Cloud and Machine Learning

Project 1: "ImageNet on BareMetal/Cloud"

PART 1 : Running ImageNet on Bare metal

The steps followed have been listed below:

1. Access the Price Cluster

   Use an SSH Client to connect to the Prince Cluster:

   - We used PuTTY to connect to host gw.hpc.nyu.edu on port 22 using SSH protocol.
   - Enter your NetID password and you should be authorized on the gateway server.
   - Use the following ssh command to connect to the prince cluster.
     ssh prince.hpc.nyu.edu

2. Get access to the ImageNet dataset that we will be using for the project. The dataset can be found on the prince cluster /scratch/work/public/imagenet

3. We proceed to create a small subset of the training data for our use under /home/net-id
   We divide the data into train and val subsets that will be used for the training and validation of the model we create.

4. Get the Imagenet example from pytorch.
   git clone https://github.com/pytorch/examples

5. Move to the imagenet folder containing the python code.
   cd examples → cd imagenet

6. Get an interactive node
   Login into frontend node. ssh prince1.hpc.nyu.edu
   srun -t10:00:00 --gres=gpu:1 --mem 102400 --pty /bin/bash

7. Load module. We are using the stable version.
   module load python3/intel/3.6.3 cuda/9.0.176 nccl/cuda9.0/2.4.2

8. Setup the python virtual environment.
   - mkdir pytorch_env
   - cd pytorch_env
   - virtualenv --system-site-packages py3.6.3
   - source py3.6.3/bin/activate

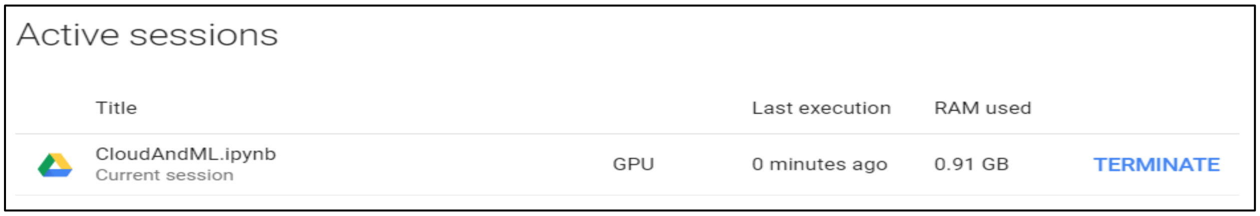
- Install pytorch
  pip3 install [http://download.pytorch.org/whl/cu92/torch-0.4.1- cp36-cp36m-linux_x86_64.whl](http://download.pytorch.org/whl/cu92/torch-0.4.1-cp36-cp36m-linux_x86_64.whl)
- Install torchvision
  pip3 install torchvision

9. Activate the virtual environment.
   source ~/pytorch_env/py3.6.3/bin/activate
10. Run the job. To train the model we run 'main.py' with the desired model architecture(Alexnet) and a path to the Imagenet dataset.
    python /home/as13594/examples/imagenet/main.py -a alexnet -b 8 --epochs 1 --lr 0.01 /home/as13594/
11. Use nvprof for profiling the performance and run the job on the reserved GPU nodes using the following command
    srun --reservation=chung --gres=gpu:1 --time=01:00:00 --gres=gpu:p40:4 --cpus-per-task=28 nvprof python /home/as13594/examples/imagenet/main.py -a alexnet -b 20 --epochs 1 --lr 0.01 /home/as13594/

PART 2 : <u>Running ImageNet on Cloud</u>

For the second part of the project we run the code on a Google Colab notebook on the AWS Cloud Computing Platform.

1. We begin with login onto the Google Colaboratory.
2. Upload the python file containing the source code that needs to be run on the drive.
3. Create a notebook instance.
4. Once the notebook instance has been created, we mount the drive containing the data(testing and validation) to be used for building the model.
   from google.colab import drive
   drive.mount('/content/gdrive')
5. Change runtime type to GPU.
6. Run the code.
   python /home/as13594/examples/imagenet/main.py -a alexnet -b 8 --epochs 1 --lr 0.01 /home/as13594/



Active sessions

Title		Last execution	RAM used	
CloudAndML.ipynb Current session	GPU	0 minutes ago	0.91 GB	TERMINATE

Implementation and Analysis:

As a part of this project we attempt to train a model with the desired model architecture(Alexnet) on the Imagenet dataset using pytorch and run it on both BareMetal and Cloud platforms to access its relative performance on both platforms.

1. **Usability:**
   - **BareMetal-** A bare metal server is a single tenant physical server. Since they offer single-tenet environments i.e. a single servers' physical resources may not be shared between two or more tenets, they can be used to run dedicated services without any interruptions for longer durations. Bare-metal servers offer isolation and are free of the "noisy neighbor " effect that plagues virtual environments. Network latency is minimized for better performance, and the tenant enjoys root access. Bare metal is highly customizable, and the tenant may optimize the server based upon their individual needs.

     Bare metal servers do not require the use of several layers of software, unlike the virtual environment, which has at least one additional layer of software – a Type 1 hypervisor. This implies that there is one less layer of software between the user and the physical hardware in everyday use. Hence, we can expect better performance.
   - **Cloud** : Cloud offers a distributed environment comprised of multi-tenant, virtualized servers. The host machine shares its resources with multiple virtual instances. They each get a portion of CPU, RAM and storage. Cloud Servers allow you to add resources to individual virtual machines (vertical scaling) or add whole new servers (horizontal scaling) at any time, in a matter of minutes. This scalability makes Cloud Servers better suited to variable workloads, where the ability to dynamically scale performance is more important than sheer horsepower. Cloud computing environments are more prone to latency for various reasons. For example, if VMs are on separate networks, it can lead to packet delays. With cloud environments, you do not have a direct connection with the physical hardware, as there is a hypervisor layer between your app and physical resources. Thus, the chances are that VMs will suffer from a higher latency than if you were running apps directly on a bare metal server. Furthermore, performance bottlenecks may occur due to the sheer number of tenants. If you have noisy neighbors who like to run resource-intensive workloads on their share of the server, they may very well impact you leading to degraded performance.

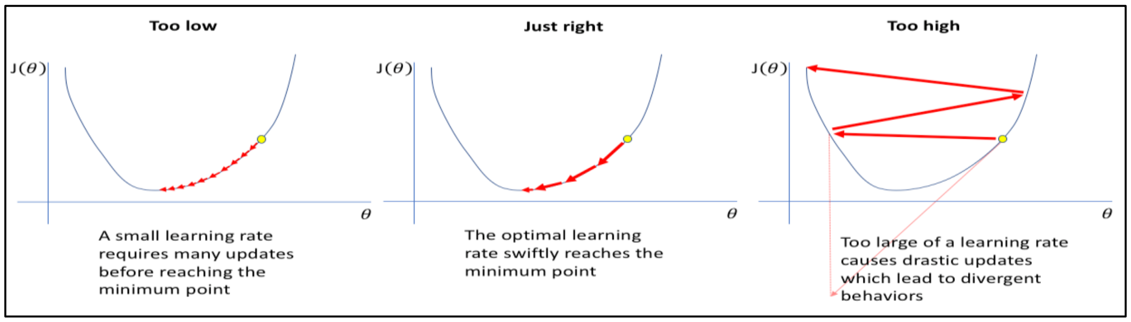
   Analysis of Usability between BareMetal and Cloud:
   Running our program on both BareMetal and Cloud environments gave us a chance to explore the pros and cons of both domains.

   - **Google Collaboratory offered a more friendly user interface. It was relatively easy to mount data and process it on a GPU using a notebook. There was no need for the user to manage the dependencies/configuration of the execution environment.**
   - **In case of BareMetal, we had to set up our own virtual execution environment before we were able to run any jobs on the GPU. It was relatively more tedious to transfer the data to the server and to run the jobs.**
   - **GPU based cloud computing provides the ease of scalability as opposed to BareMetal Servers. This makes them fit for variable workloads.**
   - **Since resources in the Bare Metal environments are dedicated to users this could lead to under -utilization of resources as opposed to cloud environments that offer better utilization of resources.**
   - **Bare-metal environments are single tenet systems and are comparatively more secure.**

2. **Hyperparameter settings:**

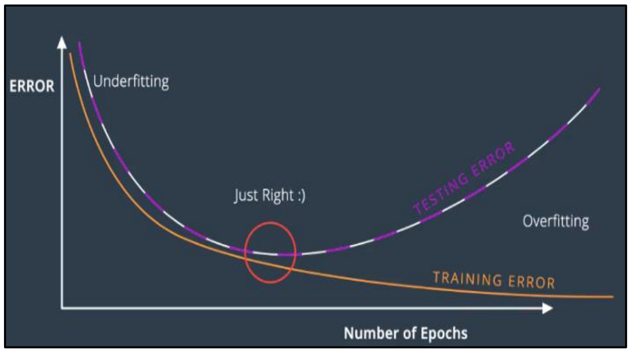
- **Learning Rate :** The weights of a neural network cannot be calculated using an analytical method. Instead, the weights must be discovered via an empirical optimization procedure called gradient descent.
  The learning rate is a hyperparameter that controls how much to change the model in response to the estimated error each time the model weights are updated. This parameter scales the magnitude of our weight updates in order to minimize the network's loss function. Choosing the learning rate is challenging as a value too small may result in a long training process that could get stuck, whereas a value too large may result in learning a sub-optimal set of weights and too fast or unstable training process.
  We might start with a large value like 0.1, then try exponentially lower values: 0.01, 0.001, etc. The training should start from a relatively large learning rate because, in the beginning, random weights are far from optimal, and then the learning rate can decrease during training to allow more fine-grained weight updates. Training with a smaller learning rate would allow more fine-grained weight updates.

  `Note: We use 0.01 as the initial learning rate for AlexNet and then change it to .05 and .0001`


Too low
Just right
Too high
$J(\theta)$
θ
A small learning rate requires many updates before reaching the minimum point
The optimal learning rate swiftly reaches the minimum point
Too large of a learning rate causes drastic updates which lead to divergent behaviors

- **BATCH SIZE**: Batch size is used in machine learning to refer to the number of training examples utilized in one iteration. Choosing a batch size that is too small will introduce a high degree of variance within each batch as it is unlikely that a small sample is a good representation of the entire dataset. Generally, the larger the batch size the faster the model will complete each epoch in training. However, the tradeoff is that the quality of the model may degrade as we increase the batch size. If a batch size is too large, it may not fit in memory of the compute instance used for training and it will have the tendency to overfit the data.
  Lowering the learning rate and decreasing the batch size will allow the network to train better, especially in the case of fine-tuning.

- **EPOCH:** The number of epochs is a hyperparameter of gradient descent that controls the number of complete passes through the training dataset. The number of epoch will decide- how many times we will change the weights of the network.
  As the number of epochs increases, the number of times weights are changed in the neural network increases and the boundary goes from underfitting to optimal to overfitting.



3. **Performance:**

BareMetal vs. Cloud Platforms: At a Glance

Deep learning consumes huge amount of computing capacity, because it often relies on deeper networks and larger training datasets to improve the model accuracy

Our experiments focuses on following performance metrics:

(1) **Throughput** measured by the number of images processed per second during the training steps, and

(2) **Time elapsed:** measured by the total execution time from the time a training job is launched until it finished.

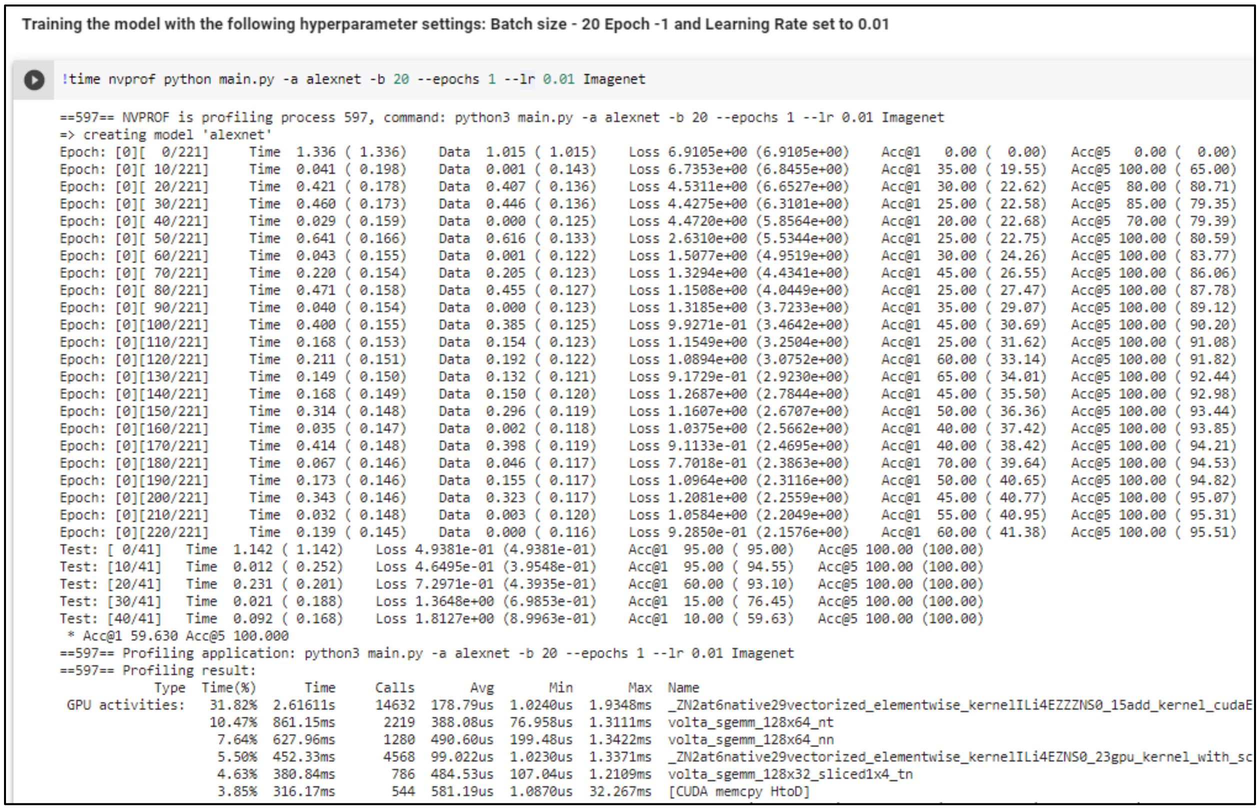
In other words, the time elapsed metric includes the performance impact of data loading and pre-processing while the throughput metric does not thereby justifying our choice of metrics.

<u>**Performance Analysis:**</u>

Example Run 1

- **Running on Cloud:**

We begin with training the model with the following hyperparameter settings: Batch size - 20 Epoch -1 and the Learning Rate has been set to 0.01. It takes approximately 221 iterations and processes images in batches of 20 and passes over the dataset one time.

