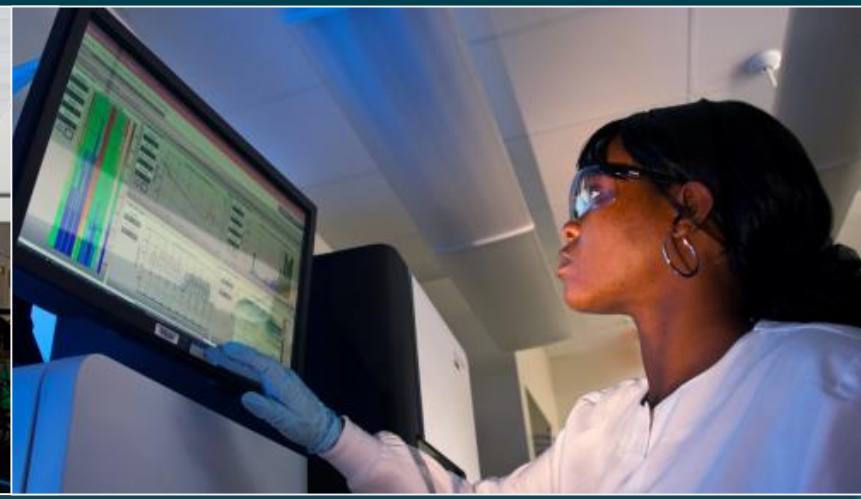
Frederick National Laboratory
for Cancer Research

sponsored by the National Cancer Institute





Hyperparameter Optimization Using CANDLE on Biowulf

Andrew Weisman
High Performance Computing Analyst

January 19, 2021

Agenda

1. Introduction to Hyperparameter Optimization
2. Overview of CANDLE
3. Walk-through: Running a grid search using CANDLE from scratch

Part I

Introduction to Hyperparameter Optimization

Please fill out the survey: https://bit.ly/CANDLE_Workshop_Jan21

Prediction using a linear model

- **Every hour, look out the window and count as many birds and deer as I can see**
- **Question: Can I predict the number of deer *y* given the number of birds *x*?**
  - **Step 1:** Choose a model:
    - Let's choose a linear one: $y = m*x + b$
  - **Step 2:** Calculate the error for time t given arbitrary values of m and b:
    - $E_t = y_t - (m*x_t + b)$
  - **Step 3:** Calculate the Mean Squared Error over the whole dataset:
    - MSE = $\frac{1}{8}\sum_{t=1}^{8} \boldsymbol{E_t}^2$
    - We can also call the MSE the "loss"
  - **Step 4:** Train the model by calculating the parameters m and b that minimize the loss
- **Answer: If the final loss is "small," then we can reasonably accurately predict *y* from *x* using our model**

Time (*t*)	Number of birds (*x*)	Number of deer (*y*)
9:30 AM	5	3
10:30 AM	2	2
11:30 AM	1	0
⋮	⋮	⋮
4:30 PM	2	1

Prediction using a quadratic model

- **Question: Does a quadratic model work better than the linear model?**
  - **Step 1:** Choose a quadratic model:
    - $y = \theta_2 * x^2 + \theta_1 * x + \theta_0$
  - **Step 2:** Calculate the error for time t given arbitrary values of $(\theta_0, \theta_1, \theta_2)$:
    - $E_t = y_t - (\theta_2 * x_t^2 + \theta_1 * x_t + \theta_0)$
  - **Step 3:** Calculate the loss over the whole dataset:
    - MSE = $\frac{1}{8}\sum_{t=1}^{8} E_t^{\,2}$
  - **Step 4:** Train the model by calculating the parameters $(\theta_0, \theta_1, \theta_2)$ that minimize the loss
- **Answer: If the final quadratic loss is smaller than the final linear loss, then we can better predict *y* from *x* using our quadratic model**

Time (*t*)	Number of birds (*x*)	Number of deer (*y*)
9:30 AM	5	3
10:30 AM	2	2
11:30 AM	1	0
⋮	⋮	⋮
4:30 PM	2	1

Prediction using a polynomial model of degree *n*

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

- **Question: A polynomial model of which degree *n* best predicts *y* from *x*?**
  - **Step 1:** Define the model:
    - $y = \sum_{i=0}^{n} \theta_i x^i$
  - **Step 2:** Calculate the error for time *t* given arbitrary values of $\{\theta_i\}$:
    - $E_t = y_t - \sum_{i=0}^{n} \theta_i x_t^i$
  - **Step 3:** Calculate the loss over the whole dataset:
    - MSE = $\frac{1}{8}\sum_{t=1}^{8} E_t^{\,2}$
  - **Step 4:** Train the model by calculating the parameters $\{\theta_i\}$ that minimize the loss
- **Answer: The polynomial model of degree *n* that best predicts *y* from *x* is that which has the smallest loss after training**
- **Assumption: We are not overfitting the data by, e.g., training each model on 75% of the data and calculating the final loss on the remaining 25% of the data (i.e., we are using a "validation set")**

Time (*t*)	Number of birds (*x*)	Number of deer (*y*)
9:30 AM	5	3
10:30 AM	2	2
11:30 AM	1	0
⋮	⋮	⋮
4:30 PM	2	1

