|  |  |
| --- | --- |
| **Name:** | *Zixiang Xu* |
| **NetID:** | *Zc40* |
| **Section:** | *<class section>* |

**ECE 408/CS483 Milestone 3 Report**

|  |
| --- |
| 1. List Op Times, whole program execution time, and accuracy for batch size of 100, 1k, and 10k images from your basic forward convolution kernel in milestone 2. This will act as your baseline this milestone. |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Batch Size | Op Time 1 | Op Time 2 | Total Execution Time | Accuracy | | 100 | *0.152321 ms* | *0.443814 ms* | *0m1.194s* | *0.86* | | 1000 | *2.66095 ms* | *5.08367 ms* | *0m9.667s* | *0.886* | | 10000 | *25.2527 ms* | *48.8241 ms* | *1m38.197s* | *0.8714* | |
| 1. **Optimization 1: *Constant Memory*** |
| * 1. Which optimization did you choose to implement and why did you choose that optimization technique. |
| *Weight matrix in constant memory.*  *Accessing constant memory saves a lot time than access global memory.* |
| * 1. How does the optimization work? Did you think the optimization would increase performance of the forward convolution? Why? Does the optimization synergize with any of your previous optimizations? |
| *Yes. Accessing constant memory saves a lot time than access global memory. As the weight matrix should be reused for many times, placing it in constant memory should optimize the runtime.* |
| * 1. List the Op Times, whole program execution time, and accuracy for batch size of 100, 1k, and 10k images using this optimization (including any previous optimizations also used). |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Batch Size | Op Time 1 | Op Time 2 | Total Execution Time | Accuracy | | 100 | *0.151995 ms* | *0.451659 ms* | *0m1.188s* | *0.86* | | 1000 | *2.18526 ms* | *4.66564 ms* | *0m10.397s* | *0.886* | | 10000 | *28.2483 ms* | *54.004 ms* | *1m41.343s* | *0.8714* | |
| * 1. Was implementing this optimization successful in improving performance? Why or why not? Include profiling results from *nsys* and *Nsight-Compute* to justify your answer, directly comparing to your baseline (or the previous optimization this one is built off of). |
| *No.* |
| * 1. What references did you use when implementing this technique? |
| *<answer here>* |
| 1. **Optimization 2: *FP16*** |
| 1. Which optimization did you choose to implement and why did you choose that optimization technique. |
| *Fixed point (FP16) arithmetic。*  *FP16 type and arithmetic fits GPU better than float type and calculation.* |
| 1. How does the optimization work? Did you think the optimization would increase performance of the forward convolution? Why? Does the optimization synergize with any of your previous optimizations? |
| *FP16 type and arithmetic fits GPU better than float type and calculation.* |
| 1. List the Op Times, whole program execution time, and accuracy for batch size of 100, 1k, and 10k images using this optimization (including any previous optimizations also used). |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Batch Size | Op Time 1 | Op Time 2 | Total Execution Time | Accuracy | | 100 | *0.150745 ms* | *0.466857 ms* | *0m1.114s* | *0.86* | | 1000 | *2.45225 ms* | *5.09768 ms* | *0m9.984s* | *0.886* | | 10000 | *28.3144 ms* | *53.9818 ms* | *1m36.424s* | *0.8714* | |
| 1. Was implementing this optimization successful in improving performance? Why or why not? Include profiling results from *nsys* and *Nsight-Compute* to justify your answer, directly comparing to your baseline (or the previous optimization this one is built off of). |
| *<answer here>* |
| 1. What references did you use when implementing this technique? |
| *<answer here>* |