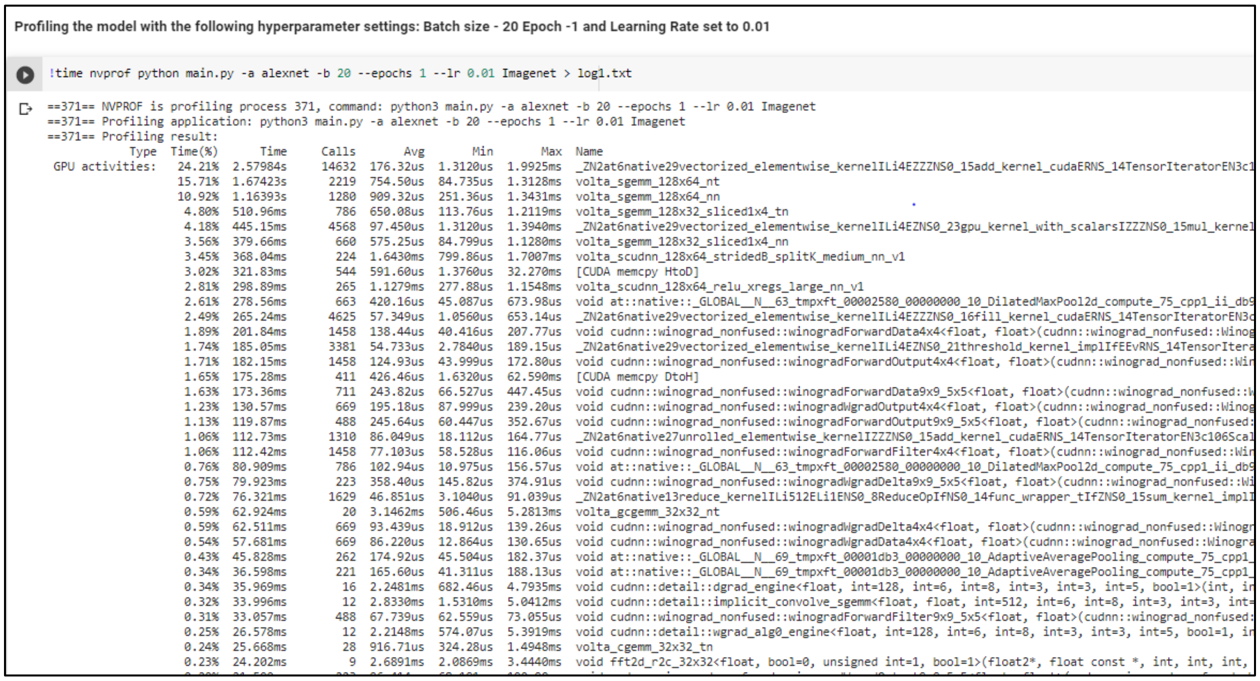


Training the model with the following hyperparameter settings: Batch size - 20 Epoch -1 and Learning Rate set to 0.01

```
!time nvprof python main.py -a alexnet -b 20 --epochs 1 --lr 0.01 Imagenet
```

```
==597== NVPROF is profiling process 597, command: python3 main.py -a alexnet -b 20 --epochs 1 --lr 0.01 Imagenet
=> creating model 'alexnet'
Epoch: [0][  0/221]    Time  1.336 ( 1.336)    Data  1.015 ( 1.015)    Loss 6.9105e+00 (6.9105e+00)    Acc@1   0.00 (  0.00)    Acc@5   0.00 (  0.00)
Epoch: [0][ 10/221]    Time  0.041 ( 0.198)    Data  0.001 ( 0.143)    Loss 6.7353e+00 (6.8455e+00)    Acc@1  35.00 ( 19.55)    Acc@5 100.00 ( 65.00)
Epoch: [0][ 20/221]    Time  0.421 ( 0.178)    Data  0.407 ( 0.136)    Loss 4.5311e+00 (6.6527e+00)    Acc@1  30.00 ( 22.62)    Acc@5  80.00 ( 80.71)
Epoch: [0][ 30/221]    Time  0.460 ( 0.173)    Data  0.446 ( 0.136)    Loss 4.4275e+00 (6.3101e+00)    Acc@1  25.00 ( 22.58)    Acc@5  85.00 ( 79.35)
Epoch: [0][ 40/221]    Time  0.029 ( 0.159)    Data  0.000 ( 0.125)    Loss 4.4720e+00 (5.8564e+00)    Acc@1  20.00 ( 22.68)    Acc@5  70.00 ( 79.39)
Epoch: [0][ 50/221]    Time  0.641 ( 0.166)    Data  0.616 ( 0.133)    Loss 2.6310e+00 (5.5344e+00)    Acc@1  25.00 ( 22.75)    Acc@5 100.00 ( 80.59)
Epoch: [0][ 60/221]    Time  0.043 ( 0.155)    Data  0.001 ( 0.122)    Loss 1.5077e+00 (4.9519e+00)    Acc@1  30.00 ( 24.26)    Acc@5 100.00 ( 83.77)
Epoch: [0][ 70/221]    Time  0.220 ( 0.154)    Data  0.205 ( 0.123)    Loss 1.3294e+00 (4.4341e+00)    Acc@1  45.00 ( 26.55)    Acc@5 100.00 ( 86.06)
Epoch: [0][ 80/221]    Time  0.471 ( 0.158)    Data  0.455 ( 0.127)    Loss 1.1508e+00 (4.0449e+00)    Acc@1  25.00 ( 27.47)    Acc@5 100.00 ( 87.78)
Epoch: [0][ 90/221]    Time  0.040 ( 0.154)    Data  0.000 ( 0.123)    Loss 1.3185e+00 (3.7233e+00)    Acc@1  35.00 ( 29.07)    Acc@5 100.00 ( 89.12)
Epoch: [0][100/221]    Time  0.400 ( 0.155)    Data  0.385 ( 0.125)    Loss 9.9271e-01 (3.4642e+00)    Acc@1  45.00 ( 30.69)    Acc@5 100.00 ( 90.20)
Epoch: [0][110/221]    Time  0.168 ( 0.153)    Data  0.154 ( 0.123)    Loss 1.1549e+00 (3.2504e+00)    Acc@1  25.00 ( 31.62)    Acc@5 100.00 ( 91.08)
Epoch: [0][120/221]    Time  0.211 ( 0.151)    Data  0.192 ( 0.122)    Loss 1.0894e+00 (3.0752e+00)    Acc@1  60.00 ( 33.14)    Acc@5 100.00 ( 91.82)
Epoch: [0][130/221]    Time  0.149 ( 0.150)    Data  0.132 ( 0.121)    Loss 9.1729e-01 (2.9230e+00)    Acc@1  65.00 ( 34.01)    Acc@5 100.00 ( 92.44)
Epoch: [0][140/221]    Time  0.168 ( 0.149)    Data  0.150 ( 0.120)    Loss 1.2687e+00 (2.7844e+00)    Acc@1  45.00 ( 35.50)    Acc@5 100.00 ( 92.98)
Epoch: [0][150/221]    Time  0.314 ( 0.148)    Data  0.296 ( 0.119)    Loss 1.1607e+00 (2.6707e+00)    Acc@1  50.00 ( 36.36)    Acc@5 100.00 ( 93.44)
Epoch: [0][160/221]    Time  0.035 ( 0.147)    Data  0.002 ( 0.118)    Loss 1.0375e+00 (2.5662e+00)    Acc@1  40.00 ( 37.42)    Acc@5 100.00 ( 93.85)
Epoch: [0][170/221]    Time  0.414 ( 0.148)    Data  0.398 ( 0.119)    Loss 9.1133e-01 (2.4695e+00)    Acc@1  40.00 ( 38.42)    Acc@5 100.00 ( 94.21)
Epoch: [0][180/221]    Time  0.067 ( 0.146)    Data  0.046 ( 0.117)    Loss 7.7018e-01 (2.3863e+00)    Acc@1  70.00 ( 39.64)    Acc@5 100.00 ( 94.53)
Epoch: [0][190/221]    Time  0.173 ( 0.146)    Data  0.155 ( 0.117)    Loss 1.0964e+00 (2.3116e+00)    Acc@1  50.00 ( 40.65)    Acc@5 100.00 ( 94.82)
Epoch: [0][200/221]    Time  0.343 ( 0.146)    Data  0.323 ( 0.117)    Loss 1.2081e+00 (2.2559e+00)    Acc@1  45.00 ( 40.77)    Acc@5 100.00 ( 95.07)
Epoch: [0][210/221]    Time  0.032 ( 0.148)    Data  0.003 ( 0.120)    Loss 1.0584e+00 (2.2049e+00)    Acc@1  55.00 ( 40.95)    Acc@5 100.00 ( 95.31)
Epoch: [0][220/221]    Time  0.139 ( 0.145)    Data  0.000 ( 0.116)    Loss 9.2850e-01 (2.1576e+00)    Acc@1  60.00 ( 41.38)    Acc@5 100.00 ( 95.51)
Test: [ 0/41]   Time  1.142 ( 1.142)    Loss 4.9381e-01 (4.9381e-01)    Acc@1  95.00 ( 95.00)    Acc@5 100.00 (100.00)
Test: [10/41]   Time  0.012 ( 0.252)    Loss 4.6495e-01 (3.9548e-01)    Acc@1  95.00 ( 94.55)    Acc@5 100.00 (100.00)
Test: [20/41]   Time  0.231 ( 0.201)    Loss 7.2971e-01 (4.3935e-01)    Acc@1  60.00 ( 93.10)    Acc@5 100.00 (100.00)
Test: [30/41]   Time  0.021 ( 0.188)    Loss 1.3648e+00 (6.9853e-01)    Acc@1  15.00 ( 76.45)    Acc@5 100.00 (100.00)
Test: [40/41]   Time  0.092 ( 0.168)    Loss 1.8127e+00 (8.9963e-01)    Acc@1  10.00 ( 59.63)    Acc@5 100.00 (100.00)
 * Acc@1 59.630 Acc@5 100.000
==597== Profiling application: python3 main.py -a alexnet -b 20 --epochs 1 --lr 0.01 Imagenet
==597== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   31.82%  2.61611s     14632  178.79us  1.0240us  1.9348ms  _ZN2at6native29vectorized_elementwise_kernelILi4EZZZNS0_15add_kernel_cudaE
                   10.47%  861.15ms      2219  388.08us  76.958us  1.3111ms  volta_sgemm_128x64_nt
                    7.64%  627.96ms      1280  490.60us  199.48us  1.3422ms  volta_sgemm_128x64_nn
                    5.50%  452.33ms      4568  99.022us  1.0230us  1.3371ms  _ZN2at6native29vectorized_elementwise_kernelILi4EZNS0_23gpu_kernel_with_sc
                    4.63%  380.84ms       786  484.53us  107.04us  1.2109ms  volta_sgemm_128x32_sliced1x4_tn
                    3.85%  316.17ms       544  581.19us  1.0870us  32.267ms  [CUDA memcpy HtoD]
```

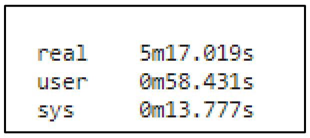
Performance Profiling using nvprof:



Profiling the model with the following hyperparameter settings: Batch size - 20 Epoch -1 and Learning Rate set to 0.01

```
!time nvprof python main.py -a alexnet -b 20 --epochs 1 --lr 0.01 Imagenet > log1.txt
```

```
==371== NVPROF is profiling process 371, command: python3 main.py -a alexnet -b 20 --epochs 1 --lr 0.01 Imagenet
==371== Profiling application: python3 main.py -a alexnet -b 20 --epochs 1 --lr 0.01 Imagenet
==371== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   24.21%  2.57984s     14632  176.32us  1.3120us  1.9925ms  _ZN2at6native29vectorized_elementwise_kernelILi4EZZZNS0_15add_kernel_cudaERNS_14TensorIteratorEN3c1
                   15.71%  1.67423s      2219  754.50us  84.735us  1.3128ms  volta_sgemm_128x64_nt
                   10.92%  1.16393s      1280  909.32us  251.36us  1.3431ms  volta_sgemm_128x64_nn
                    4.80%  510.96ms       786  650.08us  113.76us  1.2119ms  volta_sgemm_128x32_sliced1x4_tn
                    4.18%  445.15ms      4568  97.450us  1.3120us  1.3940ms  _ZN2at6native29vectorized_elementwise_kernelILi4EZNS0_23gpu_kernel_with_scalarsIZZZNS0_15mul_kernel
                    3.56%  379.66ms       660  575.25us  84.799us  1.1280ms  volta_sgemm_128x32_sliced1x4_nn
                    3.45%  368.04ms       224  1.6430ms  799.86us  1.7007ms  volta_scudnn_128x64_stridedB_splitK_medium_nn_v1
                    3.02%  321.83ms       544  591.60us  1.3760us  32.270ms  [CUDA memcpy HtoD]
                    2.81%  298.89ms       265  1.1279ms  277.88us  1.1548ms  volta_scudnn_128x64_relu_xregs_large_nn_v1
                    2.61%  278.56ms       663  420.16us  45.087us  673.98us  void at::native::_GLOBAL__N__63_tmpxft_00002580_00000000_10_DilatedMaxPool2d_compute_75_cpp1_ii_db9
                    2.49%  265.24ms      4625  57.349us  1.0560us  653.14us  _ZN2at6native29vectorized_elementwise_kernelILi4EZZZNS0_16fill_kernel_cudaERNS_14TensorIteratorEN3c
                    1.89%  201.84ms      1458  138.44us  40.416us  207.77us  void cudnn::winograd_nonfused::winogradForwardData4x4<float, float>(cudnn::winograd_nonfused::Winog
                    1.74%  185.05ms      3381  54.733us  2.7840us  189.15us  _ZN2at6native29vectorized_elementwise_kernelILi4EZNS0_21threshold_kernel_implIfEEvRNS_14TensorItera
                    1.71%  182.15ms      1458  124.93us  43.999us  172.80us  void cudnn::winograd_nonfused::winogradForwardOutput4x4<float, float>(cudnn::winograd_nonfused::Wir
                    1.65%  175.28ms       411  426.46us  1.6320us  62.590ms  [CUDA memcpy DtoH]
                    1.63%  173.36ms       711  243.82us  66.527us  447.45us  void cudnn::winograd_nonfused::winogradForwardData9x9_5x5<float, float>(cudnn::winograd_nonfused::W
                    1.23%  130.57ms       669  195.18us  87.999us  239.20us  void cudnn::winograd_nonfused::winogradWgradOutput4x4<float, float>(cudnn::winograd_nonfused::Winog
                    1.13%  119.87ms       488  245.64us  60.447us  352.67us  void cudnn::winograd_nonfused::winogradForwardOutput9x9_5x5<float, float>(cudnn::winograd_nonfused:
                    1.06%  112.73ms      1310  86.049us  18.112us  164.77us  _ZN2at6native27unrolled_elementwise_kernelIZZZNS0_15add_kernel_cudaERNS_14TensorIteratorEN3c106Scal
                    1.06%  112.42ms      1458  77.103us  58.528us  116.06us  void cudnn::winograd_nonfused::winogradForwardFilter4x4<float, float>(cudnn::winograd_nonfused::Wir
                    0.76%  80.909ms       786  102.94us  10.975us  156.57us  void at::native::_GLOBAL__N__63_tmpxft_00002580_00000000_10_DilatedMaxPool2d_compute_75_cpp1_ii_db9
                    0.75%  79.923ms       223  358.40us  145.82us  374.91us  void cudnn::winograd_nonfused::winogradWgradDelta9x9_5x5<float, float>(cudnn::winograd_nonfused::Wi
                    0.72%  76.321ms      1629  46.851us  3.1040us  91.039us  _ZN2at6native13reduce_kernelILi512ELi1ENS0_8ReduceOpIfNS0_14func_wrapper_tIfZNS0_15sum_kernel_implI
                    0.59%  62.924ms        20  3.1462ms  506.46us  5.2813ms  volta_gcgemm_32x32_nt
                    0.59%  62.511ms       669  93.439us  18.912us  139.26us  void cudnn::winograd_nonfused::winogradWgradDelta4x4<float, float>(cudnn::winograd_nonfused::Winogr
                    0.54%  57.681ms       669  86.220us  12.864us  130.65us  void cudnn::winograd_nonfused::winogradWgradData4x4<float, float>(cudnn::winograd_nonfused::Winogra
                    0.43%  45.828ms       262  174.92us  45.504us  182.37us  void at::native::_GLOBAL__N__69_tmpxft_00001db3_00000000_10_AdaptiveAveragePooling_compute_75_cpp1_
                    0.34%  36.598ms       221  165.60us  41.311us  188.13us  void at::native::_GLOBAL__N__69_tmpxft_00001db3_00000000_10_AdaptiveAveragePooling_compute_75_cpp1_
                    0.34%  35.969ms        16  2.2481ms  682.46us  4.7935ms  void cudnn::detail::dgrad_engine<float, int=128, int=6, int=8, int=3, int=3, int=5, bool=1>(int, ir
                    0.32%  33.996ms        12  2.8330ms  1.5310ms  5.0412ms  void cudnn::detail::implicit_convolve_sgemm<float, float, int=512, int=6, int=8, int=3, int=3, int=
                    0.31%  33.057ms       488  67.739us  62.559us  73.055us  void cudnn::winograd_nonfused::winogradForwardFilter9x9_5x5<float, float>(cudnn::winograd_nonfused:
                    0.25%  26.578ms        12  2.2148ms  574.07us  5.3919ms  void cudnn::detail::wgrad_alg0_engine<float, int=128, int=6, int=8, int=3, int=3, int=5, bool=1, ir
                    0.24%  25.668ms        28  916.71us  324.28us  1.4948ms  volta_cgemm_32x32_tn
                    0.23%  24.202ms         9  2.6891ms  2.0869ms  3.4440ms  void fft2d_r2c_32x32<float, bool=0, unsigned int=1, bool=1>(float2*, float const *, int, int, int,
```

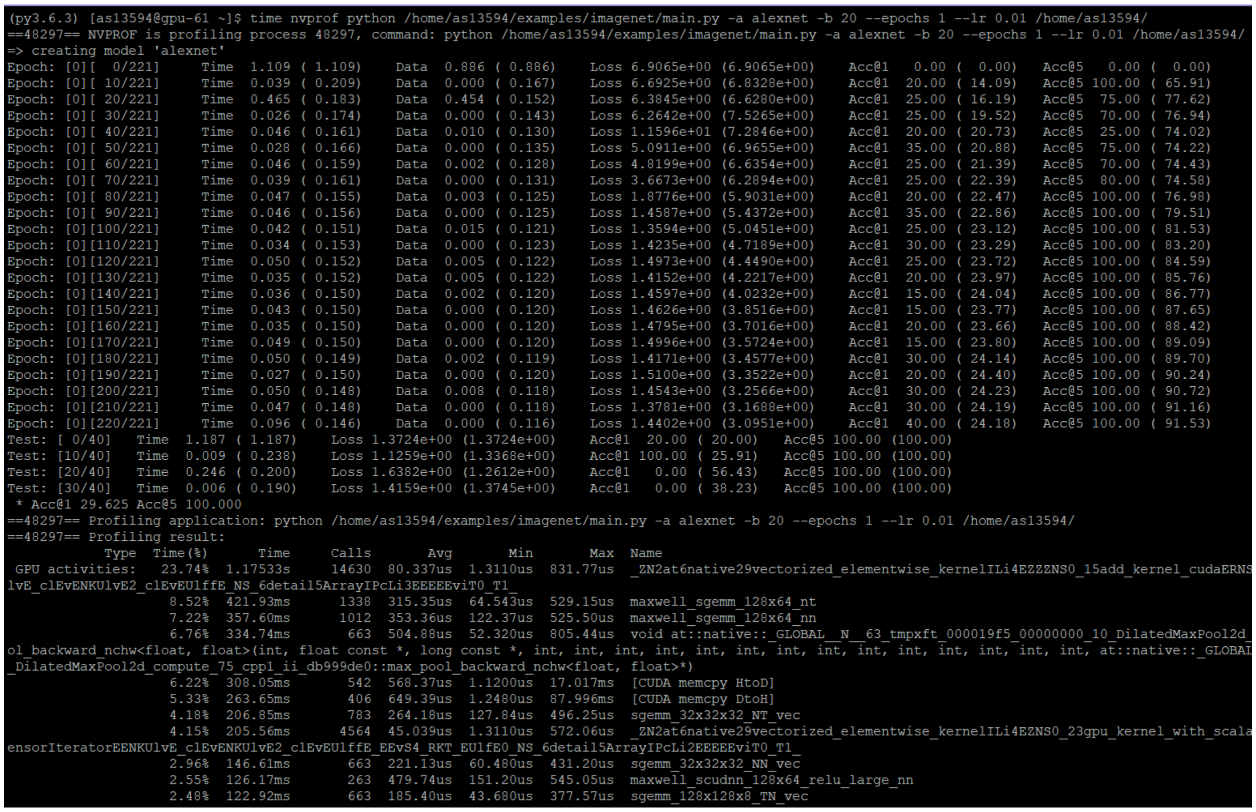


```
real    5m17.019s
user    0m58.431s
sys     0m13.777s
```

The time taken to finish training for this run is very less as could be expected from because we started with a greater value of learning rate which determines the step size. One can also note that the accuracy is just 25%.