Let's take stock of what we have

- **The general model we are using is polynomial**
- **We can define a specific type of polynomial model by choosing a fixed value of *n*:**
  - *n*=1 refers to a linear model (one non-trivial term in the model)
  - *n*=2 refers to a quadratic model (two non-trivial terms in the model)
  - etc.
- **In other words, the model is specified by the value of *n***
  - *n* is called a "hyperparameter" of the model
- **For each model (i.e., for each possible value of *n*), we train the model by calculating the weights/parameters (the $\{\theta_i\}$) that minimize the loss**
- **We declare the "best" model as that having the smallest loss**
  - In other words, we have optimized the hyperparameter *n*
  - We have just performed hyperparameter optimization (HPO)
- **Note: Hyperparameters are fixed during the training process, whereas weights/parameters are changing during the training process**

nterms (*n*)	Validation loss
1	3
2	**0.5**
3	1.5
⋮	⋮
10	2.5

Types of HPO: grid search vs. Bayesian search

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

- **In both cases, we are "searching" for the "best" hyperparameter (HP), which is that which minimizes a measure of loss**
- **We have performed a "grid search," in which the minimization metric (i.e., the validation loss) is calculated for every possible value of *n* (in this case, for *n*=1,2,…,10)**
- **Grid search is easy enough when the space of possible HP values is small and the training process is fast**
- **However, if we define a larger HP space (say, *n* could be anywhere from 1 to 100), or if the training process is slow, then performing a full grid search takes significant time and computational resources**
- **A "Bayesian search" would be more efficient:**
  - Sample the HP space a few times, performing training for, say, *n* = 1, 25, 50, 75, 100
  - If, say, smaller values of *n* lead to lower loss, then we would resample the HP space more densely for smaller *n* and more sparsely for larger *n*
  - In this way, we could determine the optimum value of *n* without having sampled the full HP space and performing training for every value, saving precious time and resources
  - The point is we are not naïvely sampling the full HP space, but rather more elegantly sampling the HP space by incorporating knowledge of prior results into the sampling process itself

nterms (*n*)	Validation loss
1	3
2	**0.5**
3	1.5
⋮	⋮
10	2.5

Machine/deep learning commonalities

- **Nearly all machine/deep learning models have hyperparameters that can (and should!) be optimized:**
  - **Polynomial:** number of terms in the polynomial expression
  - **Random forest:** number of decision trees, maximum number of levels in each tree
  - **Neural network:** number of hidden layers, type of activation function
  - **General model:** number of training epochs, learning rate, batch size
- **A measure of loss can always be defined**
  - This allows you to assess different values of HPs within a model type
  - This is true for both supervised and unsupervised learning
- **Bottom line: You can always improve your machine or deep learning models by performing HPO because there are always HPs that can be tweaked**

Part II

Overview of CANDLE

Please fill out the survey: https://bit.ly/CANDLE_Workshop_Jan21

Introduction to CANDLE

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

- **CANDLE is a software platform for performing common machine/deep learning tasks at scale**
  - **CAN**cer **D**istributed **L**earning Environment
- **Developed jointly by the Department of Energy and the National Cancer Institute**
  - Aims to support all computational needs of the JDACS4C pilot projects
  - What we end up with is a powerful open-source software package for performing machine-learning-related tasks on High Performance Computing systems
  - Non-specific to cancer
  - Continuously updated and improved
- **Broken up into "workflows" such as hyperparameter optimization (HPO), uncertainty quantification, weight-sharing, etc.**
- **We have enabled, tested, and supported the HPO functionality of CANDLE on NIH's Biowulf supercomputer**
  - Two of the most popular methods for performing HPO: grid search and Bayesian search
  - Actively supported on Biowulf by Frederick National Lab's Strategic and Data Science Initiatives team

Why you should perform HPO using CANDLE on Biowulf

- **HPO is key to finding the best model for your needs**
- **CANDLE makes performing HPO as easy as possible**
  - Minimal modification to your model
  - Minimal setup process
- **CANDLE is smart**
  - Load balancing – minimal resource downtime
  - Intelligent hyperparameter selection when using the `bayesian` workflow
- **More computational power is easily employed due to CANDLE's integration with Biowulf's resources**
- **Machine/deep learning models can be written in Python, R, or bash**
- **All you need is a Biowulf account; it is ready for you to use right now!**

How to run a sample CANDLE job on Biowulf

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

1. Log in to Biowulf and enter a directory on the `/data` partition (e.g., `/data/$USER/candle`)
2. Load the CANDLE module:

```
module load candle
```

3. Import a sample CANDLE job:

```
candle import-template <grid|bayesian|r|bash>
```

4. Submit the job to Biowulf:

```
candle submit-job <TEMPLATE-NAME>_example.in
```

5. Inspect the results:

```
candle aggregate-results $(pwd)/last-candle-job
```

Summary: How to perform HPO on your own model using CANDLE

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

1. **Ensure your model already works on Biowulf**
2. **Adapt your model to work with CANDLE**
   - Define the hyperparameters in your model using a dictionary (or data.frame) named `candle_params`
   - Assign a metric of model performance to a variable named `candle_value_to_return`
3. **Create a CANDLE input file**
   - This is easiest done by modifying one of the template input files that can be imported using `candle import-template <grid|bayesian|r|bash>`
4. **Confirm that your CANDLE-adapted model and input file settings run using CANDLE without running a full workflow**
5. **Submit the full CANDLE job to Biowulf**
6. **Inspect the results using, e.g., `candle aggregate-results <RESULTS-DIR>`**

