts. The number of fusions also depends on the types of kernels present, which differ for different types of vision models.

Next Steps

- **Analysis of reasons behind the observed discrepancies is crucial to finding areas for optimization within the TorchInductor compiler and developing a new compiler strategy for ML kernel fusion automation.**
- To achieve this goal, **further evaluation of the way each model is compiled is required.** Since speedups were expected for all models, evaluating what differences occurred with each model's compilation will provide insight into what compiler optimizations led to slowdowns instead.
- **Analyzing the memory usage of each model's inference** before and after each model is compiled is also an area for future observation, as kernel fusion optimization should decrease memory usage.
- **Expanding the benchmark suite** to include a larger and more diverse group of models will also be a priority. The inclusion of more benchmarks will provide further insight into current observations and potentially provide new ones as well.
  - Including natural language processors (NLPs)[7] and speech recognition[8] models will provide insight into other types of kernel fusion performed by TorchInductor.

References

1. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... & Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems, 32.

2. V. J. Reddi et al., "MLPerf Inference Benchmark," 2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA), Valencia, Spain, 2020, pp. 446-459, doi: 10.1109/ISCA45697.2020.00045.

3. Bucek, J., Lange, K. D., & v. Kistowski, J. (2018, April). SPEC CPU2017: Next-generation compute benchmark. In Companion of the 2018 ACM/SPEC International Conference on Performance Engineering (pp. 41-42).

4. Hao, Y., Zhao, X., Bao, B., Berard, D., Constable, W., Aziz, A., & Liu, X. (2023). Torchbench: Benchmarking pytorch with high api surface coverage. arXiv preprint arXiv:2304.14226.

5. Jain, S.M. (2022). Hugging Face. In: Introduction to Transformers for NLP. Apress, Berkeley, CA.

6. W. Sun, A. Li, S. Stuijk and H. Corporaal, "How much can we gain from Tensor Kernel Fusion on GPUs?," in IEEE Access, doi: 10.1109/ACCESS.2024.3411473.

7. Liddy, E. D. (2001). Natural language processing.

8. L. Deng and X. Li, "Machine Learning Paradigms for Speech Recognition: An Overview," in IEEE Transactions on Audio, Speech, and Language Processing, vol. 21, no. 5, pp. 1060-1089, May 2013, doi: 10.1109/TASL.2013.2244083.

COLUMBIA | ENGINEERING
The Fu Foundation School of Engineering and Applied Science

sure

Microsoft Research