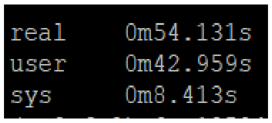
- **Running on BareMetal:**

We train the model with the same hyperparameter settings: Batch size - 20 Epoch -1 and the Learning Rate has been set to 0.01. It takes approximately 221 iterations and processes images in batches of 20 and passes over the dataset one time.



```
(py3.6.3) [as13594@gpu-61 ~]$ time nvprof python /home/as13594/examples/imagenet/main.py -a alexnet -b 20 --epochs 1 --lr 0.01 /home/as13594/
==48297== NVPROF is profiling process 48297, command: python /home/as13594/examples/imagenet/main.py -a alexnet -b 20 --epochs 1 --lr 0.01 /home/as13594/
=> creating model 'alexnet'
Epoch: [0][  0/221]     Time  1.109 ( 1.109)    Data  0.886 ( 0.886)    Loss 6.9065e+00 (6.9065e+00)    Acc@1   0.00 (  0.00)   Acc@5   0.00 (  0.00)
Epoch: [0][ 10/221]     Time  0.039 ( 0.209)    Data  0.000 ( 0.167)    Loss 6.6925e+00 (6.8328e+00)    Acc@1  20.00 ( 14.09)   Acc@5 100.00 ( 65.91)
Epoch: [0][ 20/221]     Time  0.465 ( 0.183)    Data  0.454 ( 0.152)    Loss 6.3845e+00 (6.6280e+00)    Acc@1  25.00 ( 16.19)   Acc@5  75.00 ( 77.62)
Epoch: [0][ 30/221]     Time  0.026 ( 0.174)    Data  0.000 ( 0.143)    Loss 6.2642e+00 (7.5265e+00)    Acc@1  25.00 ( 19.52)   Acc@5  70.00 ( 76.94)
Epoch: [0][ 40/221]     Time  0.046 ( 0.161)    Data  0.010 ( 0.130)    Loss 1.1596e+01 (7.2846e+00)    Acc@1  20.00 ( 20.73)   Acc@5  25.00 ( 74.02)
Epoch: [0][ 50/221]     Time  0.028 ( 0.166)    Data  0.000 ( 0.135)    Loss 5.0911e+00 (6.9655e+00)    Acc@1  35.00 ( 20.88)   Acc@5  75.00 ( 74.22)
Epoch: [0][ 60/221]     Time  0.046 ( 0.159)    Data  0.002 ( 0.128)    Loss 4.8199e+00 (6.6354e+00)    Acc@1  25.00 ( 21.39)   Acc@5  70.00 ( 74.43)
Epoch: [0][ 70/221]     Time  0.039 ( 0.161)    Data  0.000 ( 0.131)    Loss 3.6673e+00 (6.2894e+00)    Acc@1  25.00 ( 22.39)   Acc@5  80.00 ( 74.58)
Epoch: [0][ 80/221]     Time  0.047 ( 0.155)    Data  0.003 ( 0.125)    Loss 1.8776e+00 (5.9031e+00)    Acc@1  20.00 ( 22.47)   Acc@5 100.00 ( 76.98)
Epoch: [0][ 90/221]     Time  0.046 ( 0.156)    Data  0.000 ( 0.125)    Loss 1.4587e+00 (5.4372e+00)    Acc@1  35.00 ( 22.86)   Acc@5 100.00 ( 79.51)
Epoch: [0][100/221]     Time  0.042 ( 0.151)    Data  0.015 ( 0.121)    Loss 1.3594e+00 (5.0451e+00)    Acc@1  25.00 ( 23.12)   Acc@5 100.00 ( 81.53)
Epoch: [0][110/221]     Time  0.034 ( 0.153)    Data  0.000 ( 0.123)    Loss 1.4235e+00 (4.7189e+00)    Acc@1  30.00 ( 23.29)   Acc@5 100.00 ( 83.20)
Epoch: [0][120/221]     Time  0.050 ( 0.152)    Data  0.005 ( 0.122)    Loss 1.4973e+00 (4.4490e+00)    Acc@1  25.00 ( 23.72)   Acc@5 100.00 ( 84.59)
Epoch: [0][130/221]     Time  0.035 ( 0.152)    Data  0.005 ( 0.122)    Loss 1.4152e+00 (4.2217e+00)    Acc@1  20.00 ( 23.97)   Acc@5 100.00 ( 85.76)
Epoch: [0][140/221]     Time  0.036 ( 0.150)    Data  0.002 ( 0.120)    Loss 1.4597e+00 (4.0232e+00)    Acc@1  15.00 ( 24.04)   Acc@5 100.00 ( 86.77)
Epoch: [0][150/221]     Time  0.043 ( 0.150)    Data  0.000 ( 0.120)    Loss 1.4626e+00 (3.8516e+00)    Acc@1  15.00 ( 23.77)   Acc@5 100.00 ( 87.65)
Epoch: [0][160/221]     Time  0.035 ( 0.150)    Data  0.000 ( 0.120)    Loss 1.4795e+00 (3.7016e+00)    Acc@1  20.00 ( 23.66)   Acc@5 100.00 ( 88.42)
Epoch: [0][170/221]     Time  0.049 ( 0.150)    Data  0.000 ( 0.120)    Loss 1.4996e+00 (3.5724e+00)    Acc@1  15.00 ( 23.80)   Acc@5 100.00 ( 89.09)
Epoch: [0][180/221]     Time  0.050 ( 0.149)    Data  0.002 ( 0.119)    Loss 1.4171e+00 (3.4577e+00)    Acc@1  30.00 ( 24.14)   Acc@5 100.00 ( 89.70)
Epoch: [0][190/221]     Time  0.027 ( 0.150)    Data  0.000 ( 0.120)    Loss 1.5100e+00 (3.3522e+00)    Acc@1  20.00 ( 24.40)   Acc@5 100.00 ( 90.24)
Epoch: [0][200/221]     Time  0.050 ( 0.148)    Data  0.008 ( 0.118)    Loss 1.4543e+00 (3.2566e+00)    Acc@1  30.00 ( 24.23)   Acc@5 100.00 ( 90.72)
Epoch: [0][210/221]     Time  0.047 ( 0.148)    Data  0.000 ( 0.118)    Loss 1.3781e+00 (3.1688e+00)    Acc@1  30.00 ( 24.19)   Acc@5 100.00 ( 91.16)
Epoch: [0][220/221]     Time  0.096 ( 0.146)    Data  0.000 ( 0.116)    Loss 1.4402e+00 (3.0951e+00)    Acc@1  40.00 ( 24.18)   Acc@5 100.00 ( 91.53)
Test: [ 0/40]   Time  1.187 ( 1.187)    Loss 1.3724e+00 (1.3724e+00)    Acc@1  20.00 ( 20.00)   Acc@5 100.00 (100.00)
Test: [10/40]   Time  0.009 ( 0.238)    Loss 1.1259e+00 (1.3368e+00)    Acc@1 100.00 ( 25.91)   Acc@5 100.00 (100.00)
Test: [20/40]   Time  0.246 ( 0.200)    Loss 1.6382e+00 (1.2612e+00)    Acc@1   0.00 ( 56.43)   Acc@5 100.00 (100.00)
Test: [30/40]   Time  0.006 ( 0.190)    Loss 1.4159e+00 (1.3745e+00)    Acc@1   0.00 ( 38.23)   Acc@5 100.00 (100.00)
 * Acc@1 29.625 Acc@5 100.000
==48297== Profiling application: python /home/as13594/examples/imagenet/main.py -a alexnet -b 20 --epochs 1 --lr 0.01 /home/as13594/
==48297== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   23.74%  1.17533s     14630  80.337us  1.3110us  831.77us  _ZN2at6native29vectorized_elementwise_kernelILi4EZZZNS0_15add_kernel_cudaERNS
lvE_clEvENKUlvE2_clEvEUlffE_NS_6detail5ArrayIPcLi3EEEEEviT0_T1_
                    8.52%  421.93ms      1338  315.35us  64.543us  529.15us  maxwell_sgemm_128x64_nt
                    7.22%  357.60ms      1012  353.36us  122.37us  525.50us  maxwell_sgemm_128x64_nn
                    6.76%  334.74ms       663  504.88us  52.320us  805.44us  void at::native::_GLOBAL__N__63_tmpxft_000019f5_00000000_10_DilatedMaxPool2d_
ol_backward_nchw<float, float>(int, float const *, long const *, int, int, int, int, int, int, int, int, int, int, int, int, int, int, at::native::_GLOBAL
_DilatedMaxPool2d_compute_75_cpp1_ii_db999de0::max_pool_backward_nchw<float, float>*)
                    6.22%  308.05ms       542  568.37us  1.1200us  17.017ms  [CUDA memcpy HtoD]
                    5.33%  263.65ms       406  649.39us  1.2480us  87.996ms  [CUDA memcpy DtoH]
                    4.18%  206.85ms       783  264.18us  127.84us  496.25us  sgemm_32x32x32_NT_vec
                    4.15%  205.56ms      4564  45.039us  1.3110us  572.06us  _ZN2at6native29vectorized_elementwise_kernelILi4EZNS0_23gpu_kernel_with_scala
ensorIteratorEENKUlvE_clEvENKUlvE2_clEvEUlffE_EEvS4_RKT_EUlfE0_NS_6detail5ArrayIPcLi2EEEEEviT0_T1_
                    2.96%  146.61ms       663  221.13us  60.480us  431.20us  sgemm_32x32x32_NN_vec
                    2.55%  126.17ms       263  479.74us  151.20us  545.05us  maxwell_scudnn_128x64_relu_large_nn
                    2.48%  122.92ms       663  185.40us  43.680us  377.57us  sgemm_128x128x8_TN_vec
```

As can be observed from the screenshot, we have used nvprof – a profiling tool to collect data about the GPU activity.

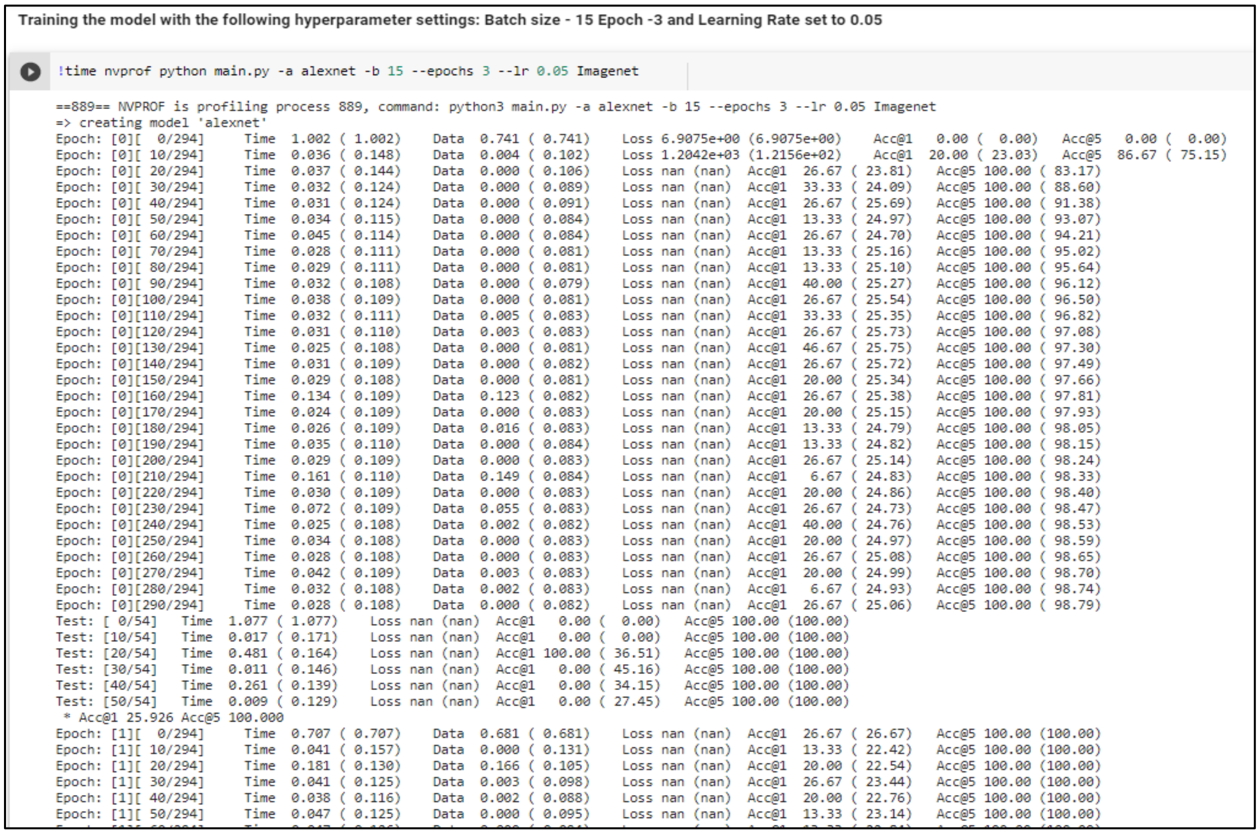


```
real    0m54.131s
user    0m42.959s
sys     0m8.413s
```

When we compared the runtime of the same job on the BareMetal and Cloud Environments, we noticed that the job ran much faster on the bare-metal system(as the time-elapsed was relatively lesser) than the cloud environment. The relative GPU time activity was the same. The throughput or the number of instructions processed per second is also relatively greater in the BareMetal environment.

Example Run 2:

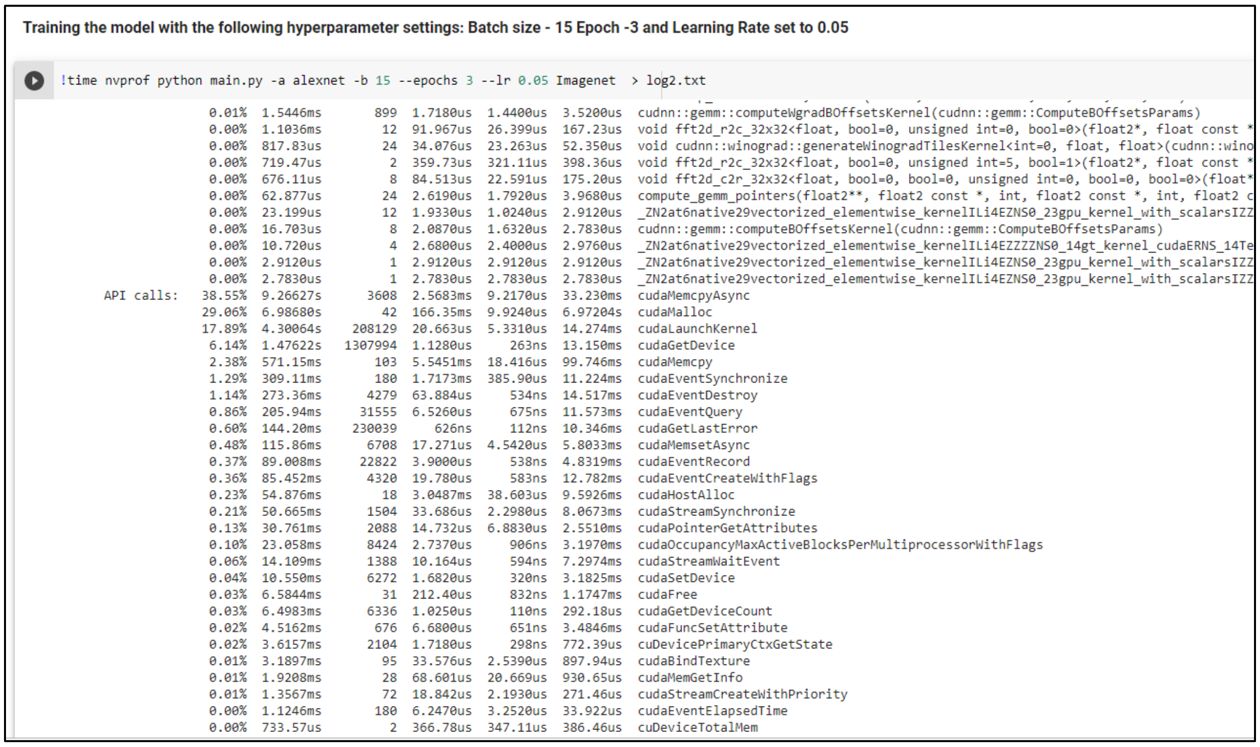
- **Running on Cloud:** For the second iteration, we train the model with the following hyperparameter settings: Batch size - 15 Epoch -3 and the Learning Rate has been set to 0.05. It takes approximately 294 iterations and processes images in batches of 15 and passes over the entire dataset three times. We see an accuracy of about 25% at the end of the first epoch.



Training the model with the following hyperparameter settings: Batch size - 15 Epoch -3 and Learning Rate set to 0.05

```
!time nvprof python main.py -a alexnet -b 15 --epochs 3 --lr 0.05 Imagenet
```

```
==889== NVPROF is profiling process 889, command: python3 main.py -a alexnet -b 15 --epochs 3 --lr 0.05 Imagenet
=> creating model 'alexnet'
Epoch: [0][  0/294]     Time  1.002 ( 1.002)    Data  0.741 ( 0.741)    Loss 6.9075e+00 (6.9075e+00)    Acc@1   0.00 (  0.00)   Acc@5   0.00 (  0.00)
Epoch: [0][ 10/294]     Time  0.036 ( 0.148)    Data  0.004 ( 0.102)    Loss 1.2042e+03 (1.2156e+02)    Acc@1  20.00 ( 23.03)   Acc@5  86.67 ( 75.15)
Epoch: [0][ 20/294]     Time  0.037 ( 0.144)    Data  0.000 ( 0.106)    Loss nan (nan)  Acc@1  26.67 ( 23.81)   Acc@5 100.00 ( 83.17)
Epoch: [0][ 30/294]     Time  0.032 ( 0.124)    Data  0.000 ( 0.089)    Loss nan (nan)  Acc@1  33.33 ( 24.09)   Acc@5 100.00 ( 88.60)
Epoch: [0][ 40/294]     Time  0.031 ( 0.124)    Data  0.000 ( 0.091)    Loss nan (nan)  Acc@1  26.67 ( 25.69)   Acc@5 100.00 ( 91.38)
Epoch: [0][ 50/294]     Time  0.034 ( 0.115)    Data  0.000 ( 0.084)    Loss nan (nan)  Acc@1  13.33 ( 24.97)   Acc@5 100.00 ( 93.07)
Epoch: [0][ 60/294]     Time  0.045 ( 0.114)    Data  0.000 ( 0.084)    Loss nan (nan)  Acc@1  26.67 ( 24.70)   Acc@5 100.00 ( 94.21)
Epoch: [0][ 70/294]     Time  0.028 ( 0.111)    Data  0.000 ( 0.081)    Loss nan (nan)  Acc@1  13.33 ( 25.16)   Acc@5 100.00 ( 95.02)
Epoch: [0][ 80/294]     Time  0.029 ( 0.111)    Data  0.000 ( 0.081)    Loss nan (nan)  Acc@1  13.33 ( 25.10)   Acc@5 100.00 ( 95.64)
Epoch: [0][ 90/294]     Time  0.032 ( 0.108)    Data  0.000 ( 0.079)    Loss nan (nan)  Acc@1  40.00 ( 25.27)   Acc@5 100.00 ( 96.12)
Epoch: [0][100/294]     Time  0.038 ( 0.109)    Data  0.000 ( 0.081)    Loss nan (nan)  Acc@1  26.67 ( 25.54)   Acc@5 100.00 ( 96.50)
Epoch: [0][110/294]     Time  0.032 ( 0.111)    Data  0.005 ( 0.083)    Loss nan (nan)  Acc@1  33.33 ( 25.35)   Acc@5 100.00 ( 96.82)
Epoch: [0][120/294]     Time  0.031 ( 0.110)    Data  0.003 ( 0.083)    Loss nan (nan)  Acc@1  26.67 ( 25.73)   Acc@5 100.00 ( 97.08)
Epoch: [0][130/294]     Time  0.025 ( 0.108)    Data  0.000 ( 0.081)    Loss nan (nan)  Acc@1  46.67 ( 25.75)   Acc@5 100.00 ( 97.30)
Epoch: [0][140/294]     Time  0.031 ( 0.109)    Data  0.000 ( 0.082)    Loss nan (nan)  Acc@1  26.67 ( 25.72)   Acc@5 100.00 ( 97.49)
Epoch: [0][150/294]     Time  0.029 ( 0.108)    Data  0.000 ( 0.081)    Loss nan (nan)  Acc@1  20.00 ( 25.34)   Acc@5 100.00 ( 97.66)
Epoch: [0][160/294]     Time  0.134 ( 0.109)    Data  0.123 ( 0.082)    Loss nan (nan)  Acc@1  26.67 ( 25.38)   Acc@5 100.00 ( 97.81)
Epoch: [0][170/294]     Time  0.024 ( 0.109)    Data  0.000 ( 0.083)    Loss nan (nan)  Acc@1  20.00 ( 25.15)   Acc@5 100.00 ( 97.93)
Epoch: [0][180/294]     Time  0.026 ( 0.109)    Data  0.016 ( 0.083)    Loss nan (nan)  Acc@1  13.33 ( 24.79)   Acc@5 100.00 ( 98.05)
Epoch: [0][190/294]     Time  0.035 ( 0.110)    Data  0.000 ( 0.084)    Loss nan (nan)  Acc@1  13.33 ( 24.82)   Acc@5 100.00 ( 98.15)
Epoch: [0][200/294]     Time  0.029 ( 0.109)    Data  0.000 ( 0.083)    Loss nan (nan)  Acc@1  26.67 ( 25.14)   Acc@5 100.00 ( 98.24)
Epoch: [0][210/294]     Time  0.161 ( 0.110)    Data  0.149 ( 0.084)    Loss nan (nan)  Acc@1   6.67 ( 24.83)   Acc@5 100.00 ( 98.33)
Epoch: [0][220/294]     Time  0.030 ( 0.109)    Data  0.000 ( 0.083)    Loss nan (nan)  Acc@1  20.00 ( 24.86)   Acc@5 100.00 ( 98.40)
Epoch: [0][230/294]     Time  0.072 ( 0.109)    Data  0.055 ( 0.083)    Loss nan (nan)  Acc@1  26.67 ( 24.73)   Acc@5 100.00 ( 98.47)
Epoch: [0][240/294]     Time  0.025 ( 0.108)    Data  0.002 ( 0.082)    Loss nan (nan)  Acc@1  40.00 ( 24.76)   Acc@5 100.00 ( 98.53)
Epoch: [0][250/294]     Time  0.034 ( 0.108)    Data  0.000 ( 0.083)    Loss nan (nan)  Acc@1  20.00 ( 24.97)   Acc@5 100.00 ( 98.59)
Epoch: [0][260/294]     Time  0.028 ( 0.108)    Data  0.000 ( 0.083)    Loss nan (nan)  Acc@1  26.67 ( 25.08)   Acc@5 100.00 ( 98.65)
Epoch: [0][270/294]     Time  0.042 ( 0.109)    Data  0.003 ( 0.083)    Loss nan (nan)  Acc@1  20.00 ( 24.99)   Acc@5 100.00 ( 98.70)
Epoch: [0][280/294]     Time  0.032 ( 0.108)    Data  0.002 ( 0.083)    Loss nan (nan)  Acc@1   6.67 ( 24.93)   Acc@5 100.00 ( 98.74)
Epoch: [0][290/294]     Time  0.028 ( 0.108)    Data  0.000 ( 0.082)    Loss nan (nan)  Acc@1  26.67 ( 25.06)   Acc@5 100.00 ( 98.79)
Test: [ 0/54]   Time  1.077 ( 1.077)    Loss nan (nan)  Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [10/54]   Time  0.017 ( 0.171)    Loss nan (nan)  Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [20/54]   Time  0.481 ( 0.164)    Loss nan (nan)  Acc@1 100.00 ( 36.51)   Acc@5 100.00 (100.00)
Test: [30/54]   Time  0.011 ( 0.146)    Loss nan (nan)  Acc@1   0.00 ( 45.16)   Acc@5 100.00 (100.00)
Test: [40/54]   Time  0.261 ( 0.139)    Loss nan (nan)  Acc@1   0.00 ( 34.15)   Acc@5 100.00 (100.00)
Test: [50/54]   Time  0.009 ( 0.129)    Loss nan (nan)  Acc@1   0.00 ( 27.45)   Acc@5 100.00 (100.00)
 * Acc@1 25.926 Acc@5 100.000
Epoch: [1][  0/294]     Time  0.707 ( 0.707)    Data  0.681 ( 0.681)    Loss nan (nan)  Acc@1  26.67 ( 26.67)   Acc@5 100.00 (100.00)
Epoch: [1][ 10/294]     Time  0.041 ( 0.157)    Data  0.000 ( 0.131)    Loss nan (nan)  Acc@1  13.33 ( 22.42)   Acc@5 100.00 (100.00)
Epoch: [1][ 20/294]     Time  0.181 ( 0.130)    Data  0.166 ( 0.105)    Loss nan (nan)  Acc@1  20.00 ( 22.54)   Acc@5 100.00 (100.00)
Epoch: [1][ 30/294]     Time  0.041 ( 0.125)    Data  0.003 ( 0.098)    Loss nan (nan)  Acc@1  26.67 ( 23.44)   Acc@5 100.00 (100.00)
Epoch: [1][ 40/294]     Time  0.038 ( 0.116)    Data  0.002 ( 0.088)    Loss nan (nan)  Acc@1  20.00 ( 22.76)   Acc@5 100.00 (100.00)
Epoch: [1][ 50/294]     Time  0.047 ( 0.125)    Data  0.000 ( 0.095)    Loss nan (nan)  Acc@1  13.33 ( 23.14)   Acc@5 100.00 (100.00)
```