(1) Ensure your model already works on Biowulf

- **Acquire a model:**
  - Internet/GitHub
  - Textbook, journal article, or software documentation
  - You already have one
- **Request an interactive node on Biowulf using, e.g.:**

```
sinteractive --gres=gpu:k80:1 --mem=20G
```

- **Run the model using, e.g.:**

```
python my_model_script.py
Rscript my_model_script.R
```

- **See the Biowulf user guide for more information on running scripts on Biowulf**

(2) Adapt your model to work with CANDLE

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

A. Define the hyperparameters in your model by the `candle_params` dictionary (Python) or data.frame (R), e.g.:

Python example

```
tree_depth = 4
nhidden_layers = 5
```

→

```
tree_depth = candle_params['decision_tree_depth']
nhidden_layers = candle_params['nhidden_layers']
```

R example

```
tree_depth <- 4
nhidden_layers <- 5
```

→

```
tree_depth <- candle_params[["decision_tree_depth"]]
nhidden_layers <- candle_params[["nhidden_layers"]]
```

(2) Adapt your model to work with CANDLE ctd.

B. Assign a minimization metric of model performance to the `candle_value_to_return` variable, e.g.:

Python example

```
score = model.evaluate(x_test, y_test)
candle_value_to_return = score[0]
```

R example

```
candle_value_to_return <- my_validation_loss
```

(3) Create a CANDLE input file

- **This is a single text file containing all the settings you need for your CANDLE job**
- **The easiest way to create an input file is to modify one of the templates, which can be imported using**

```
candle import-template <grid|bayesian|r|bash>
```

- **The input file contains three sections:**
  - `&control`: General job settings
  - `&default_model`: Default values of whatever settings are defined using the `candle_params` dictionary in the adapted model script from Step (2)
  - `&param_space`: Specification of the space of the hyperparameter values to sample during the CANDLE workflow
- **See the CANDLE documentation for details on the possible keywords**

```
&control
  model_script="$(pwd)/mnist_mlp.py"
  workflow="grid"
  nworkers=2
  worker_type="k80"
  walltime="00:20:00"
  run_workflow=1
/

&default_model
  epochs=20
  batch_size=128
  activation='relu'
  optimizer='rmsprop'
  num_filters=32
/

&param_space
  {"id": "hpset_01", "epochs": 15, "activation": "tanh"}
  {"id": "hpset_02", "epochs": 30, "activation": "tanh"}
  {"id": "hpset_03", "epochs": 15, "activation": "relu"}
  {"id": "hpset_04", "epochs": 30, "activation": "relu"}
  {"id": "hpset_05", "epochs": 10, "batch_size": 128}
  {"id": "hpset_06", "epochs": 10, "batch_size": 256}
/
```

(3) Create a CANDLE input file ctd.

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

- For a grid search, instead of creating the `&param_space` section by hand, you can create one automatically using:

```
candle generate-grid <PYTHON-LIST-1> \
       <PYTHON-LIST-2> ...
```

- For example:

```
candle generate-grid \
       "['nlayers',np.arange(5,15,2)]" \
       "['dir',['x','y','z']]"
```

- The resulting output in `hyperparameter_grid.txt` in the `candle_generated_files` will look like

```
{"id": "hpset_00001", "nlayers": 5, "dir": "x"}
{"id": "hpset_00002", "nlayers": 5, "dir": "y"}
{"id": "hpset_00003", "nlayers": 5, "dir": "z"}
{"id": "hpset_00004", "nlayers": 7, "dir": "x"}
{"id": "hpset_00005", "nlayers": 7, "dir": "y"}
{"id": "hpset_00006", "nlayers": 7, "dir": "z"}
{"id": "hpset_00007", "nlayers": 9, "dir": "x"}
{"id": "hpset_00008", "nlayers": 9, "dir": "y"}
{"id": "hpset_00009", "nlayers": 9, "dir": "z"}
{"id": "hpset_00010", "nlayers": 11, "dir": "x"}
{"id": "hpset_00011", "nlayers": 11, "dir": "y"}
{"id": "hpset_00012", "nlayers": 11, "dir": "z"}
{"id": "hpset_00013", "nlayers": 13, "dir": "x"}
{"id": "hpset_00014", "nlayers": 13, "dir": "y"}
{"id": "hpset_00015", "nlayers": 13, "dir": "z"}
```

- Then copy-paste these settings into the `&param_space` section
- As the example shows, you can access Python's NumPy library through `np`

(3) Create a CANDLE input file ctd.

