

Tensorfolding: Improving Convolutional Neural Network Performance with Fused Microkernels

Michael Anderson, Evangelos Georganas, Sasikanth Avancha, and Alexander Heinecke
Intel Corporation, Parallel Computing Lab, Intel Labs

I. ARTIFACT DESCRIPTION APPENDIX

A. Abstract

This artifact description sketches how to obtain the several software packages needed, how they are compiled and how the reported performance can be re-measured.

B. Description

1) Check-list (artifact meta information):

- **Algorithm:** direct convolutions for deep learning training
- **Program:** LIBXSMM available via github under BSD, https://github.com/mjanderson09/libxsmm/tree/fused_bn3_jit
- **Compilation:** make
- **Data set:** synthetic for kernel tests
- **Run-time environment:** Linux
- **Hardware:** Intel Xeon Scalable Processor (Skylake)
- **Execution:** Via shell scripts/job scheduler
- **Output:** timings and accuracies from logfiles
- **Experiment workflow:** see below
- **Experiment customization:** different dataset for training can be chosen, different hardware platforms can be used.
- **Publicly available?:** yes, on github, BSD license

2) *How software can be obtained (if available):* Via github, more details will be provided after the paper gets accepted.

3) *Hardware dependencies:* Intel Xeon Scalable Processor (Skylake) for the results in this paper. The kernel JITer presented here, also supports Intel SSE3, Intel AVX, Intel AVX2 platforms which is literally every x86 CPU since 2006.

4) Software dependencies:

- 64-bit Linux or Mac-OS. 32-bit OS is not supported.
- GCC, Clang, PGI, Intel or Cray C/C++ compiler
- a BLAS library for fallback code paths

5) *Datasets:* All layer performance runs presented in this work were carried out with runs which auto-generated input data.

C. Installation

1) This Software:

```
git clone xyz
cd xyz
make realclean && make AVX=3 OMP=1 STATIC=1
cd samples/deeplearning/cnnlayer
make realclean && make AVX=3 OMP=1 STATIC=1
```

D. Experiment workflow

1) *This Software:* Then we can run ResNet-50 layers on single-socket Skylake

```
export OMP_NUM_THREADS=28
export KMP_AFFINITY=granularity=fine,compact,1,0
./run_resnet50.sh 28 1000 1 f32 F L 1
./run_resnet50.sh 28 1000 1 f32 B L 1
./run_resnet50.sh 28 1000 1 f32 U L 1
```

E. Evaluation and expected result

Performance can be simply evaluated by console output provided by our simple layer benchmark in GFLOPS and runtime in ms.

Numerical accuracy is provided also by both tests: the layer example runs a simple loop nest as reference code for each convolution operation. The JIT is compared using several norms (Linf of absolute error, L2 of absolute error, Linf of relative error, L2 of relative error).

F. Experiment customization

Whereas the provided scripts focus on the architectures covered in this paper. Based on the information provided here and our github page, it should be fairly simple for the user to adjust the parameters in the provided run scripts.

G. Notes

Our github pages contains far more information than covered here, e.g. on debugger support, performance profile tools support of out JITer and custom configuration of our library.