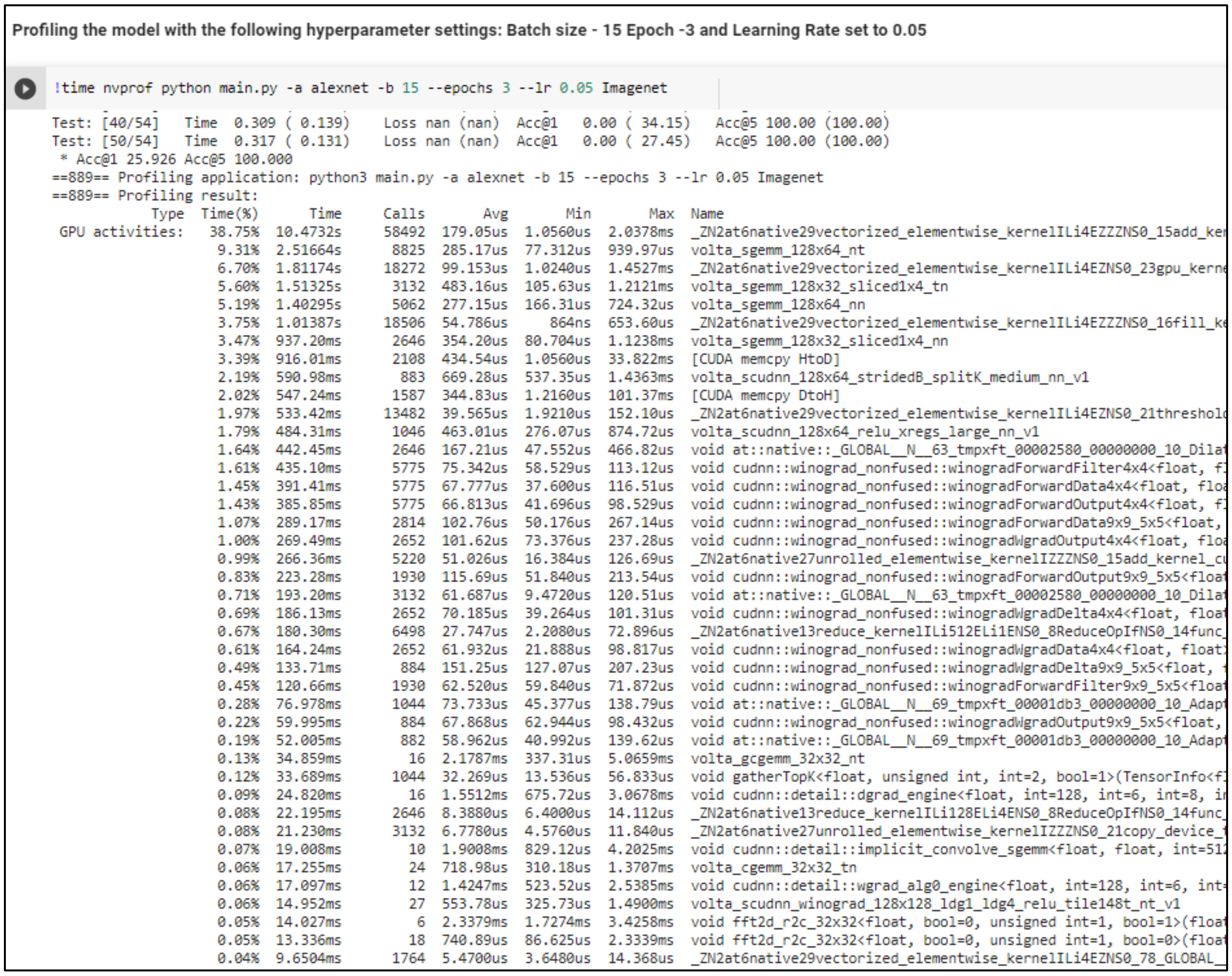


Training the model with the following hyperparameter settings: Batch size - 15 Epoch -3 and Learning Rate set to 0.05

```
!time nvprof python main.py -a alexnet -b 15 --epochs 3 --lr 0.05 Imagenet  > log2.txt
```

```
                    0.01%  1.5446ms       899  1.7180us  1.4400us  3.5200us  cudnn::gemm::computeWgradBOffsetsKernel(cudnn::gemm::ComputeBOffsetsParams)
                    0.00%  1.1036ms        12  91.967us  26.399us  167.23us  void fft2d_r2c_32x32<float, bool=0, unsigned int=0, bool=0>(float2*, float const *
                    0.00%  817.83us        24  34.076us  23.263us  52.350us  void cudnn::winograd::generateWinogradTilesKernel<int=0, float, float>(cudnn::wino
                    0.00%  719.47us         2  359.73us  321.11us  398.36us  void fft2d_r2c_32x32<float, bool=0, unsigned int=5, bool=1>(float2*, float const *
                    0.00%  676.11us         8  84.513us  22.591us  175.20us  void fft2d_c2r_32x32<float, bool=0, bool=0, unsigned int=0, bool=0, bool=0>(float*
                    0.00%  62.877us        24  2.6190us  1.7920us  3.9680us  compute_gemm_pointers(float2**, float2 const *, int, float2 const *, int, float2 c
                    0.00%  23.199us        12  1.9330us  1.0240us  2.9120us  _ZN2at6native29vectorized_elementwise_kernelILi4EZNS0_23gpu_kernel_with_scalarsIZZ
                    0.00%  16.703us         8  2.0870us  1.6320us  2.7830us  cudnn::gemm::computeBOffsetsKernel(cudnn::gemm::ComputeBOffsetsParams)
                    0.00%  10.720us         4  2.6800us  2.4000us  2.9760us  _ZN2at6native29vectorized_elementwise_kernelILi4EZZZNS0_14gt_kernel_cudaERNS_14Te
                    0.00%  2.9120us         1  2.9120us  2.9120us  2.9120us  _ZN2at6native29vectorized_elementwise_kernelILi4EZNS0_23gpu_kernel_with_scalarsIZZ
                    0.00%  2.7830us         1  2.7830us  2.7830us  2.7830us  _ZN2at6native29vectorized_elementwise_kernelILi4EZNS0_23gpu_kernel_with_scalarsIZZ
      API calls:   38.55%  9.26627s      3608  2.5683ms  9.2170us  33.230ms  cudaMemcpyAsync
                   29.06%  6.98680s        42  166.35ms  9.9240us  6.97204s  cudaMalloc
                   17.89%  4.30064s    208129  20.663us  5.3310us  14.274ms  cudaLaunchKernel
                    6.14%  1.47622s   1307994  1.1280us     263ns  13.150ms  cudaGetDevice
                    2.38%  571.15ms       103  5.5451ms  18.416us  99.746ms  cudaMemcpy
                    1.29%  309.11ms       180  1.7173ms  385.90us  11.224ms  cudaEventSynchronize
                    1.14%  273.36ms      4279  63.884us     534ns  14.517ms  cudaEventDestroy
                    0.86%  205.94ms     31555  6.5260us     675ns  11.573ms  cudaEventQuery
                    0.60%  144.20ms    230039     626ns     112ns  10.346ms  cudaGetLastError
                    0.48%  115.86ms      6708  17.271us  4.5420us  5.8033ms  cudaMemsetAsync
                    0.37%  89.008ms     22822  3.9000us     538ns  4.8319ms  cudaEventRecord
                    0.36%  85.452ms      4320  19.780us     583ns  12.782ms  cudaEventCreateWithFlags
                    0.23%  54.876ms        18  3.0487ms  38.603us  9.5926ms  cudaHostAlloc
                    0.21%  50.665ms      1504  33.686us  2.2980us  8.0673ms  cudaStreamSynchronize
                    0.13%  30.761ms      2088  14.732us  6.8830us  2.5510ms  cudaPointerGetAttributes
                    0.10%  23.058ms      8424  2.7370us     906ns  3.1970ms  cudaOccupancyMaxActiveBlocksPerMultiprocessorWithFlags
                    0.06%  14.109ms      1388  10.164us     594ns  7.2974ms  cudaStreamWaitEvent
                    0.04%  10.550ms      6272  1.6820us     320ns  3.1825ms  cudaSetDevice
                    0.03%  6.5844ms        31  212.40us     832ns  1.1747ms  cudaFree
                    0.03%  6.4983ms      6336  1.0250us     110ns  292.18us  cudaGetDeviceCount
                    0.02%  4.5162ms       676  6.6800us     651ns  3.4846ms  cudaFuncSetAttribute
                    0.02%  3.6157ms      2104  1.7180us     298ns  772.39us  cuDevicePrimaryCtxGetState
                    0.01%  3.1897ms        95  33.576us  2.5390us  897.94us  cudaBindTexture
                    0.01%  1.9208ms        28  68.601us  20.669us  930.65us  cudaMemGetInfo
                    0.01%  1.3567ms        72  18.842us  2.1930us  271.46us  cudaStreamCreateWithPriority
                    0.00%  1.1246ms       180  6.2470us  3.2520us  33.922us  cudaEventElapsedTime
                    0.00%  733.57us         2  366.78us  347.11us  386.46us  cuDeviceTotalMem
```

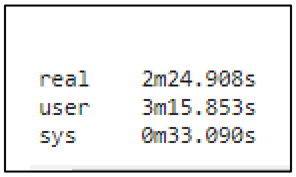
Once all three epochs' have been processed and the training is complete. The overall time taken to process has comparatively increased for this run as compared to the previous example -could be expected from the increase in epoch and decrease in batch size. The learning rate was increased and hence the step size has also increased. One can also note that the accuracy didn't change much.



Profiling the model with the following hyperparameter settings: Batch size - 15 Epoch -3 and Learning Rate set to 0.05

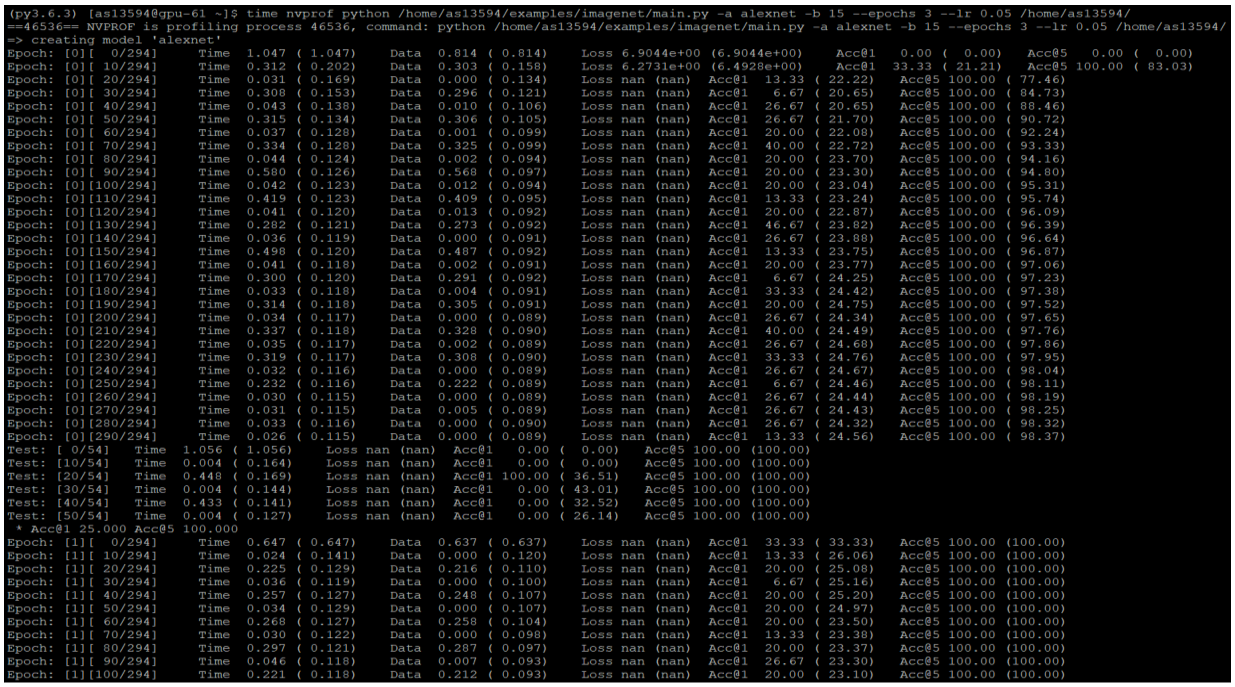
```
!time nvprof python main.py -a alexnet -b 15 --epochs 3 --lr 0.05 Imagenet
```

```
Test: [40/54]   Time  0.309 ( 0.139)    Loss nan (nan)  Acc@1   0.00 ( 34.15)   Acc@5 100.00 (100.00)
Test: [50/54]   Time  0.317 ( 0.131)    Loss nan (nan)  Acc@1   0.00 ( 27.45)   Acc@5 100.00 (100.00)
 * Acc@1 25.926 Acc@5 100.000
==889== Profiling application: python3 main.py -a alexnet -b 15 --epochs 3 --lr 0.05 Imagenet
==889== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   38.75%  10.4732s     58492  179.05us  1.0560us  2.0378ms  _ZN2at6native29vectorized_elementwise_kernelILi4EZZZNS0_15add_ker
                    9.31%  2.51664s      8825  285.17us  77.312us  939.97us  volta_sgemm_128x64_nt
                    6.70%  1.81174s     18272  99.153us  1.0240us  1.4527ms  _ZN2at6native29vectorized_elementwise_kernelILi4EZNS0_23gpu_kerne
                    5.60%  1.51325s      3132  483.16us  105.63us  1.2121ms  volta_sgemm_128x32_sliced1x4_tn
                    5.19%  1.40295s      5062  277.15us  166.31us  724.32us  volta_sgemm_128x64_nn
                    3.75%  1.01387s     18506  54.786us     864ns  653.60us  _ZN2at6native29vectorized_elementwise_kernelILi4EZZZNS0_16fill_ke
                    3.47%  937.20ms      2646  354.20us  80.704us  1.1238ms  volta_sgemm_128x32_sliced1x4_nn
                    3.39%  916.01ms      2108  434.54us  1.0560us  33.822ms  [CUDA memcpy HtoD]
                    2.19%  590.98ms       883  669.28us  537.35us  1.4363ms  volta_scudnn_128x64_stridedB_splitK_medium_nn_v1
                    2.02%  547.24ms      1587  344.83us  1.2160us  101.37ms  [CUDA memcpy DtoH]
                    1.97%  533.42ms     13482  39.565us  1.9210us  152.10us  _ZN2at6native29vectorized_elementwise_kernelILi4EZNS0_21threshold
                    1.79%  484.31ms      1046  463.01us  276.07us  874.72us  volta_scudnn_128x64_relu_xregs_large_nn_v1
                    1.64%  442.45ms      2646  167.21us  47.552us  466.82us  void at::native::_GLOBAL__N__63_tmpxft_00002580_00000000_10_Dila
                    1.61%  435.10ms      5775  75.342us  58.529us  113.12us  void cudnn::winograd_nonfused::winogradForwardFilter4x4<float, f
                    1.45%  391.41ms      5775  67.777us  37.600us  116.51us  void cudnn::winograd_nonfused::winogradForwardData4x4<float, flo
                    1.43%  385.85ms      5775  66.813us  41.696us  98.529us  void cudnn::winograd_nonfused::winogradForwardOutput4x4<float, f
                    1.07%  289.17ms      2814  102.76us  50.176us  267.14us  void cudnn::winograd_nonfused::winogradForwardData9x9_5x5<float,
                    1.00%  269.49ms      2652  101.62us  73.376us  237.28us  void cudnn::winograd_nonfused::winogradWgradOutput4x4<float, flo
                    0.99%  266.36ms      5220  51.026us  16.384us  126.69us  _ZN2at6native27unrolled_elementwise_kernelIZZZNS0_15add_kernel_cu
                    0.83%  223.28ms      1930  115.69us  51.840us  213.54us  void cudnn::winograd_nonfused::winogradForwardOutput9x9_5x5<float
                    0.71%  193.20ms      3132  61.687us  9.4720us  120.51us  void at::native::_GLOBAL__N__63_tmpxft_00002580_00000000_10_Dila
                    0.69%  186.13ms      2652  70.185us  39.264us  101.31us  void cudnn::winograd_nonfused::winogradWgradDelta4x4<float, float
                    0.67%  180.30ms      6498  27.747us  2.2080us  72.896us  _ZN2at6native13reduce_kernelILi512ELi1ENS0_8ReduceOpIfNS0_14func_
                    0.61%  164.24ms      2652  61.932us  21.888us  98.817us  void cudnn::winograd_nonfused::winogradWgradData4x4<float, float>
                    0.49%  133.71ms       884  151.25us  127.07us  207.23us  void cudnn::winograd_nonfused::winogradWgradDelta9x9_5x5<float,
                    0.45%  120.66ms      1930  62.520us  59.840us  71.872us  void cudnn::winograd_nonfused::winogradForwardFilter9x9_5x5<float
                    0.28%  76.978ms      1044  73.733us  45.377us  138.79us  void at::native::_GLOBAL__N__69_tmpxft_00001db3_00000000_10_Adap
                    0.22%  59.995ms       884  67.868us  62.944us  98.432us  void cudnn::winograd_nonfused::winogradWgradOutput9x9_5x5<float,
                    0.19%  52.005ms       882  58.962us  40.992us  139.62us  void at::native::_GLOBAL__N__69_tmpxft_00001db3_00000000_10_Adap
                    0.13%  34.859ms        16  2.1787ms  337.31us  5.0659ms  volta_gcgemm_32x32_nt
                    0.12%  33.689ms      1044  32.269us  13.536us  56.833us  void gatherTopK<float, unsigned int, int=2, bool=1>(TensorInfo<f
                    0.09%  24.820ms        16  1.5512ms  675.72us  3.0678ms  void cudnn::detail::dgrad_engine<float, int=128, int=6, int=8, i
                    0.08%  22.195ms      2646  8.3880us  6.4000us  14.112us  _ZN2at6native13reduce_kernelILi128ELi4ENS0_8ReduceOpIfNS0_14func_
                    0.08%  21.230ms      3132  6.7780us  4.5760us  11.840us  _ZN2at6native27unrolled_elementwise_kernelIZZZNS0_21copy_device_
                    0.07%  19.008ms        10  1.9008ms  829.12us  4.2025ms  void cudnn::detail::implicit_convolve_sgemm<float, float, int=51
                    0.06%  17.255ms        24  718.98us  310.18us  1.3707ms  volta_cgemm_32x32_tn
                    0.06%  17.097ms        12  1.4247ms  523.52us  2.5385ms  void cudnn::detail::wgrad_alg0_engine<float, int=128, int=6, int
                    0.06%  14.952ms        27  553.78us  325.73us  1.4900ms  volta_scudnn_winograd_128x128_ldg1_ldg4_relu_tile148t_nt_v1
                    0.05%  14.027ms         6  2.3379ms  1.7274ms  3.4258ms  void fft2d_r2c_32x32<float, bool=0, unsigned int=1, bool=1>(floa
                    0.05%  13.336ms        18  740.89us  86.625us  2.3339ms  void fft2d_r2c_32x32<float, bool=0, unsigned int=1, bool=0>(floa
                    0.04%  9.6504ms      1764  5.4700us  3.6480us  14.368us  _ZN2at6native29vectorized_elementwise_kernelILi4EZNS0_78_GLOBAL_
```