- To run a Bayesian search, change the `workflow` setting to `bayesian`
- Change the `&param_space` section of the input file to something like:

```
&param_space
  makeDiscreteParam("batch_size", values = c(16, 32))
  makeIntegerParam("epochs", lower = 2, upper = 5)
  makeDiscreteParam("optimizer", values = c("adam", "sgd", "rmsprop", "adagrad"))
  makeNumericParam("drop", lower = 0, upper = 0.9)
  makeNumericParam("learning_rate", lower = 0.00001, upper = 0.1)
/
```

(4) Confirm that your CANDLE-adapted model and input file settings run using CANDLE without running a full workflow

- **To do this, run the model once using the default set of hyperparameters set in the `&default_model` section of the input file:**
  - Allocate an interactive node, e.g.:

    ```
    sinteractive --gres=gpu:k80:1 --mem=20G
    ```

  - Set the `run_workflow` keyword to `0` in `&control` section of the input file
  - Run CANDLE using:

    ```
    candle submit-job <INPUT-FILE>
    ```

- **As long as the last message before the prompt is returned is `Input file submitted successfully` and the output in the new file `subprocess_out_and_err.txt` looks reasonable, you can be almost guaranteed that running an actual CANDLE workflow will work as well!**

(5) Submit the full CANDLE job to Biowulf

- **Set the `run_workflow` keyword to `1` in `&control` section of the input file**
- **Run CANDLE using, again:**

```
candle submit-job <INPUT-FILE>
```

- **You will know you have submitted the CANDLE job successfully to Biowulf if you see, at the end of the output to the screen:**

```
JOB_ID=<YOUR-SLURM-JOB-ID>
Input file submitted successfully
```

- **You can monitor the status of the job using, e.g.:**

```
watch squeue -u <YOUR-BIOWULF-USERNAME>
```

(6) Inspect the results of the CANDLE job – manual inspection

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

- Once the job is complete, enter the directory `last-candle-job`, which is a symbolic link
- Ensure the content of the file output.txt looks reasonable, and in particular that it ends with something like:

```
MPIEXEC TIME: 263.763
EXIT CODE: 0
COMPLETE: 2021-01-14 18:56:51
```

- Then enter the `run` directory, which contains one subdirectory per hyperparameter set run on the model
- Skim through the output of each run of the model using, e.g.,

```
less */subprocess_out_and_err.txt
```

- In each hyperparameter subdirectory, there should be a file called `result.txt` containing the value of the minimization metric you defined in step (2) (`candle_value_to_return`)

(6) Inspect the results of the CANDLE job – automated inspection

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

- **Alternatively, once the job is complete, collect all the minimization metrics and values of the hyperparameters into a a single file using:**

```
candle aggregate-results $(pwd)/last-candle-job
```

- **This will create a file called `candle_results.csv` in the `candle_generated_files` directory that contains the metrics (in increasing order) and their corresponding hyperparameter values, e.g.:**

```
result,dirname,id,mincorr,maxcorr,number_cv,extfolds
000.796,hpset_00001,hpset_00001,0.200000,0.80,2,5
000.796,hpset_00004,hpset_00004,0.200000,0.80,5,5
000.837,hpset_00002,hpset_00002,0.200000,0.80,3,5
000.878,hpset_00003,hpset_00003,0.200000,0.80,4,5
000.905,hpset_00007,hpset_00007,0.200000,0.80,8,5
000.964,hpset_00005,hpset_00005,0.200000,0.80,6,5
000.964,hpset_00006,hpset_00006,0.200000,0.80,7,5
001.000,hpset_00008,hpset_00008,0.200000,0.80,9,5
```

- **This file can, e.g., be imported into Excel to for more careful observation and analysis such as Pearson correlation**

Part III

Walk-through: Running a grid search using CANDLE from scratch

Please fill out the survey: https://bit.ly/CANDLE_Workshop_Jan21

Introduction to the model

- **An autoencoder is a type of unsupervised deep learning algorithm**
  - It reduces the dimensionality of the inputs and then reconstructs them, calling the error in the reconstruction process the “reconstruction loss”; this is the loss we want to minimize
- **A <u>variational</u> autoencoder (VAE) is a type of autoencoder that is better capable of “generating” new samples (it is a “generative” model) by defining a continuous latent space**
  - It is pretty cutting-edge
- **PyTorch is a deep learning library written in Python and developed by Facebook**
  - It is a popular alternative to the Python deep learning library TensorFlow/Keras developed by Google
- **We will optimize some hyperparameters of a PyTorch-based VAE using CANDLE on Biowulf, with the usual goal of finding the hyperparameters that minimize the reconstruction loss**

(1) Ensure the model already works on Biowulf

- **Create and enter a working directory on Biowulf's data partition:**

```
mkdir /data/$USER/vae_with_pytorch
cd /data/$USER/vae_with_pytorch
```

- **Clone PyTorch's "examples" repository from GitHub and enter the directory for the VAE example:**

```
git clone https://github.com/pytorch/examples.git repo
cd repo/vae
```

- **Test the example:**

```
sinteractive --gres=gpu:k80:1 --mem=20G
module load python
python main.py
```

(1) Ensure the model already works on a Biowulf compute node ctd.