|  |
| --- |
| 1. **Optimization 3: *Stream***   ***(Delete this section blank if you did not implement this many optimizations.)*** |
| * 1. Which optimization did you choose to implement and why did you choose that optimization technique. |
| *<answer here>* |
| * 1. How does the optimization work? Did you think the optimization would increase performance of the forward convolution? Why? Does the optimization synergize with any of your previous optimizations? |
| *<answer here>* |
| * 1. List the Op Times, whole program execution time, and accuracy for batch size of 100, 1k, and 10k images using this optimization (including any previous optimizations also used). |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Batch Size | Op Time 1 | Op Time 2 | Total Execution Time | Accuracy | | 100 | *0.150579 ms* | *0.466815 ms* | *0m1.167s* | *0.86* | | 1000 | *2.1814 ms* | *4.6481 ms* | *0m9.788s* | *0.886* | | 10000 | *28.2112 ms* | *54.0234 ms* | *1m38.814s* | *0.8714* | |
| * 1. Was implementing this optimization successful in improving performance? Why or why not? Include profiling results from *nsys* and *Nsight-Compute* to justify your answer, directly comparing to your baseline (or the previous optimization this one is built off of). |
| *<answer here>* |
| * 1. What references did you use when implementing this technique? |
| *<answer here>* |
| 1. **Optimization 4: *Tiled Convolution***   ***(Delete this section blank if you did not implement this many optimizations.)*** |
| * 1. Which optimization did you choose to implement and why did you choose that optimization technique. |
| *<answer here>* |
| * 1. How does the optimization work? Did you think the optimization would increase performance of the forward convolution? Why? Does the optimization synergize with any of your previous optimizations? |
| *<answer here>* |
| * 1. List the Op Times, whole program execution time, and accuracy for batch size of 100, 1k, and 10k images using this optimization (including any previous optimizations also used). |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Batch Size | Op Time 1 | Op Time 2 | Total Execution Time | Accuracy | | 100 | *<op\_time>* | *<op\_time>* | *<exec\_time>* | *<accuracy>* | | 1000 | *2.44749 ms* | *5.09461 ms* | *0m9.969s* | *0.886* | | 10000 | *28.2405 ms* | *54.0094 ms* | *1m38.324s* | *0.8714* | |
| * 1. Was implementing this optimization successful in improving performance? Why or why not? Include profiling results from *nsys* and *Nsight-Compute* to justify your answer, directly comparing to your baseline (or the previous optimization this one is built off of). |
| *No.*  *CUDA Kernel Statistics (nanoseconds)*  *Time(%) Total Time Instances Average Minimum Maximum Name*  *------- -------------- ---------- -------------- -------------- -------------- --------------------------------------------------------------------------------*  *100.0 73658614 2 36829307.0 24857456 48801158 conv\_forward\_kernel*  *0.0 2784 2 1392.0 1376 1408 do\_not\_remove\_this\_kernel*  *0.0 2656 2 1328.0 1280 1376 prefn\_marker\_kernel* |
| * 1. What references did you use when implementing this technique? |
| *<answer here>* |
| 1. **Optimization 5: Tiled Convolution + FP16 + Stream** |
| * 1. Which optimization did you choose to implement and why did you choose that optimization technique. |
| *<answer here>* |
| * 1. How does the optimization work? Did you think the optimization would increase performance of the forward convolution? Why? Does the optimization synergize with any of your previous optimizations? |
| *<answer here>* |
| * 1. List the Op Times, whole program execution time, and accuracy for batch size of 100, 1k, and 10k images using this optimization (including any previous optimizations also used). |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Batch Size | Op Time 1 | Op Time 2 | Total Execution Time | Accuracy | | 100 | *0.152563 ms* | *0.533814 ms* | *0m1.170s* | *0.86* | | 1000 | *2.69325 ms* | *5.08583 ms* | *0m10.026s* | *0.886* | | 10000 | *27.8238 ms* | *53.9145 ms* | *1m38.415s* | *0.8714* | |
| * 1. Was implementing this optimization successful in improving performance? Why or why not? Include profiling results from *nsys* and *Nsight-Compute* to justify your answer, directly comparing to your baseline (or the previous optimization this one is built off of). |
| *<answer here>* |
| * 1. What references did you use when implementing this technique? |
| *<answer here>* |
| 1. **Optimization 6: *<optimization name>***   ***(Delete this section if you did not implement this many optimizations.)*** |
| * 1. Which optimization did you choose to implement and why did you choose that optimization technique. |
| *<answer here>* |
| * 1. How does the optimization work? Did you think the optimization would increase performance of the forward convolution? Why? Does the optimization synergize with any of your previous optimizations? |
| *<answer here>* |
| * 1. List the Op Times, whole program execution time, and accuracy for batch size of 100, 1k, and 10k images using this optimization (including any previous optimizations also used). |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | Batch Size | Op Time 1 | Op Time 2 | Total Execution Time | Accuracy | | 100 | *<op\_time>* | *<op\_time>* | *<exec\_time>* | *<accuracy>* | | 1000 | *<op\_time>* | *<op\_time>* | *<exec\_time>* | *<accuracy>* | | 10000 | *<op\_time>* | *<op\_time>* | *<exec\_time>* | *<accuracy>* | |
| * 1. Was implementing this optimization successful in improving performance? Why or why not? Include profiling results from *nsys* and *Nsight-Compute* to justify your answer, directly comparing to your baseline (or the previous optimization this one is built off of). |
| *<answer here>* |
| * 1. What references did you use when implementing this technique? |
| *<answer here>* |