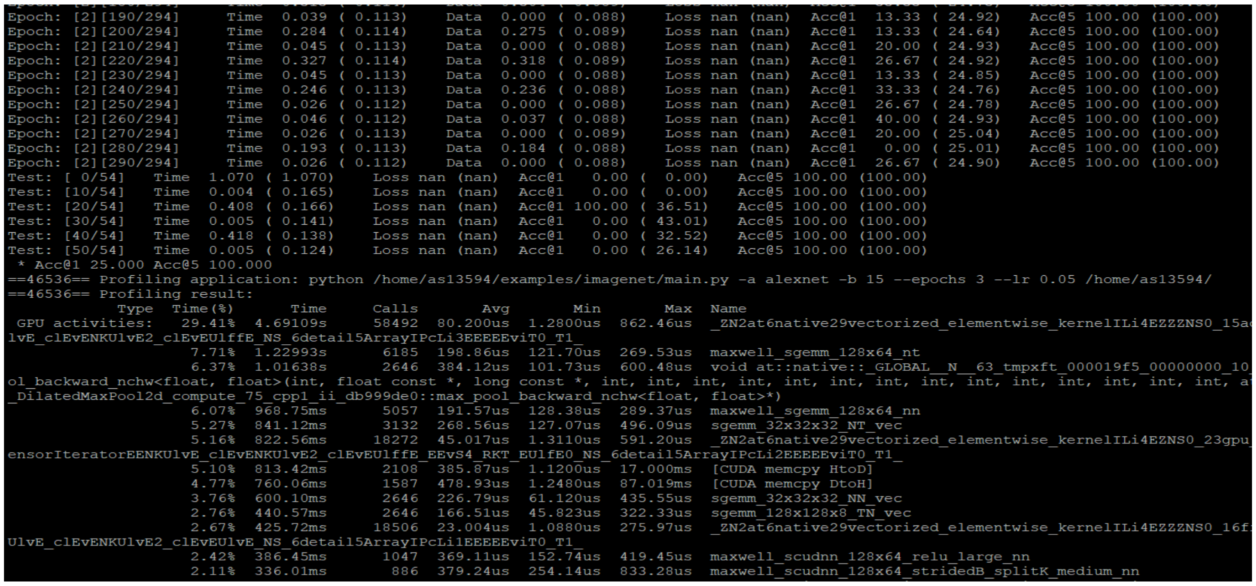
```
real    2m24.908s
user    3m15.853s
sys     0m33.090s
```

- **Running on BareMetal:** For the second iteration, we train the model with the same hyperparameter settings: Batch size - 15 Epoch -3 and the Learning Rate has been set to 0.05. It takes approximately 294 iterations and processes images in batches of 15 and passes over the entire dataset three times.

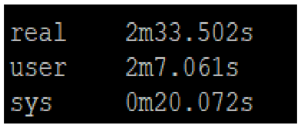


```
(py3.6.3) [as13594@gpu-61 ~]$ time nvprof python /home/as13594/examples/imagenet/main.py -a alexnet -b 15 --epochs 3 --lr 0.05 /home/as13594/
==46536== NVPROF is profiling process 46536, command: python /home/as13594/examples/imagenet/main.py -a alexnet -b 15 --epochs 3 --lr 0.05 /home/as13594/
=> creating model 'alexnet'
Epoch: [0][  0/294]     Time  1.047 ( 1.047)    Data  0.814 ( 0.814)    Loss 6.9044e+00 (6.9044e+00)    Acc@1   0.00 (  0.00)   Acc@5   0.00 (  0.00)
Epoch: [0][ 10/294]     Time  0.312 ( 0.202)    Data  0.303 ( 0.158)    Loss 6.2731e+00 (6.4928e+00)    Acc@1  33.33 ( 21.21)   Acc@5 100.00 ( 83.03)
Epoch: [0][ 20/294]     Time  0.031 ( 0.169)    Data  0.000 ( 0.134)    Loss nan (nan)  Acc@1  13.33 ( 22.22)   Acc@5 100.00 ( 77.46)
Epoch: [0][ 30/294]     Time  0.308 ( 0.153)    Data  0.296 ( 0.121)    Loss nan (nan)  Acc@1   6.67 ( 20.65)   Acc@5 100.00 ( 84.73)
Epoch: [0][ 40/294]     Time  0.043 ( 0.138)    Data  0.010 ( 0.106)    Loss nan (nan)  Acc@1  26.67 ( 20.65)   Acc@5 100.00 ( 88.46)
Epoch: [0][ 50/294]     Time  0.315 ( 0.134)    Data  0.306 ( 0.105)    Loss nan (nan)  Acc@1  26.67 ( 21.70)   Acc@5 100.00 ( 90.72)
Epoch: [0][ 60/294]     Time  0.037 ( 0.128)    Data  0.001 ( 0.099)    Loss nan (nan)  Acc@1  20.00 ( 22.08)   Acc@5 100.00 ( 92.24)
Epoch: [0][ 70/294]     Time  0.334 ( 0.128)    Data  0.325 ( 0.099)    Loss nan (nan)  Acc@1  40.00 ( 22.72)   Acc@5 100.00 ( 93.33)
Epoch: [0][ 80/294]     Time  0.044 ( 0.124)    Data  0.002 ( 0.094)    Loss nan (nan)  Acc@1  20.00 ( 23.70)   Acc@5 100.00 ( 94.16)
Epoch: [0][ 90/294]     Time  0.580 ( 0.126)    Data  0.568 ( 0.097)    Loss nan (nan)  Acc@1  20.00 ( 23.30)   Acc@5 100.00 ( 94.80)
Epoch: [0][100/294]     Time  0.042 ( 0.123)    Data  0.012 ( 0.094)    Loss nan (nan)  Acc@1  20.00 ( 23.04)   Acc@5 100.00 ( 95.31)
Epoch: [0][110/294]     Time  0.419 ( 0.123)    Data  0.409 ( 0.095)    Loss nan (nan)  Acc@1  13.33 ( 23.24)   Acc@5 100.00 ( 95.74)
Epoch: [0][120/294]     Time  0.041 ( 0.120)    Data  0.013 ( 0.092)    Loss nan (nan)  Acc@1  20.00 ( 22.87)   Acc@5 100.00 ( 96.09)
Epoch: [0][130/294]     Time  0.282 ( 0.121)    Data  0.273 ( 0.092)    Loss nan (nan)  Acc@1  46.67 ( 23.82)   Acc@5 100.00 ( 96.39)
Epoch: [0][140/294]     Time  0.036 ( 0.119)    Data  0.000 ( 0.091)    Loss nan (nan)  Acc@1  26.67 ( 23.88)   Acc@5 100.00 ( 96.64)
Epoch: [0][150/294]     Time  0.498 ( 0.120)    Data  0.487 ( 0.092)    Loss nan (nan)  Acc@1  13.33 ( 23.75)   Acc@5 100.00 ( 96.87)
Epoch: [0][160/294]     Time  0.041 ( 0.118)    Data  0.002 ( 0.091)    Loss nan (nan)  Acc@1  20.00 ( 23.77)   Acc@5 100.00 ( 97.06)
Epoch: [0][170/294]     Time  0.300 ( 0.120)    Data  0.291 ( 0.092)    Loss nan (nan)  Acc@1   6.67 ( 24.25)   Acc@5 100.00 ( 97.23)
Epoch: [0][180/294]     Time  0.033 ( 0.118)    Data  0.004 ( 0.091)    Loss nan (nan)  Acc@1  33.33 ( 24.42)   Acc@5 100.00 ( 97.38)
Epoch: [0][190/294]     Time  0.314 ( 0.118)    Data  0.305 ( 0.091)    Loss nan (nan)  Acc@1  20.00 ( 24.75)   Acc@5 100.00 ( 97.52)
Epoch: [0][200/294]     Time  0.034 ( 0.117)    Data  0.000 ( 0.089)    Loss nan (nan)  Acc@1  26.67 ( 24.34)   Acc@5 100.00 ( 97.65)
Epoch: [0][210/294]     Time  0.337 ( 0.118)    Data  0.328 ( 0.090)    Loss nan (nan)  Acc@1  40.00 ( 24.49)   Acc@5 100.00 ( 97.76)
Epoch: [0][220/294]     Time  0.035 ( 0.117)    Data  0.002 ( 0.089)    Loss nan (nan)  Acc@1  26.67 ( 24.68)   Acc@5 100.00 ( 97.86)
Epoch: [0][230/294]     Time  0.319 ( 0.117)    Data  0.308 ( 0.090)    Loss nan (nan)  Acc@1  33.33 ( 24.76)   Acc@5 100.00 ( 97.95)
Epoch: [0][240/294]     Time  0.032 ( 0.116)    Data  0.000 ( 0.089)    Loss nan (nan)  Acc@1  26.67 ( 24.67)   Acc@5 100.00 ( 98.04)
Epoch: [0][250/294]     Time  0.232 ( 0.116)    Data  0.222 ( 0.089)    Loss nan (nan)  Acc@1   6.67 ( 24.46)   Acc@5 100.00 ( 98.11)
Epoch: [0][260/294]     Time  0.030 ( 0.115)    Data  0.000 ( 0.089)    Loss nan (nan)  Acc@1  26.67 ( 24.44)   Acc@5 100.00 ( 98.19)
Epoch: [0][270/294]     Time  0.031 ( 0.115)    Data  0.005 ( 0.089)    Loss nan (nan)  Acc@1  26.67 ( 24.43)   Acc@5 100.00 ( 98.25)
Epoch: [0][280/294]     Time  0.033 ( 0.116)    Data  0.000 ( 0.090)    Loss nan (nan)  Acc@1  26.67 ( 24.32)   Acc@5 100.00 ( 98.32)
Epoch: [0][290/294]     Time  0.026 ( 0.115)    Data  0.000 ( 0.089)    Loss nan (nan)  Acc@1  13.33 ( 24.56)   Acc@5 100.00 ( 98.37)
Test: [ 0/54]   Time  1.056 ( 1.056)    Loss nan (nan)  Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [10/54]   Time  0.004 ( 0.164)    Loss nan (nan)  Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [20/54]   Time  0.448 ( 0.169)    Loss nan (nan)  Acc@1 100.00 ( 36.51)   Acc@5 100.00 (100.00)
Test: [30/54]   Time  0.004 ( 0.144)    Loss nan (nan)  Acc@1   0.00 ( 43.01)   Acc@5 100.00 (100.00)
Test: [40/54]   Time  0.433 ( 0.141)    Loss nan (nan)  Acc@1   0.00 ( 32.52)   Acc@5 100.00 (100.00)
Test: [50/54]   Time  0.004 ( 0.127)    Loss nan (nan)  Acc@1   0.00 ( 26.14)   Acc@5 100.00 (100.00)
 * Acc@1 25.000 Acc@5 100.000
Epoch: [1][  0/294]     Time  0.647 ( 0.647)    Data  0.637 ( 0.637)    Loss nan (nan)  Acc@1  33.33 ( 33.33)   Acc@5 100.00 (100.00)
Epoch: [1][ 10/294]     Time  0.024 ( 0.141)    Data  0.000 ( 0.120)    Loss nan (nan)  Acc@1  13.33 ( 26.06)   Acc@5 100.00 (100.00)
Epoch: [1][ 20/294]     Time  0.225 ( 0.129)    Data  0.216 ( 0.110)    Loss nan (nan)  Acc@1  20.00 ( 25.08)   Acc@5 100.00 (100.00)
Epoch: [1][ 30/294]     Time  0.036 ( 0.119)    Data  0.000 ( 0.100)    Loss nan (nan)  Acc@1   6.67 ( 25.16)   Acc@5 100.00 (100.00)
Epoch: [1][ 40/294]     Time  0.257 ( 0.127)    Data  0.248 ( 0.107)    Loss nan (nan)  Acc@1  20.00 ( 25.20)   Acc@5 100.00 (100.00)
Epoch: [1][ 50/294]     Time  0.034 ( 0.129)    Data  0.000 ( 0.107)    Loss nan (nan)  Acc@1  20.00 ( 24.97)   Acc@5 100.00 (100.00)
Epoch: [1][ 60/294]     Time  0.268 ( 0.127)    Data  0.258 ( 0.104)    Loss nan (nan)  Acc@1  20.00 ( 23.50)   Acc@5 100.00 (100.00)
Epoch: [1][ 70/294]     Time  0.030 ( 0.122)    Data  0.000 ( 0.098)    Loss nan (nan)  Acc@1  13.33 ( 23.38)   Acc@5 100.00 (100.00)
Epoch: [1][ 80/294]     Time  0.297 ( 0.121)    Data  0.287 ( 0.097)    Loss nan (nan)  Acc@1  20.00 ( 23.37)   Acc@5 100.00 (100.00)
Epoch: [1][ 90/294]     Time  0.046 ( 0.118)    Data  0.007 ( 0.093)    Loss nan (nan)  Acc@1  26.67 ( 23.30)   Acc@5 100.00 (100.00)
Epoch: [1][100/294]     Time  0.221 ( 0.118)    Data  0.212 ( 0.093)    Loss nan (nan)  Acc@1  20.00 ( 23.10)   Acc@5 100.00 (100.00)
```

As can be observed in the screenshot, the profiling information for the run has been generated at the end of the third epoch by nvprof.



```
Epoch: [2][190/294]     Time  0.039 ( 0.113)    Data  0.000 ( 0.088)    Loss nan (nan)  Acc@1  13.33 ( 24.92)   Acc@5 100.00 (100.00)
Epoch: [2][200/294]     Time  0.284 ( 0.114)    Data  0.275 ( 0.089)    Loss nan (nan)  Acc@1  13.33 ( 24.64)   Acc@5 100.00 (100.00)
Epoch: [2][210/294]     Time  0.045 ( 0.113)    Data  0.000 ( 0.088)    Loss nan (nan)  Acc@1  20.00 ( 24.93)   Acc@5 100.00 (100.00)
Epoch: [2][220/294]     Time  0.327 ( 0.114)    Data  0.318 ( 0.089)    Loss nan (nan)  Acc@1  26.67 ( 24.92)   Acc@5 100.00 (100.00)
Epoch: [2][230/294]     Time  0.045 ( 0.113)    Data  0.000 ( 0.088)    Loss nan (nan)  Acc@1  13.33 ( 24.85)   Acc@5 100.00 (100.00)
Epoch: [2][240/294]     Time  0.246 ( 0.113)    Data  0.236 ( 0.088)    Loss nan (nan)  Acc@1  33.33 ( 24.76)   Acc@5 100.00 (100.00)
Epoch: [2][250/294]     Time  0.026 ( 0.112)    Data  0.000 ( 0.088)    Loss nan (nan)  Acc@1  26.67 ( 24.78)   Acc@5 100.00 (100.00)
Epoch: [2][260/294]     Time  0.046 ( 0.112)    Data  0.037 ( 0.088)    Loss nan (nan)  Acc@1  40.00 ( 24.93)   Acc@5 100.00 (100.00)
Epoch: [2][270/294]     Time  0.026 ( 0.113)    Data  0.000 ( 0.089)    Loss nan (nan)  Acc@1  20.00 ( 25.04)   Acc@5 100.00 (100.00)
Epoch: [2][280/294]     Time  0.193 ( 0.113)    Data  0.184 ( 0.088)    Loss nan (nan)  Acc@1   0.00 ( 25.01)   Acc@5 100.00 (100.00)
Epoch: [2][290/294]     Time  0.026 ( 0.112)    Data  0.000 ( 0.088)    Loss nan (nan)  Acc@1  26.67 ( 24.90)   Acc@5 100.00 (100.00)
Test: [ 0/54]   Time  1.070 ( 1.070)    Loss nan (nan)  Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [10/54]   Time  0.004 ( 0.165)    Loss nan (nan)  Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [20/54]   Time  0.408 ( 0.166)    Loss nan (nan)  Acc@1 100.00 ( 36.51)   Acc@5 100.00 (100.00)
Test: [30/54]   Time  0.005 ( 0.141)    Loss nan (nan)  Acc@1   0.00 ( 43.01)   Acc@5 100.00 (100.00)
Test: [40/54]   Time  0.418 ( 0.138)    Loss nan (nan)  Acc@1   0.00 ( 32.52)   Acc@5 100.00 (100.00)
Test: [50/54]   Time  0.005 ( 0.124)    Loss nan (nan)  Acc@1   0.00 ( 26.14)   Acc@5 100.00 (100.00)
 * Acc@1 25.000 Acc@5 100.000
==46536== Profiling application: python /home/as13594/examples/imagenet/main.py -a alexnet -b 15 --epochs 3 --lr 0.05 /home/as13594/
==46536== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   29.41%  4.69109s     58492  80.200us  1.2800us  862.46us  _ZN2at6native29vectorized_elementwise_kernelILi4EZZZNS0_15a
lvE_clEvENKUlvE2_clEvEUlffE_NS_6detail5ArrayIPcLi3EEEEEviT0_T1_
                    7.71%  1.22993s      6185  198.86us  121.70us  269.53us  maxwell_sgemm_128x64_nt
                    6.37%  1.01638s      2646  384.12us  101.73us  600.48us  void at::native::_GLOBAL__N__63_tmpxft_000019f5_00000000_10
ol_backward_nchw<float, float>(int, float const *, long const *, int, int, int, int, int, int, int, int, int, int, int, int, int, int, a
_DilatedMaxPool2d_compute_75_cpp1_ii_db999de0::max_pool_backward_nchw<float, float>*)
                    6.07%  968.75ms      5057  191.57us  128.38us  289.37us  maxwell_sgemm_128x64_nn
                    5.27%  841.12ms      3132  268.56us  127.07us  496.09us  sgemm_32x32x32_NT_vec
                    5.16%  822.56ms     18272  45.017us  1.3110us  591.20us  _ZN2at6native29vectorized_elementwise_kernelILi4EZNS0_23gpu
ensorIteratorEENKUlvE_clEvENKUlvE2_clEvEUlffE_EEvS4_RKT_EUlfE0_NS_6detail5ArrayIPcLi2EEEEviT0_T1_
                    5.10%  813.42ms      2108  385.87us  1.1200us  17.000ms  [CUDA memcpy HtoD]
                    4.77%  760.06ms      1587  478.93us  1.2480us  87.019ms  [CUDA memcpy DtoH]
                    3.76%  600.10ms      2646  226.79us  61.120us  435.55us  sgemm_32x32x32_NN_vec
                    2.76%  440.57ms      2646  166.51us  45.823us  322.33us  sgemm_128x128x8_TN_vec
                    2.67%  425.72ms     18506  23.004us  1.0880us  275.97us  _ZN2at6native29vectorized_elementwise_kernelILi4EZZZNS0_16f
UlvE_clEvENKUlvE2_clEvEUlvE_NS_6detail5ArrayIPcLi1EEEEviT0_T1_
                    2.42%  386.45ms      1047  369.11us  152.74us  419.45us  maxwell_scudnn_128x64_relu_large_nn
                    2.11%  336.01ms       886  379.24us  254.14us  833.28us  maxwell_scudnn_128x64_stridedB_splitK_medium_nn
```