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

- **Output to screen (after a couple minutes):**

```
weismanal@cn4227:/data/weismanal/vae_with_pytorch/repo/vae $ python main.py
Downloading http://yann.lecun.com/exdb/mnist/train-images-idx3-ubyte.gz to .
9920512it [00:00, 24625593.18it/s]
Extracting ../data/MNIST/raw/train-images-idx3-ubyte.gz to ../data/MNIST/raw
Downloading http://yann.lecun.com/exdb/mnist/train-labels-idx1-ubyte.gz to .
32768it [00:00, 382718.91it/s]
Extracting ../data/MNIST/raw/train-labels-idx1-ubyte.gz to ../data/MNIST/raw
Downloading http://yann.lecun.com/exdb/mnist/t10k-images-idx3-ubyte.gz to ..
1654784it [00:00, 10500316.42it/s]
Extracting ../data/MNIST/raw/t10k-images-idx3-ubyte.gz to ../data/MNIST/raw
Downloading http://yann.lecun.com/exdb/mnist/t10k-labels-idx1-ubyte.gz to ..
8192it [00:00, 203787.16it/s]
Extracting ../data/MNIST/raw/t10k-labels-idx1-ubyte.gz to ../data/MNIST/raw
Processing...
/usr/local/Anaconda/envs/py3.7/lib/python3.7/site-packages/torchvision/datas
nsor. You may want to copy the array to protect its data or make it writeabl
.)
  return torch.from_numpy(parsed.astype(m[2], copy=False)).view(*s)
Done!
Train Epoch: 1 [0/60000 (0%)]   Loss: 550.800781
Train Epoch: 1 [1280/60000 (2%)]        Loss: 322.319275
Train Epoch: 1 [2560/60000 (4%)]        Loss: 238.075089
Train Epoch: 1 [3840/60000 (6%)]        Loss: 217.480148
Train Epoch: 1 [5120/60000 (9%)]        Loss: 207.814041
Train Epoch: 1 [6400/60000 (11%)]       Loss: 210.372253
Train Epoch: 1 [7680/60000 (13%)]       Loss: 206.411575
Train Epoch: 1 [8960/60000 (15%)]       Loss: 198.051224
Train Epoch: 1 [10240/60000 (17%)]      Loss: 196.847107
Train Epoch: 1 [11520/60000 (19%)]      Loss: 188.627228
Train Epoch: 1 [12800/60000 (21%)]      Loss: 184.369156
Train Epoch: 1 [14080/60000 (23%)]      Loss: 183.649902
Train Epoch: 1 [15360/60000 (26%)]      Loss: 175.880432
Train Epoch: 1 [16640/60000 (28%)]      Loss: 163.250214
```

```
Train Epoch: 10 [23040/60000 (38%)]     Loss: 105.889717
Train Epoch: 10 [24320/60000 (41%)]     Loss: 107.185249
Train Epoch: 10 [25600/60000 (43%)]     Loss: 107.920563
Train Epoch: 10 [26880/60000 (45%)]     Loss: 109.149643
Train Epoch: 10 [28160/60000 (47%)]     Loss: 109.661118
Train Epoch: 10 [29440/60000 (49%)]     Loss: 109.378181
Train Epoch: 10 [30720/60000 (51%)]     Loss: 107.560875
Train Epoch: 10 [32000/60000 (53%)]     Loss: 107.892548
Train Epoch: 10 [33280/60000 (55%)]     Loss: 103.155792
Train Epoch: 10 [34560/60000 (58%)]     Loss: 104.760223
Train Epoch: 10 [35840/60000 (60%)]     Loss: 104.411865
Train Epoch: 10 [37120/60000 (62%)]     Loss: 106.796677
Train Epoch: 10 [38400/60000 (64%)]     Loss: 105.444489
Train Epoch: 10 [39680/60000 (66%)]     Loss: 104.048927
Train Epoch: 10 [40960/60000 (68%)]     Loss: 105.839996
Train Epoch: 10 [42240/60000 (70%)]     Loss: 104.069458
Train Epoch: 10 [43520/60000 (72%)]     Loss: 107.286446
Train Epoch: 10 [44800/60000 (75%)]     Loss: 106.501312
Train Epoch: 10 [46080/60000 (77%)]     Loss: 107.803185
Train Epoch: 10 [47360/60000 (79%)]     Loss: 106.015686
Train Epoch: 10 [48640/60000 (81%)]     Loss: 104.442245
Train Epoch: 10 [49920/60000 (83%)]     Loss: 107.585327
Train Epoch: 10 [51200/60000 (85%)]     Loss: 108.005600
Train Epoch: 10 [52480/60000 (87%)]     Loss: 102.704224
Train Epoch: 10 [53760/60000 (90%)]     Loss: 103.745224
Train Epoch: 10 [55040/60000 (92%)]     Loss: 107.164246
Train Epoch: 10 [56320/60000 (94%)]     Loss: 107.190544
Train Epoch: 10 [57600/60000 (96%)]     Loss: 107.507843
Train Epoch: 10 [58880/60000 (98%)]     Loss: 103.408936
====> Epoch: 10 Average loss: 106.3941
====> Test set loss: 105.6771
```