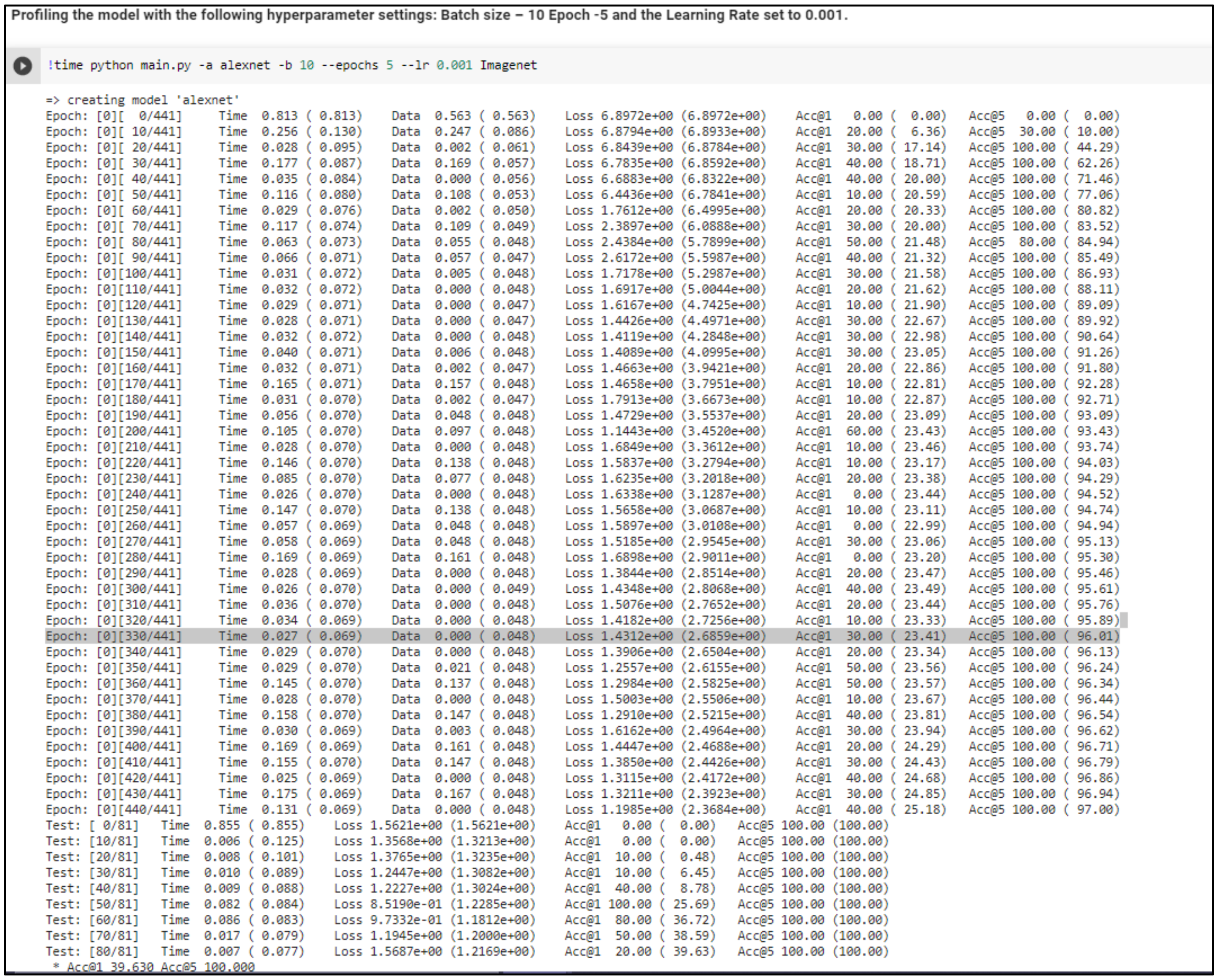


```
real    2m33.502s
user    2m7.061s
sys     0m20.072s
```

When we compare the runtime of the same job on the BareMetal and Cloud Environments, we noticed that the time elapsed in both cases was relatively same. The relative GPU time activity on Cloud was comparatively more.

Example Run 3:

Running on Cloud: We train the model with the following hyperparameter settings: Batch size – 10 Epoch -5 and the Learning Rate has been set to 0.001. It takes approximately 441 iterations in each epoch and processes images in batches of 10.

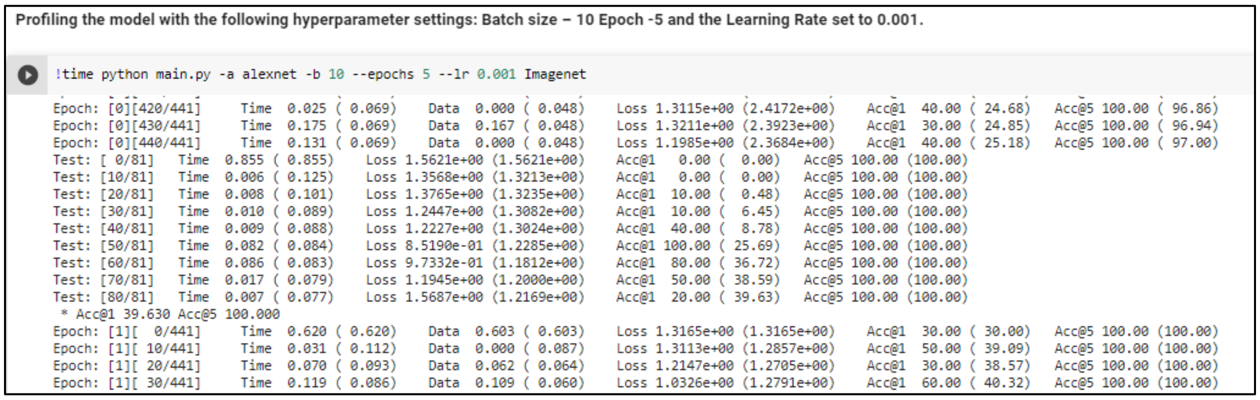


Profiling the model with the following hyperparameter settings: Batch size – 10 Epoch -5 and the Learning Rate set to 0.001.

```
!time python main.py -a alexnet -b 10 --epochs 5 --lr 0.001 Imagenet

=> creating model 'alexnet'
Epoch: [0][  0/441]    Time  0.813 ( 0.813)    Data  0.563 ( 0.563)    Loss 6.8972e+00 (6.8972e+00)    Acc@1   0.00 (  0.00)   Acc@5   0.00 (  0.00)
Epoch: [0][ 10/441]    Time  0.256 ( 0.130)    Data  0.247 ( 0.086)    Loss 6.8794e+00 (6.8933e+00)    Acc@1  20.00 (  6.36)   Acc@5  30.00 ( 10.00)
Epoch: [0][ 20/441]    Time  0.028 ( 0.095)    Data  0.002 ( 0.061)    Loss 6.8439e+00 (6.8784e+00)    Acc@1  30.00 ( 17.14)   Acc@5 100.00 ( 44.29)
Epoch: [0][ 30/441]    Time  0.177 ( 0.087)    Data  0.169 ( 0.057)    Loss 6.7835e+00 (6.8592e+00)    Acc@1  40.00 ( 18.71)   Acc@5 100.00 ( 62.26)
Epoch: [0][ 40/441]    Time  0.035 ( 0.084)    Data  0.000 ( 0.056)    Loss 6.6883e+00 (6.8322e+00)    Acc@1  40.00 ( 20.00)   Acc@5 100.00 ( 71.46)
Epoch: [0][ 50/441]    Time  0.116 ( 0.080)    Data  0.108 ( 0.053)    Loss 6.4436e+00 (6.7841e+00)    Acc@1  10.00 ( 20.59)   Acc@5 100.00 ( 77.06)
Epoch: [0][ 60/441]    Time  0.029 ( 0.076)    Data  0.002 ( 0.050)    Loss 1.7612e+00 (6.4995e+00)    Acc@1  20.00 ( 20.33)   Acc@5 100.00 ( 80.82)
Epoch: [0][ 70/441]    Time  0.117 ( 0.074)    Data  0.109 ( 0.049)    Loss 2.3897e+00 (6.0888e+00)    Acc@1  30.00 ( 20.00)   Acc@5 100.00 ( 83.52)
Epoch: [0][ 80/441]    Time  0.063 ( 0.073)    Data  0.055 ( 0.048)    Loss 2.4384e+00 (5.7899e+00)    Acc@1  50.00 ( 21.48)   Acc@5  80.00 ( 84.94)
Epoch: [0][ 90/441]    Time  0.066 ( 0.071)    Data  0.057 ( 0.047)    Loss 2.6172e+00 (5.5987e+00)    Acc@1  40.00 ( 21.32)   Acc@5 100.00 ( 85.49)
Epoch: [0][100/441]    Time  0.031 ( 0.072)    Data  0.005 ( 0.048)    Loss 1.7178e+00 (5.2987e+00)    Acc@1  30.00 ( 21.58)   Acc@5 100.00 ( 86.93)
Epoch: [0][110/441]    Time  0.032 ( 0.072)    Data  0.000 ( 0.048)    Loss 1.6917e+00 (5.0044e+00)    Acc@1  20.00 ( 21.62)   Acc@5 100.00 ( 88.11)
Epoch: [0][120/441]    Time  0.029 ( 0.071)    Data  0.000 ( 0.047)    Loss 1.6167e+00 (4.7425e+00)    Acc@1  10.00 ( 21.90)   Acc@5 100.00 ( 89.09)
Epoch: [0][130/441]    Time  0.028 ( 0.071)    Data  0.000 ( 0.047)    Loss 1.4426e+00 (4.4971e+00)    Acc@1  30.00 ( 22.67)   Acc@5 100.00 ( 89.92)
Epoch: [0][140/441]    Time  0.032 ( 0.072)    Data  0.000 ( 0.048)    Loss 1.4119e+00 (4.2848e+00)    Acc@1  30.00 ( 22.98)   Acc@5 100.00 ( 90.64)
Epoch: [0][150/441]    Time  0.040 ( 0.071)    Data  0.006 ( 0.048)    Loss 1.4089e+00 (4.0995e+00)    Acc@1  30.00 ( 23.05)   Acc@5 100.00 ( 91.26)
Epoch: [0][160/441]    Time  0.032 ( 0.071)    Data  0.002 ( 0.047)    Loss 1.4663e+00 (3.9421e+00)    Acc@1  20.00 ( 22.86)   Acc@5 100.00 ( 91.80)
Epoch: [0][170/441]    Time  0.165 ( 0.071)    Data  0.157 ( 0.048)    Loss 1.4658e+00 (3.7951e+00)    Acc@1  10.00 ( 22.81)   Acc@5 100.00 ( 92.28)
Epoch: [0][180/441]    Time  0.031 ( 0.070)    Data  0.002 ( 0.047)    Loss 1.7913e+00 (3.6673e+00)    Acc@1  10.00 ( 22.87)   Acc@5 100.00 ( 92.71)
Epoch: [0][190/441]    Time  0.056 ( 0.070)    Data  0.048 ( 0.048)    Loss 1.4729e+00 (3.5537e+00)    Acc@1  20.00 ( 23.09)   Acc@5 100.00 ( 93.09)
Epoch: [0][200/441]    Time  0.105 ( 0.070)    Data  0.097 ( 0.048)    Loss 1.1443e+00 (3.4520e+00)    Acc@1  60.00 ( 23.43)   Acc@5 100.00 ( 93.43)
Epoch: [0][210/441]    Time  0.028 ( 0.070)    Data  0.000 ( 0.048)    Loss 1.6849e+00 (3.3612e+00)    Acc@1  10.00 ( 23.46)   Acc@5 100.00 ( 93.74)
Epoch: [0][220/441]    Time  0.146 ( 0.070)    Data  0.138 ( 0.048)    Loss 1.5837e+00 (3.2794e+00)    Acc@1  10.00 ( 23.17)   Acc@5 100.00 ( 94.03)
Epoch: [0][230/441]    Time  0.085 ( 0.070)    Data  0.077 ( 0.048)    Loss 1.6235e+00 (3.2018e+00)    Acc@1  20.00 ( 23.38)   Acc@5 100.00 ( 94.29)
Epoch: [0][240/441]    Time  0.026 ( 0.070)    Data  0.000 ( 0.048)    Loss 1.6338e+00 (3.1287e+00)    Acc@1   0.00 ( 23.44)   Acc@5 100.00 ( 94.52)
Epoch: [0][250/441]    Time  0.147 ( 0.070)    Data  0.138 ( 0.048)    Loss 1.5658e+00 (3.0687e+00)    Acc@1  10.00 ( 23.11)   Acc@5 100.00 ( 94.74)
Epoch: [0][260/441]    Time  0.057 ( 0.069)    Data  0.048 ( 0.048)    Loss 1.5897e+00 (3.0108e+00)    Acc@1   0.00 ( 22.99)   Acc@5 100.00 ( 94.94)
Epoch: [0][270/441]    Time  0.058 ( 0.069)    Data  0.048 ( 0.048)    Loss 1.5185e+00 (2.9545e+00)    Acc@1  30.00 ( 23.06)   Acc@5 100.00 ( 95.13)
Epoch: [0][280/441]    Time  0.169 ( 0.069)    Data  0.161 ( 0.048)    Loss 1.6898e+00 (2.9011e+00)    Acc@1   0.00 ( 23.20)   Acc@5 100.00 ( 95.30)
Epoch: [0][290/441]    Time  0.028 ( 0.069)    Data  0.000 ( 0.048)    Loss 1.3844e+00 (2.8514e+00)    Acc@1  20.00 ( 23.47)   Acc@5 100.00 ( 95.46)
Epoch: [0][300/441]    Time  0.026 ( 0.070)    Data  0.000 ( 0.049)    Loss 1.4348e+00 (2.8068e+00)    Acc@1  40.00 ( 23.49)   Acc@5 100.00 ( 95.61)
Epoch: [0][310/441]    Time  0.036 ( 0.070)    Data  0.000 ( 0.048)    Loss 1.5076e+00 (2.7652e+00)    Acc@1  20.00 ( 23.44)   Acc@5 100.00 ( 95.76)
Epoch: [0][320/441]    Time  0.034 ( 0.069)    Data  0.000 ( 0.048)    Loss 1.4182e+00 (2.7256e+00)    Acc@1  10.00 ( 23.33)   Acc@5 100.00 ( 95.89)
Epoch: [0][330/441]    Time  0.027 ( 0.069)    Data  0.000 ( 0.048)    Loss 1.4312e+00 (2.6859e+00)    Acc@1  30.00 ( 23.41)   Acc@5 100.00 ( 96.01)
Epoch: [0][340/441]    Time  0.029 ( 0.070)    Data  0.000 ( 0.048)    Loss 1.3906e+00 (2.6504e+00)    Acc@1  20.00 ( 23.34)   Acc@5 100.00 ( 96.13)
Epoch: [0][350/441]    Time  0.029 ( 0.070)    Data  0.021 ( 0.048)    Loss 1.2557e+00 (2.6155e+00)    Acc@1  50.00 ( 23.56)   Acc@5 100.00 ( 96.24)
Epoch: [0][360/441]    Time  0.145 ( 0.070)    Data  0.137 ( 0.048)    Loss 1.2984e+00 (2.5825e+00)    Acc@1  50.00 ( 23.57)   Acc@5 100.00 ( 96.34)
Epoch: [0][370/441]    Time  0.028 ( 0.070)    Data  0.000 ( 0.048)    Loss 1.5003e+00 (2.5506e+00)    Acc@1  10.00 ( 23.67)   Acc@5 100.00 ( 96.44)
Epoch: [0][380/441]    Time  0.158 ( 0.070)    Data  0.147 ( 0.048)    Loss 1.2910e+00 (2.5215e+00)    Acc@1  40.00 ( 23.81)   Acc@5 100.00 ( 96.54)
Epoch: [0][390/441]    Time  0.030 ( 0.069)    Data  0.003 ( 0.048)    Loss 1.6162e+00 (2.4964e+00)    Acc@1  30.00 ( 23.94)   Acc@5 100.00 ( 96.62)
Epoch: [0][400/441]    Time  0.169 ( 0.069)    Data  0.161 ( 0.048)    Loss 1.4447e+00 (2.4688e+00)    Acc@1  20.00 ( 24.29)   Acc@5 100.00 ( 96.71)
Epoch: [0][410/441]    Time  0.155 ( 0.070)    Data  0.147 ( 0.048)    Loss 1.3850e+00 (2.4426e+00)    Acc@1  30.00 ( 24.43)   Acc@5 100.00 ( 96.79)
Epoch: [0][420/441]    Time  0.025 ( 0.069)    Data  0.000 ( 0.048)    Loss 1.3115e+00 (2.4172e+00)    Acc@1  40.00 ( 24.68)   Acc@5 100.00 ( 96.86)
Epoch: [0][430/441]    Time  0.175 ( 0.069)    Data  0.167 ( 0.048)    Loss 1.3211e+00 (2.3923e+00)    Acc@1  30.00 ( 24.85)   Acc@5 100.00 ( 96.94)
Epoch: [0][440/441]    Time  0.131 ( 0.069)    Data  0.000 ( 0.048)    Loss 1.1985e+00 (2.3684e+00)    Acc@1  40.00 ( 25.18)   Acc@5 100.00 ( 97.00)
Test: [ 0/81]   Time  0.855 ( 0.855)    Loss 1.5621e+00 (1.5621e+00)    Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [10/81]   Time  0.006 ( 0.125)    Loss 1.3568e+00 (1.3213e+00)    Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [20/81]   Time  0.008 ( 0.101)    Loss 1.3765e+00 (1.3235e+00)    Acc@1  10.00 (  0.48)   Acc@5 100.00 (100.00)
Test: [30/81]   Time  0.010 ( 0.089)    Loss 1.2447e+00 (1.3082e+00)    Acc@1  10.00 (  6.45)   Acc@5 100.00 (100.00)
Test: [40/81]   Time  0.009 ( 0.088)    Loss 1.2227e+00 (1.3024e+00)    Acc@1  40.00 (  8.78)   Acc@5 100.00 (100.00)
Test: [50/81]   Time  0.082 ( 0.084)    Loss 8.5190e-01 (1.2285e+00)    Acc@1 100.00 ( 25.69)   Acc@5 100.00 (100.00)
Test: [60/81]   Time  0.086 ( 0.083)    Loss 9.7332e-01 (1.1812e+00)    Acc@1  80.00 ( 36.72)   Acc@5 100.00 (100.00)
Test: [70/81]   Time  0.017 ( 0.079)    Loss 1.1945e+00 (1.2000e+00)    Acc@1  50.00 ( 38.59)   Acc@5 100.00 (100.00)
Test: [80/81]   Time  0.007 ( 0.077)    Loss 1.5687e+00 (1.2169e+00)    Acc@1  20.00 ( 39.63)   Acc@5 100.00 (100.00)
 * Acc@1 39.630 Acc@5 100.000
```

At the end of the first epoch:

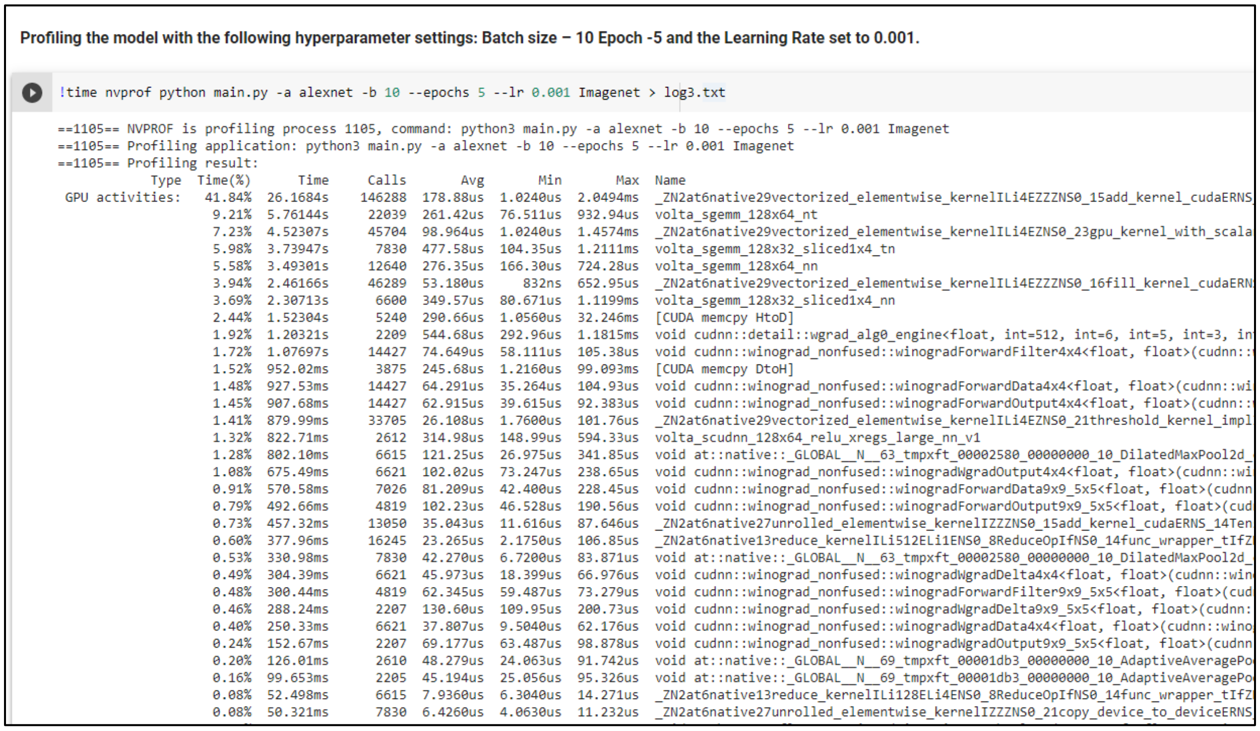


Profiling the model with the following hyperparameter settings: Batch size – 10 Epoch -5 and the Learning Rate set to 0.001.

```
!time python main.py -a alexnet -b 10 --epochs 5 --lr 0.001 Imagenet

Epoch: [0][420/441]    Time  0.025 ( 0.069)    Data  0.000 ( 0.048)    Loss 1.3115e+00 (2.4172e+00)    Acc@1  40.00 ( 24.68)   Acc@5 100.00 ( 96.86)
Epoch: [0][430/441]    Time  0.175 ( 0.069)    Data  0.167 ( 0.048)    Loss 1.3211e+00 (2.3923e+00)    Acc@1  30.00 ( 24.85)   Acc@5 100.00 ( 96.94)
Epoch: [0][440/441]    Time  0.131 ( 0.069)    Data  0.000 ( 0.048)    Loss 1.1985e+00 (2.3684e+00)    Acc@1  40.00 ( 25.18)   Acc@5 100.00 ( 97.00)
Test: [ 0/81]   Time  0.855 ( 0.855)    Loss 1.5621e+00 (1.5621e+00)    Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [10/81]   Time  0.006 ( 0.125)    Loss 1.3568e+00 (1.3213e+00)    Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [20/81]   Time  0.008 ( 0.101)    Loss 1.3765e+00 (1.3235e+00)    Acc@1  10.00 (  0.48)   Acc@5 100.00 (100.00)
Test: [30/81]   Time  0.010 ( 0.089)    Loss 1.2447e+00 (1.3082e+00)    Acc@1  10.00 (  6.45)   Acc@5 100.00 (100.00)
Test: [40/81]   Time  0.009 ( 0.088)    Loss 1.2227e+00 (1.3024e+00)    Acc@1  40.00 (  8.78)   Acc@5 100.00 (100.00)
Test: [50/81]   Time  0.082 ( 0.084)    Loss 8.5190e-01 (1.2285e+00)    Acc@1 100.00 ( 25.69)   Acc@5 100.00 (100.00)
Test: [60/81]   Time  0.086 ( 0.083)    Loss 9.7332e-01 (1.1812e+00)    Acc@1  80.00 ( 36.72)   Acc@5 100.00 (100.00)
Test: [70/81]   Time  0.017 ( 0.079)    Loss 1.1945e+00 (1.2000e+00)    Acc@1  50.00 ( 38.59)   Acc@5 100.00 (100.00)
Test: [80/81]   Time  0.007 ( 0.077)    Loss 1.5687e+00 (1.2169e+00)    Acc@1  20.00 ( 39.63)   Acc@5 100.00 (100.00)
 * Acc@1 39.630 Acc@5 100.000
Epoch: [1][  0/441]    Time  0.620 ( 0.620)    Data  0.603 ( 0.603)    Loss 1.3165e+00 (1.3165e+00)    Acc@1  30.00 ( 30.00)   Acc@5 100.00 (100.00)
Epoch: [1][ 10/441]    Time  0.031 ( 0.112)    Data  0.000 ( 0.087)    Loss 1.3113e+00 (1.2857e+00)    Acc@1  50.00 ( 39.09)   Acc@5 100.00 (100.00)
Epoch: [1][ 20/441]    Time  0.070 ( 0.093)    Data  0.062 ( 0.064)    Loss 1.2147e+00 (1.2705e+00)    Acc@1  30.00 ( 38.57)   Acc@5 100.00 (100.00)
Epoch: [1][ 30/441]    Time  0.119 ( 0.086)    Data  0.109 ( 0.060)    Loss 1.0326e+00 (1.2791e+00)    Acc@1  60.00 ( 40.32)   Acc@5 100.00 (100.00)
```

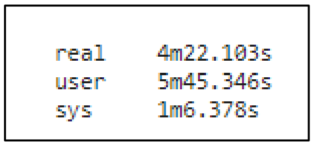
The time taken to process has significantly increased as compared to the previous two runs as could be expected from the increase in epoch and smaller learning rate. With the decreases in learning rate the step size decreases, and the processing time increases. One can also note that the accuracy has significantly improved for this run in both environments.



Profiling the model with the following hyperparameter settings: Batch size – 10 Epoch -5 and the Learning Rate set to 0.001.

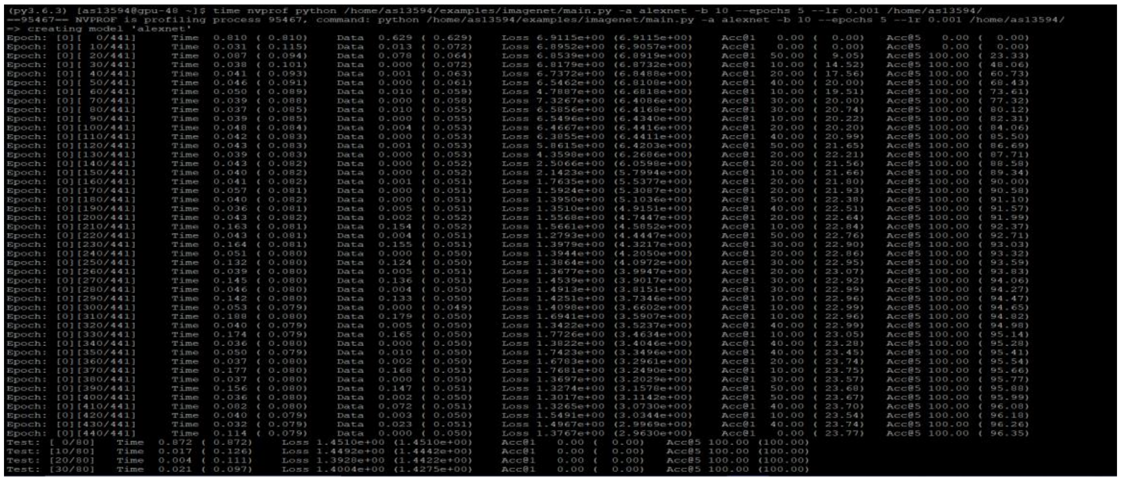
```
!time nvprof python main.py -a alexnet -b 10 --epochs 5 --lr 0.001 Imagenet > log3.txt

==1105== NVPROF is profiling process 1105, command: python3 main.py -a alexnet -b 10 --epochs 5 --lr 0.001 Imagenet
==1105== Profiling application: python3 main.py -a alexnet -b 10 --epochs 5 --lr 0.001 Imagenet
==1105== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   41.84%  26.1684s    146288  178.88us  1.0240us  2.0494ms  _ZN2at6native29vectorized_elementwise_kernelILi4EZZZNS0_15add_kernel_cudaERNS
                    9.21%  5.76144s     22039  261.42us  76.511us  932.94us  volta_sgemm_128x64_nt
                    7.23%  4.52307s     45704  98.964us  1.0240us  1.4574ms  _ZN2at6native29vectorized_elementwise_kernelILi4EZNS0_23gpu_kernel_with_scala
                    5.98%  3.73947s      7830  477.58us  104.35us  1.2111ms  volta_sgemm_128x32_sliced1x4_tn
                    5.58%  3.49301s     12640  276.35us  166.30us  724.28us  volta_sgemm_128x64_nn
                    3.94%  2.46166s     46289  53.180us     832ns  652.95us  _ZN2at6native29vectorized_elementwise_kernelILi4EZZZNS0_16fill_kernel_cudaERN
                    3.69%  2.30713s      6600  349.57us  80.671us  1.1199ms  volta_sgemm_128x32_sliced1x4_nn
                    2.44%  1.52304s      5240  290.66us  1.0560us  32.246ms  [CUDA memcpy HtoD]
                    1.92%  1.20321s      2209  544.68us  292.96us  1.1815ms  void cudnn::detail::wgrad_alg0_engine<float, int=512, int=6, int=5, int=3, in
                    1.72%  1.07697s     14427  74.649us  58.111us  105.38us  void cudnn::winograd_nonfused::winogradForwardFilter4x4<float, float>(cudnn::
                    1.52%  952.02ms      3875  245.68us  1.2160us  99.093ms  [CUDA memcpy DtoH]
                    1.48%  927.53ms     14427  64.291us  35.264us  104.93us  void cudnn::winograd_nonfused::winogradForwardData4x4<float, float>(cudnn::wi
                    1.45%  907.68ms     14427  62.915us  39.615us  92.383us  void cudnn::winograd_nonfused::winogradForwardOutput4x4<float, float>(cudnn::
                    1.41%  879.99ms     33705  26.108us  1.7600us  101.76us  _ZN2at6native29vectorized_elementwise_kernelILi4EZNS0_21threshold_kernel_impl
                    1.32%  822.71ms      2612  314.98us  148.99us  594.33us  volta_scudnn_128x64_relu_xregs_large_nn_v1
                    1.28%  802.10ms      6615  121.25us  26.975us  341.85us  void at::native::_GLOBAL__N__63_tmpxft_00002580_00000000_10_DilatedMaxPool2d_
                    1.08%  675.49ms      6621  102.02us  73.247us  238.65us  void cudnn::winograd_nonfused::winogradWgradOutput4x4<float, float>(cudnn::wi
                    0.91%  570.58ms      7026  81.209us  42.400us  228.45us  void cudnn::winograd_nonfused::winogradForwardData9x9_5x5<float, float>(cudnn
                    0.79%  492.66ms      4819  102.23us  46.528us  190.56us  void cudnn::winograd_nonfused::winogradForwardOutput9x9_5x5<float, float>(cud
                    0.73%  457.32ms     13050  35.043us  11.616us  87.646us  _ZN2at6native27unrolled_elementwise_kernelIZZZNS0_15add_kernel_cudaERNS_14Ten
                    0.60%  377.96ms     16245  23.265us  2.1750us  106.85us  _ZN2at6native13reduce_kernelILi512ELi1ENS0_8ReduceOpIfNS0_14func_wrapper_tIfZ
                    0.53%  330.98ms      7830  42.270us  6.7200us  83.871us  void at::native::_GLOBAL__N__63_tmpxft_00002580_00000000_10_DilatedMaxPool2d_
                    0.49%  304.39ms      6621  45.973us  18.399us  66.976us  void cudnn::winograd_nonfused::winogradWgradDelta4x4<float, float>(cudnn::win
                    0.48%  300.44ms      4819  62.345us  59.487us  73.279us  void cudnn::winograd_nonfused::winogradForwardFilter9x9_5x5<float, float>(cud
                    0.46%  288.24ms      2207  130.60us  109.95us  200.73us  void cudnn::winograd_nonfused::winogradWgradDelta9x9_5x5<float, float>(cudnn:
                    0.40%  250.33ms      6621  37.807us  9.5040us  62.176us  void cudnn::winograd_nonfused::winogradWgradData4x4<float, float>(cudnn::wino
                    0.24%  152.67ms      2207  69.177us  63.487us  98.878us  void cudnn::winograd_nonfused::winogradWgradOutput9x9_5x5<float, float>(cudnn
                    0.20%  126.01ms      2610  48.279us  24.063us  91.742us  void at::native::_GLOBAL__N__69_tmpxft_00001db3_00000000_10_AdaptiveAveragePo
                    0.16%  99.653ms      2205  45.194us  25.056us  95.326us  void at::native::_GLOBAL__N__69_tmpxft_00001db3_00000000_10_AdaptiveAveragePo
                    0.08%  52.498ms      6615  7.9360us  6.3040us  14.271us  _ZN2at6native13reduce_kernelILi128ELi4ENS0_8ReduceOpIfNS0_14func_wrapper_tIfZ
                    0.08%  50.321ms      7830  6.4260us  4.0630us  11.232us  _ZN2at6native27unrolled_elementwise_kernelIZZZNS0_21copy_device_to_deviceERNS
```

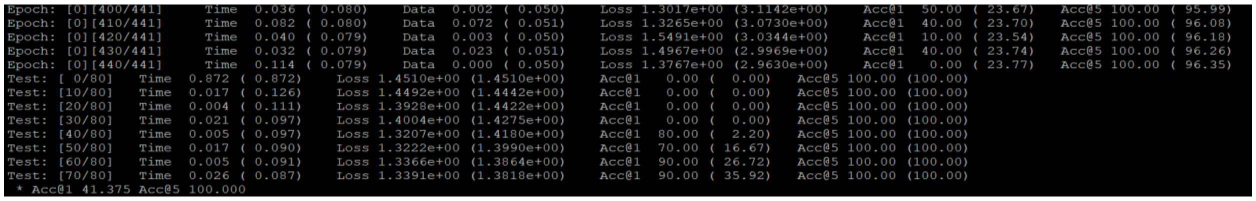


```
real    4m22.103s
user    5m45.346s
sys     1m6.378s
```

Running on BareMetal : We train the model with the same hyperparameter settings: Batch size – 10 Epoch -5 and the Learning Rate has been set to 0.001. It takes approximately 441 iterations in each epoch and processes images in batches of 10 and passes over the entire dataset 5 times.