(2) Adapt the model to work with CANDLE

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

A. Define the hyperparameters in `main.py` using the `candle_params` dictionary:

```
weismanal@cn4227:/data/weismanal/vae_with_pytorch/repo/vae $ git diff
diff --git a/vae/main.py b/vae/main.py
index d7df336..17c6ba4 100644
--- a/vae/main.py
+++ b/vae/main.py
@@ -9,11 +9,11 @@ from torchvision.utils import save_image


 parser = argparse.ArgumentParser(description='VAE MNIST Example')
-parser.add_argument('--batch-size', type=int, default=128, metavar='N',
+parser.add_argument('--batch-size', type=int, default=candle_params['batch_size'], metavar='N',
                     help='input batch size for training (default: 128)')
-parser.add_argument('--epochs', type=int, default=10, metavar='N',
+parser.add_argument('--epochs', type=int, default=candle_params['epochs'], metavar='N',
                     help='number of epochs to train (default: 10)')
-parser.add_argument('--no-cuda', action='store_true', default=False,
+parser.add_argument('--no-cuda', action='store_true', default=candle_params['no_cuda'],
                     help='disables CUDA training')
 parser.add_argument('--seed', type=int, default=1, metavar='S',
                     help='random seed (default: 1)')
```

(2) Adapt the model to work with CANDLE ctd.

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

B. Assign a minimization metric of model performance to the `candle_value_to_return` variable in `main.py`:

```
weismanal@cn4227:/data/weismanal/vae_with_pytorch/repo/vae $ git diff
diff --git a/vae/main.py b/vae/main.py
index 17c6ba4..d9f3bba 100644
--- a/vae/main.py
+++ b/vae/main.py
@@ -120,11 +120,12 @@ def test(epoch):

     test_loss /= len(test_loader.dataset)
     print('====> Test set loss: {:.4f}'.format(test_loss))
+    return(test_loss)

 if __name__ == "__main__":
     for epoch in range(1, args.epochs + 1):
         train(epoch)
-        test(epoch)
+        candle_value_to_return = test(epoch)
         with torch.no_grad():
             sample = torch.randn(64, 20).to(device)
             sample = model.decode(sample).cpu()
```

(2) Adapt the model to work with CANDLE ctd.

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

- **Make some additional minor changes due to how this model script was written**
- **Note: We quickly discovered the need for these changes by performing Step (4) (running CANDLE without running a full workflow)**

```
weismanal@cn4227:/data/weismanal/vae_with_pytorch/repo/vae $ git diff
diff --git a/vae/main.py b/vae/main.py
index d9f3bba..1e01b0f 100644
--- a/vae/main.py
+++ b/vae/main.py
@@ -1,4 +1,4 @@
-from __future__ import print_function
+#from __future__ import print_function
 import argparse
 import torch
 import torch.utils.data
@@ -7,6 +7,10 @@ from torch.nn import functional as F
 from torchvision import datasets, transforms
 from torchvision.utils import save_image

+import os
+os.makedirs('data', exist_ok=True)
+os.makedirs('results', exist_ok=True)
+

 parser = argparse.ArgumentParser(description='VAE MNIST Example')
 parser.add_argument('--batch-size', type=int, default=candle_params['batch_size'], metavar='N',
@@ -28,11 +32,11 @@ device = torch.device("cuda" if args.cuda else "cpu")

 kwargs = {'num_workers': 1, 'pin_memory': True} if args.cuda else {}
 train_loader = torch.utils.data.DataLoader(
-    datasets.MNIST('../data', train=True, download=True,
+    datasets.MNIST('data', train=True, download=True,
                    transform=transforms.ToTensor()),
     batch_size=args.batch_size, shuffle=True, **kwargs)
 test_loader = torch.utils.data.DataLoader(
-    datasets.MNIST('../data', train=False, transform=transforms.ToTensor()),
+    datasets.MNIST('data', train=False, transform=transforms.ToTensor()),
     batch_size=args.batch_size, shuffle=True, **kwargs)
```

(3) Create a CANDLE input file

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

- **Import the grid search template:**

```
mkdir /data/$USER/vae_with_pytorch/candle_job
cd /data/$USER/vae_with_pytorch/candle_job
module purge
module load candle
candle import-template grid
rm mnist_mlp.py
mv grid_example.in vae.in
ls /data/$USER/vae_with_pytorch/*
```

```
weismanal@cn4227:/data/weismanal/vae_with_pytorch/candle_job $ ls /data/$USER/vae_with_pytorch/*
/data/weismanal/vae_with_pytorch/candle_job:
vae.in

/data/weismanal/vae_with_pytorch/repo:
cpp                imagenet       regression              time_sequence_prediction
data               LICENSE        reinforcement_learning  vae
dcgan              mnist          run_python_examples.sh  word_language_model
distributed        mnist_hogwild  snli
fast_neural_style  README.md      super_resolution
```

(3) Create a CANDLE input file ctd.

- **Modify the input file from the grid search template**

- **Note that even though we don't vary `no_cuda` in the `&param_space` section, CANDLE knows what value to use since it is defined in the `&default_model` section**

```
weismanal@cn4227:/data/weismanal/vae_with_pytorch/candle_job $ git diff
diff --git a/vae.in b/vae.in
index 5c89655..33dd53b 100644
--- a/vae.in
+++ b/vae.in
@@ -1,25 +1,23 @@
 &control
-  model_script="$(pwd)/mnist_mlp.py"
+  model_script = /data/$USER/vae_with_pytorch/repo/vae/main.py
   workflow="grid"
-  nworkers=2
+  nworkers=3
   worker_type="k80"
-  walltime="00:20:00"
+  walltime = 00:10:00
   run_workflow=1
 /

 &default_model
-  epochs=20
-  batch_size=128
-  activation='relu'
-  optimizer='rmsprop'
-  num_filters=32
+  batch_size = 256
+  epochs = 5
+  no_cuda = False
 /

 &param_space
-  {"id": "hpset_01", "epochs": 15, "activation": "tanh"}
-  {"id": "hpset_02", "epochs": 30, "activation": "tanh"}
-  {"id": "hpset_03", "epochs": 15, "activation": "relu"}
-  {"id": "hpset_04", "epochs": 30, "activation": "relu"}
-  {"id": "hpset_05", "epochs": 10, "batch_size": 128}
-  {"id": "hpset_06", "epochs": 10, "batch_size": 256}
+  {"id": "hpset_00001", "batch_size": 128, "epochs": 5}
+  {"id": "hpset_00002", "batch_size": 128, "epochs": 10}
+  {"id": "hpset_00003", "batch_size": 256, "epochs": 5}
+  {"id": "hpset_00004", "batch_size": 256, "epochs": 10}
+  {"id": "hpset_00005", "batch_size": 512, "epochs": 5}
+  {"id": "hpset_00006", "batch_size": 512, "epochs": 10}
 /
```

(3) Create a CANDLE input file ctd.

- **Note that we generated the `&param_space` section using:**

```
candle generate-grid "['batch_size',[128,256,512]]" \
      "['epochs',[5,10]]"
```

```
weismanal@cn4227:/data/weismanal/vae_with_pytorch/candle_job $ candle generate-grid "['batch_size',[128,256,512]]" "['epochs',[5,10]]"
Generating hyperparameter grid into file "hyperparameter_grid.txt"... [+] Loading python 3.7  ...
done
weismanal@cn4227:/data/weismanal/vae_with_pytorch/candle_job $ cat candle_generated_files/hyperparameter_grid.txt
{"id": "hpset_00001", "batch_size": 128, "epochs": 5}
{"id": "hpset_00002", "batch_size": 128, "epochs": 10}
{"id": "hpset_00003", "batch_size": 256, "epochs": 5}
{"id": "hpset_00004", "batch_size": 256, "epochs": 10}
{"id": "hpset_00005", "batch_size": 512, "epochs": 5}
{"id": "hpset_00006", "batch_size": 512, "epochs": 10}
```

(4) Run the model using CANDLE without running a workflow

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

- **Set `run_workflow=0` and run from an interactive node: `candle submit-job vae.in`:**

```
weismanal@cn4227:/data/weismanal/vae_with_pytorch/candle_job $ candle submit-job vae.in
Submitting the CANDLE input file "vae.in"...
[+] Loading python 3.7  ...
Possible keywords and their default values:

  model_script:           None
  workflow:               None
  walltime:               00:05:00
  worker_type:            k80
  nworkers:               1
  nthreads:               1
  custom_sbatch_args:
  mem_per_cpu:            7
  dl_backend:             keras
  supp_modules:
  python_bin_path:
  exec_python_module:
  supp_pythonpath:
  extra_script_args:
  exec_r_module:
  supp_r_libs:
  run_workflow:           1
  dry_run:                0
  default_model_file:
  param_space_file:
NOTE: Keyword "model_script" has a valid value of /data/weismanal/vae_with_pytorch/repo/
vae/main.py
NOTE: Keyword "workflow" has a valid value of grid
WARNING: No error-checking done on "walltime" keyword
NOTE: Keyword "walltime" has a valid value of 00:10:00
NOTE: Keyword "worker_type" has a valid value of k80
```

```
/data/BIDS-HPC/public/software/distributions/candle/main/Benchmarks/common/default_utils
.py:408: RuntimeWarning: These keywords used in the configuration file are not defined i
n CANDLE: ['no_cuda']
  warnings.warn(message, RuntimeWarning)
Params:
{'batch_size': 256,
 'data_type': <class 'numpy.float32'>,
 'epochs': 5,
 'experiment_id': 'EXP000',
 'logfile': None,
 'no_cuda': False,
 'output_dir': '/gpfs/gsfs10/users/weismanal/vae_with_pytorch/candle_job/Output/EXP000/R
UN000',
 'profiling': False,
 'rng_seed': 7102,
 'run_id': 'RUN000',
 'shuffle': False,
 'timeout': -1,
 'train_bool': True,
 'verbose': None}
Params: {'batch_size': 256, 'epochs': 5, 'no_cuda': False, 'verbose': None, 'logfile': N
one, 'train_bool': True, 'experiment_id': 'EXP000', 'run_id': 'RUN000', 'shuffle': False
, 'profiling': False, 'data_type': "<class 'numpy.float32'>", 'output_dir': '/gpfs/gsfs1
0/users/weismanal/vae_with_pytorch/candle_job/Output/EXP000/RUN000', 'rng_seed': 7102, '
timeout': -1}
Starting run of model_wrapper.sh from candle_compliant_wrapper.py...
Finished run of model_wrapper.sh from candle_compliant_wrapper.py
Input file submitted successfully
weismanal@cn4227:/data/weismanal/vae_with_pytorch/candle_job $ ls
candle_generated_files       params.json                      vae.in
candle_value_to_return.json  results                          wrapped_model.py
data                         run_candle_model_standalone.sh
Output                       subprocess_out_and_err.txt
```

(4) Run the model using CANDLE without running a workflow ctd.