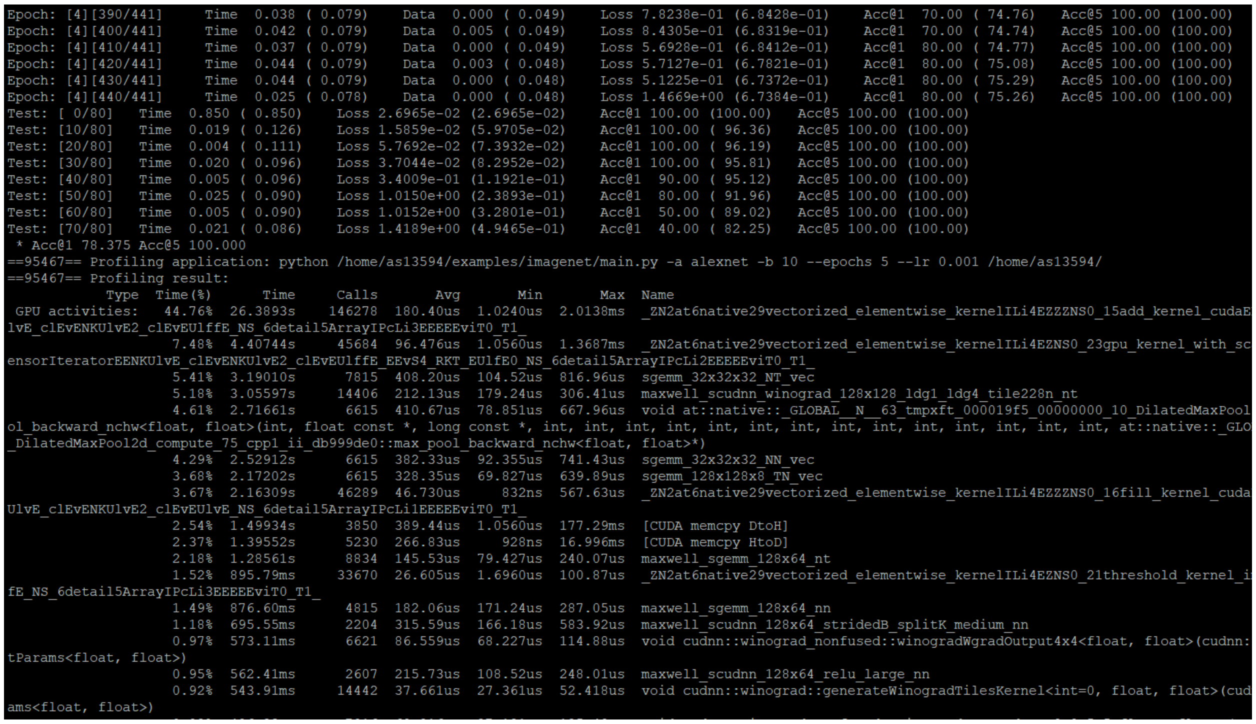


```
(py3.6.3) [as13594@gpu-48 ~]$ time nvprof python /home/as13594/examples/imagenet/main.py -a alexnet -b 10 --epochs 5 --lr 0.001 /home/as13594/
==95467== NVPROF is profiling process 95467, command: python /home/as13594/examples/imagenet/main.py -a alexnet -b 10 --epochs 5 --lr 0.001 /home/as13594/
=> creating model 'alexnet'
Epoch: [0][  0/441]     Time  0.810 ( 0.810)    Data  0.629 ( 0.629)    Loss 6.9115e+00 (6.9115e+00)    Acc@1   0.00 (  0.00)   Acc@5   0.00 (  0.00)
Epoch: [0][ 10/441]     Time  0.031 ( 0.115)    Data  0.013 ( 0.072)    Loss 6.8952e+00 (6.9057e+00)    Acc@1   0.00 (  0.00)   Acc@5   0.00 (  0.00)
Epoch: [0][ 20/441]     Time  0.087 ( 0.094)    Data  0.078 ( 0.064)    Loss 6.8539e+00 (6.8919e+00)    Acc@1  50.00 (  9.05)   Acc@5 100.00 ( 23.33)
Epoch: [0][ 30/441]     Time  0.038 ( 0.101)    Data  0.000 ( 0.072)    Loss 6.8179e+00 (6.8732e+00)    Acc@1  10.00 ( 14.52)   Acc@5 100.00 ( 48.06)
Epoch: [0][ 40/441]     Time  0.041 ( 0.093)    Data  0.001 ( 0.063)    Loss 6.7372e+00 (6.8488e+00)    Acc@1  20.00 ( 17.56)   Acc@5 100.00 ( 60.73)
Epoch: [0][ 50/441]     Time  0.046 ( 0.091)    Data  0.000 ( 0.061)    Loss 6.5462e+00 (6.8108e+00)    Acc@1  40.00 ( 20.00)   Acc@5 100.00 ( 68.43)
Epoch: [0][ 60/441]     Time  0.050 ( 0.089)    Data  0.010 ( 0.059)    Loss 4.7887e+00 (6.6818e+00)    Acc@1  10.00 ( 19.51)   Acc@5 100.00 ( 73.61)
Epoch: [0][ 70/441]     Time  0.039 ( 0.088)    Data  0.000 ( 0.058)    Loss 7.3267e+00 (6.4086e+00)    Acc@1  30.00 ( 20.00)   Acc@5 100.00 ( 77.32)
Epoch: [0][ 80/441]     Time  0.037 ( 0.085)    Data  0.010 ( 0.055)    Loss 6.5856e+00 (6.4168e+00)    Acc@1  30.00 ( 20.74)   Acc@5 100.00 ( 80.12)
Epoch: [0][ 90/441]     Time  0.039 ( 0.085)    Data  0.000 ( 0.055)    Loss 6.5496e+00 (6.4340e+00)    Acc@1  10.00 ( 20.22)   Acc@5 100.00 ( 82.31)
Epoch: [0][100/441]     Time  0.048 ( 0.084)    Data  0.004 ( 0.053)    Loss 6.4667e+00 (6.4416e+00)    Acc@1  20.00 ( 20.20)   Acc@5 100.00 ( 84.06)
Epoch: [0][110/441]     Time  0.042 ( 0.083)    Data  0.000 ( 0.053)    Loss 6.3855e+00 (6.4411e+00)    Acc@1  40.00 ( 20.99)   Acc@5 100.00 ( 85.50)
Epoch: [0][120/441]     Time  0.043 ( 0.083)    Data  0.001 ( 0.053)    Loss 5.8615e+00 (6.4203e+00)    Acc@1  50.00 ( 21.65)   Acc@5 100.00 ( 86.69)
Epoch: [0][130/441]     Time  0.039 ( 0.083)    Data  0.000 ( 0.053)    Loss 4.3598e+00 (6.2686e+00)    Acc@1  20.00 ( 22.21)   Acc@5 100.00 ( 87.71)
Epoch: [0][140/441]     Time  0.043 ( 0.082)    Data  0.000 ( 0.052)    Loss 2.5066e+00 (6.0598e+00)    Acc@1  20.00 ( 21.56)   Acc@5 100.00 ( 88.58)
Epoch: [0][150/441]     Time  0.040 ( 0.082)    Data  0.000 ( 0.052)    Loss 2.1423e+00 (5.7994e+00)    Acc@1  10.00 ( 21.66)   Acc@5 100.00 ( 89.34)
Epoch: [0][160/441]     Time  0.041 ( 0.082)    Data  0.001 ( 0.051)    Loss 1.7635e+00 (5.5377e+00)    Acc@1  20.00 ( 21.80)   Acc@5 100.00 ( 90.00)
Epoch: [0][170/441]     Time  0.057 ( 0.081)    Data  0.000 ( 0.051)    Loss 1.5924e+00 (5.3087e+00)    Acc@1  20.00 ( 21.93)   Acc@5 100.00 ( 90.58)
Epoch: [0][180/441]     Time  0.040 ( 0.082)    Data  0.000 ( 0.051)    Loss 1.3950e+00 (5.1036e+00)    Acc@1  50.00 ( 22.38)   Acc@5 100.00 ( 91.10)
Epoch: [0][190/441]     Time  0.036 ( 0.081)    Data  0.005 ( 0.051)    Loss 1.3510e+00 (4.9151e+00)    Acc@1  40.00 ( 22.51)   Acc@5 100.00 ( 91.57)
Epoch: [0][200/441]     Time  0.043 ( 0.082)    Data  0.002 ( 0.052)    Loss 1.5568e+00 (4.7447e+00)    Acc@1  20.00 ( 22.64)   Acc@5 100.00 ( 91.99)
Epoch: [0][210/441]     Time  0.163 ( 0.081)    Data  0.154 ( 0.052)    Loss 1.5661e+00 (4.5852e+00)    Acc@1  10.00 ( 22.84)   Acc@5 100.00 ( 92.37)
Epoch: [0][220/441]     Time  0.043 ( 0.081)    Data  0.004 ( 0.051)    Loss 1.2793e+00 (4.4447e+00)    Acc@1  50.00 ( 22.76)   Acc@5 100.00 ( 92.71)
Epoch: [0][230/441]     Time  0.164 ( 0.081)    Data  0.155 ( 0.051)    Loss 1.3979e+00 (4.3217e+00)    Acc@1  30.00 ( 22.90)   Acc@5 100.00 ( 93.03)
Epoch: [0][240/441]     Time  0.051 ( 0.080)    Data  0.000 ( 0.050)    Loss 1.3944e+00 (4.2050e+00)    Acc@1  20.00 ( 22.86)   Acc@5 100.00 ( 93.32)
Epoch: [0][250/441]     Time  0.132 ( 0.080)    Data  0.124 ( 0.050)    Loss 1.3864e+00 (4.0972e+00)    Acc@1  30.00 ( 22.95)   Acc@5 100.00 ( 93.59)
Epoch: [0][260/441]     Time  0.039 ( 0.080)    Data  0.005 ( 0.051)    Loss 1.3677e+00 (3.9947e+00)    Acc@1  20.00 ( 23.07)   Acc@5 100.00 ( 93.83)
Epoch: [0][270/441]     Time  0.145 ( 0.080)    Data  0.136 ( 0.051)    Loss 1.4539e+00 (3.9017e+00)    Acc@1  30.00 ( 22.92)   Acc@5 100.00 ( 94.06)
Epoch: [0][280/441]     Time  0.046 ( 0.080)    Data  0.004 ( 0.050)    Loss 1.4913e+00 (3.8151e+00)    Acc@1  30.00 ( 22.99)   Acc@5 100.00 ( 94.27)
Epoch: [0][290/441]     Time  0.142 ( 0.080)    Data  0.133 ( 0.050)    Loss 1.4251e+00 (3.7346e+00)    Acc@1  10.00 ( 22.96)   Acc@5 100.00 ( 94.47)
Epoch: [0][300/441]     Time  0.053 ( 0.079)    Data  0.000 ( 0.049)    Loss 1.4098e+00 (3.6602e+00)    Acc@1  10.00 ( 22.99)   Acc@5 100.00 ( 94.65)
Epoch: [0][310/441]     Time  0.188 ( 0.080)    Data  0.179 ( 0.050)    Loss 1.6941e+00 (3.5907e+00)    Acc@1  10.00 ( 22.96)   Acc@5 100.00 ( 94.82)
Epoch: [0][320/441]     Time  0.040 ( 0.079)    Data  0.005 ( 0.050)    Loss 1.3422e+00 (3.5237e+00)    Acc@1  40.00 ( 22.99)   Acc@5 100.00 ( 94.98)
Epoch: [0][330/441]     Time  0.174 ( 0.079)    Data  0.165 ( 0.050)    Loss 1.7726e+00 (3.4634e+00)    Acc@1  10.00 ( 23.05)   Acc@5 100.00 ( 95.14)
Epoch: [0][340/441]     Time  0.036 ( 0.080)    Data  0.000 ( 0.050)    Loss 1.3822e+00 (3.4046e+00)    Acc@1  40.00 ( 23.28)   Acc@5 100.00 ( 95.28)
Epoch: [0][350/441]     Time  0.050 ( 0.079)    Data  0.010 ( 0.050)    Loss 1.7423e+00 (3.3496e+00)    Acc@1  40.00 ( 23.45)   Acc@5 100.00 ( 95.41)
Epoch: [0][360/441]     Time  0.037 ( 0.080)    Data  0.002 ( 0.050)    Loss 1.6783e+00 (3.2961e+00)    Acc@1  20.00 ( 23.74)   Acc@5 100.00 ( 95.54)
Epoch: [0][370/441]     Time  0.177 ( 0.080)    Data  0.168 ( 0.051)    Loss 1.7681e+00 (3.2490e+00)    Acc@1  10.00 ( 23.75)   Acc@5 100.00 ( 95.66)
Epoch: [0][380/441]     Time  0.037 ( 0.080)    Data  0.000 ( 0.050)    Loss 1.3697e+00 (3.2029e+00)    Acc@1  30.00 ( 23.57)   Acc@5 100.00 ( 95.77)
Epoch: [0][390/441]     Time  0.156 ( 0.080)    Data  0.147 ( 0.051)    Loss 1.3274e+00 (3.1578e+00)    Acc@1  50.00 ( 23.68)   Acc@5 100.00 ( 95.88)
Epoch: [0][400/441]     Time  0.036 ( 0.080)    Data  0.002 ( 0.050)    Loss 1.3017e+00 (3.1142e+00)    Acc@1  50.00 ( 23.67)   Acc@5 100.00 ( 95.99)
Epoch: [0][410/441]     Time  0.082 ( 0.080)    Data  0.072 ( 0.051)    Loss 1.3265e+00 (3.0730e+00)    Acc@1  40.00 ( 23.70)   Acc@5 100.00 ( 96.08)
Epoch: [0][420/441]     Time  0.040 ( 0.079)    Data  0.003 ( 0.050)    Loss 1.5491e+00 (3.0344e+00)    Acc@1  10.00 ( 23.54)   Acc@5 100.00 ( 96.18)
Epoch: [0][430/441]     Time  0.032 ( 0.079)    Data  0.023 ( 0.051)    Loss 1.4967e+00 (2.9969e+00)    Acc@1  40.00 ( 23.74)   Acc@5 100.00 ( 96.26)
Epoch: [0][440/441]     Time  0.114 ( 0.079)    Data  0.000 ( 0.050)    Loss 1.3767e+00 (2.9630e+00)    Acc@1   0.00 ( 23.77)   Acc@5 100.00 ( 96.35)
Test: [ 0/80]   Time  0.872 ( 0.872)    Loss 1.4510e+00 (1.4510e+00)    Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [10/80]   Time  0.017 ( 0.126)    Loss 1.4492e+00 (1.4442e+00)    Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [20/80]   Time  0.004 ( 0.111)    Loss 1.3928e+00 (1.4422e+00)    Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [30/80]   Time  0.021 ( 0.097)    Loss 1.4004e+00 (1.4275e+00)    Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
```

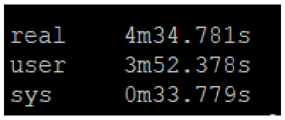


```
Epoch: [0][400/441]     Time  0.036 ( 0.080)    Data  0.002 ( 0.050)    Loss 1.3017e+00 (3.1142e+00)    Acc@1  50.00 ( 23.67)   Acc@5 100.00 ( 95.99)
Epoch: [0][410/441]     Time  0.082 ( 0.080)    Data  0.072 ( 0.051)    Loss 1.3265e+00 (3.0730e+00)    Acc@1  40.00 ( 23.70)   Acc@5 100.00 ( 96.08)
Epoch: [0][420/441]     Time  0.040 ( 0.079)    Data  0.003 ( 0.050)    Loss 1.5491e+00 (3.0344e+00)    Acc@1  10.00 ( 23.54)   Acc@5 100.00 ( 96.18)
Epoch: [0][430/441]     Time  0.032 ( 0.079)    Data  0.023 ( 0.051)    Loss 1.4967e+00 (2.9969e+00)    Acc@1  40.00 ( 23.74)   Acc@5 100.00 ( 96.26)
Epoch: [0][440/441]     Time  0.114 ( 0.079)    Data  0.000 ( 0.050)    Loss 1.3767e+00 (2.9630e+00)    Acc@1   0.00 ( 23.77)   Acc@5 100.00 ( 96.35)
Test: [ 0/80]   Time  0.872 ( 0.872)    Loss 1.4510e+00 (1.4510e+00)    Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [10/80]   Time  0.017 ( 0.126)    Loss 1.4492e+00 (1.4442e+00)    Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [20/80]   Time  0.004 ( 0.111)    Loss 1.3928e+00 (1.4422e+00)    Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [30/80]   Time  0.021 ( 0.097)    Loss 1.4004e+00 (1.4275e+00)    Acc@1   0.00 (  0.00)   Acc@5 100.00 (100.00)
Test: [40/80]   Time  0.005 ( 0.097)    Loss 1.3207e+00 (1.4180e+00)    Acc@1  80.00 (  2.20)   Acc@5 100.00 (100.00)
Test: [50/80]   Time  0.017 ( 0.090)    Loss 1.3222e+00 (1.3990e+00)    Acc@1  70.00 ( 16.67)   Acc@5 100.00 (100.00)
Test: [60/80]   Time  0.005 ( 0.091)    Loss 1.3366e+00 (1.3864e+00)    Acc@1  90.00 ( 26.72)   Acc@5 100.00 (100.00)
Test: [70/80]   Time  0.026 ( 0.087)    Loss 1.3391e+00 (1.3818e+00)    Acc@1  90.00 ( 35.92)   Acc@5 100.00 (100.00)
 * Acc@1 41.375 Acc@5 100.000
```

At the end of the first epoch: One can observe the accuracy is 41.375. Once all five epochs' have been processed and the training is complete. The time taken to process has significantly increased as compared to the previous two runs as could be expected from the increase in epoch and smaller learning rate. With the decreases in learning rate the step size decreases, and the processing time increases. One can also note that the accuracy has significantly improved for this run due to hyperparameter tuning to 78%.



```
Epoch: [4][390/441]     Time  0.038 ( 0.079)    Data  0.000 ( 0.049)    Loss 7.8238e-01 (6.8428e-01)    Acc@1  70.00 ( 74.76)   Acc@5 100.00 (100.00)
Epoch: [4][400/441]     Time  0.042 ( 0.079)    Data  0.005 ( 0.049)    Loss 8.4305e-01 (6.8319e-01)    Acc@1  70.00 ( 74.74)   Acc@5 100.00 (100.00)
Epoch: [4][410/441]     Time  0.037 ( 0.079)    Data  0.000 ( 0.049)    Loss 5.6928e-01 (6.8412e-01)    Acc@1  80.00 ( 74.77)   Acc@5 100.00 (100.00)
Epoch: [4][420/441]     Time  0.044 ( 0.079)    Data  0.003 ( 0.048)    Loss 5.7127e-01 (6.7821e-01)    Acc@1  80.00 ( 75.08)   Acc@5 100.00 (100.00)
Epoch: [4][430/441]     Time  0.044 ( 0.079)    Data  0.000 ( 0.048)    Loss 5.1225e-01 (6.7372e-01)    Acc@1  80.00 ( 75.29)   Acc@5 100.00 (100.00)
Epoch: [4][440/441]     Time  0.025 ( 0.078)    Data  0.000 ( 0.048)    Loss 1.4669e+00 (6.7384e-01)    Acc@1  80.00 ( 75.26)   Acc@5 100.00 (100.00)
Test: [ 0/80]   Time  0.850 ( 0.850)    Loss 2.6965e-02 (2.6965e-02)    Acc@1 100.00 (100.00)   Acc@5 100.00 (100.00)
Test: [10/80]   Time  0.019 ( 0.126)    Loss 1.5859e-02 (5.9705e-02)    Acc@1 100.00 ( 96.36)   Acc@5 100.00 (100.00)
Test: [20/80]   Time  0.004 ( 0.111)    Loss 5.7692e-02 (7.3932e-02)    Acc@1 100.00 ( 96.19)   Acc@5 100.00 (100.00)
Test: [30/80]   Time  0.020 ( 0.096)    Loss 3.7044e-02 (8.2952e-02)    Acc@1 100.00 ( 95.81)   Acc@5 100.00 (100.00)
Test: [40/80]   Time  0.005 ( 0.096)    Loss 3.4009e-01 (1.1921e-01)    Acc@1  90.00 ( 95.12)   Acc@5 100.00 (100.00)
Test: [50/80]   Time  0.025 ( 0.090)    Loss 1.0150e+00 (2.3893e-01)    Acc@1  80.00 ( 91.96)   Acc@5 100.00 (100.00)
Test: [60/80]   Time  0.005 ( 0.090)    Loss 1.0152e+00 (3.2801e-01)    Acc@1  50.00 ( 89.02)   Acc@5 100.00 (100.00)
Test: [70/80]   Time  0.021 ( 0.086)    Loss 1.4189e+00 (4.9465e-01)    Acc@1  40.00 ( 82.25)   Acc@5 100.00 (100.00)
 * Acc@1 78.375 Acc@5 100.000
==95467== Profiling application: python /home/as13594/examples/imagenet/main.py -a alexnet -b 10 --epochs 5 --lr 0.001 /home/as13594/
==95467== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   44.76%  26.3893s    146278  180.40us  1.0240us  2.0138ms  _ZN2at6native29vectorized_elementwise_kernelILi4EZZZNS0_15add_kernel_cudaE
lvE_clEvENKUlvE2_clEvEUlffE_NS_6detail5ArrayIPcLi3EEEEEviT0_T1_
                    7.48%  4.40744s     45684  96.476us  1.0560us  1.3687ms  _ZN2at6native29vectorized_elementwise_kernelILi4EZNS0_23gpu_kernel_with_sc
ensorIteratorEENKUlvE_clEvENKUlvE2_clEvEUlffE_EEvS4_RKT_EUlfE0_NS_6detail5ArrayIPcLi2EEEEEviT0_T1_
                    5.41%  3.19010s      7815  408.20us  104.52us  816.96us  sgemm_32x32x32_NT_vec
                    5.18%  3.05597s     14406  212.13us  179.24us  306.41us  maxwell_scudnn_winograd_128x128_ldg1_ldg4_tile228n_nt
                    4.61%  2.71661s      6615  410.67us  78.851us  667.96us  void at::native::_GLOBAL__N__63_tmpxft_000019f5_00000000_10_DilatedMaxPool
ol_backward_nchw<float, float>(int, float const *, long const *, int, int, int, int, int, int, int, int, int, int, int, int, int, int, at::native::_GLO
_DilatedMaxPool2d_compute_75_cpp1_ii_db999de0::max_pool_backward_nchw<float, float>*)
                    4.29%  2.52912s      6615  382.33us  92.355us  741.43us  sgemm_32x32x32_NN_vec
                    3.68%  2.17202s      6615  328.35us  69.827us  639.89us  sgemm_128x128x8_TN_vec
                    3.67%  2.16309s     46289  46.730us     832ns  567.63us  _ZN2at6native29vectorized_elementwise_kernelILi4EZZZNS0_16fill_kernel_cuda
UlvE_clEvENKUlvE2_clEvEUlvE_NS_6detail5ArrayIPcLi1EEEEEviT0_T1_
                    2.54%  1.49934s      3850  389.44us  1.0560us  177.29ms  [CUDA memcpy DtoH]
                    2.37%  1.39552s      5230  266.83us     928ns  16.996ms  [CUDA memcpy HtoD]
                    2.18%  1.28561s      8834  145.53us  79.427us  240.07us  maxwell_sgemm_128x64_nt
                    1.52%  895.79ms     33670  26.605us  1.6960us  100.87us  _ZN2at6native29vectorized_elementwise_kernelILi4EZNS0_21threshold_kernel_i
fE_NS_6detail5ArrayIPcLi3EEEEEviT0_T1_
                    1.49%  876.60ms      4815  182.06us  171.24us  287.05us  maxwell_sgemm_128x64_nn
                    1.18%  695.55ms      2204  315.59us  166.18us  583.92us  maxwell_scudnn_128x64_stridedB_splitK_medium_nn
                    0.97%  573.11ms      6621  86.559us  68.227us  114.88us  void cudnn::winograd_nonfused::winogradWgradOutput4x4<float, float>(cudnn:
tParams<float, float>)
                    0.95%  562.41ms      2607  215.73us  108.52us  248.01us  maxwell_scudnn_128x64_relu_large_nn
                    0.92%  543.91ms     14442  37.661us  27.361us  52.418us  void cudnn::winograd::generateWinogradTilesKernel<int=0, float, float>(cud
ams<float, float>)
```



```
real    4m34.781s
user    3m52.378s
sys     0m33.779s
```

When we compare the runtime of this job on the BareMetal and Cloud Environments, we notice that the time elapsed in both cases was relatively same. The relative peak in GPU time activity on was also comparable in both cases. The throughput or the number of instructions processed per second is also relatively same.

Conclusion: The choice between using BareMetal and Cloud based solutions depends a lot on the kind of application and data we need to process. In case the application needs scalable, easy to use, secure and variable workloads with a focus towards better resource utilization, cloud environments are preferable. If the focus is on highly optimizable secure environments that offer low network latency and dedicated resource utilization the outlook is better with BareMetal Solutions.