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

- **Review the contents of `subprocess_out_and_err.txt`:**

```
weismanal@cn4227:/data/weismanal/vae_with_pytorch/candle_job $ cat subprocess_out_and_err.txt
MODEL_WRAPPER.SH START TIME: 1611021785
HOST: cn4227
GPU: 0
[-] Unloading python 3.7  ...
[+] Loading candle main  ...
[+] Loading python 3.7  ...
Using Python for execution: /usr/local/Anaconda/envs/py3.7/bin/python
Andrew adding this as a check of the 'device' setting: cuda
Train Epoch: 1 [0/60000 (0%)]   Loss: 550.741028
Train Epoch: 1 [2560/60000 (4%)]        Loss: 304.129547
Train Epoch: 1 [5120/60000 (9%)]        Loss: 234.225998
Train Epoch: 1 [7680/60000 (13%)]       Loss: 222.815308
Train Epoch: 1 [10240/60000 (17%)]      Loss: 213.785507
Train Epoch: 1 [12800/60000 (21%)]      Loss: 206.170288
Train Epoch: 1 [15360/60000 (26%)]      Loss: 207.267151
Train Epoch: 1 [17920/60000 (30%)]      Loss: 199.061737
Train Epoch: 1 [20480/60000 (34%)]      Loss: 192.688736
Train Epoch: 1 [23040/60000 (38%)]      Loss: 188.461182
```

```
Train Epoch: 4 [56320/60000 (94%)]      Loss: 115.278954
Train Epoch: 4 [58880/60000 (98%)]      Loss: 114.284561
====> Epoch: 4 Average loss: 116.2631
====> Test set loss: 113.6888
Train Epoch: 5 [0/60000 (0%)]   Loss: 112.970299
Train Epoch: 5 [2560/60000 (4%)]        Loss: 115.350601
Train Epoch: 5 [5120/60000 (9%)]        Loss: 117.025635
Train Epoch: 5 [7680/60000 (13%)]       Loss: 114.819489
Train Epoch: 5 [10240/60000 (17%)]      Loss: 114.006920
Train Epoch: 5 [12800/60000 (21%)]      Loss: 116.714493
Train Epoch: 5 [15360/60000 (26%)]      Loss: 112.948586
Train Epoch: 5 [17920/60000 (30%)]      Loss: 113.665970
Train Epoch: 5 [20480/60000 (34%)]      Loss: 113.379929
Train Epoch: 5 [23040/60000 (38%)]      Loss: 112.894157
Train Epoch: 5 [25600/60000 (43%)]      Loss: 110.664818
Train Epoch: 5 [28160/60000 (47%)]      Loss: 112.720894
Train Epoch: 5 [30720/60000 (51%)]      Loss: 112.036499
Train Epoch: 5 [33280/60000 (55%)]      Loss: 115.482460
Train Epoch: 5 [35840/60000 (60%)]      Loss: 110.831451
Train Epoch: 5 [38400/60000 (64%)]      Loss: 114.881470
Train Epoch: 5 [40960/60000 (68%)]      Loss: 111.105072
Train Epoch: 5 [43520/60000 (72%)]      Loss: 114.555351
Train Epoch: 5 [46080/60000 (77%)]      Loss: 113.871315
Train Epoch: 5 [48640/60000 (81%)]      Loss: 111.388084
Train Epoch: 5 [51200/60000 (85%)]      Loss: 114.239548
Train Epoch: 5 [53760/60000 (89%)]      Loss: 111.347534
Train Epoch: 5 [56320/60000 (94%)]      Loss: 114.522316
Train Epoch: 5 [58880/60000 (98%)]      Loss: 111.100021
====> Epoch: 5 Average loss: 113.4483
====> Test set loss: 111.4594
MODEL_WRAPPER.SH END TIME: 1611021832
```

Everything looks reasonable; and we can be reasonably confident that the full CANDLE job will work as expected!

(5) Submit the full CANDLE job

- **Clean up the working directory:**

```
mkdir no_workflow
mv * no_workflow/
cp no_workflow/vae.in .
```

- **Set** `run_workflow=1` **and run** `candle submit-job vae.in`

(5) Submit the full CANDLE job ctd.

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

- **Output to the screen:**

```
weismanal@cn4227:/data/weismanal/vae_with_pytorch/candle_job $ candle submit-job vae.in
Submitting the CANDLE input file "vae.in"...
[+] Loading python 3.7  ...
Possible keywords and their default values:

  model_script:           None
  workflow:               None
  walltime:               00:05:00
  worker_type:            k80
  nworkers:               1
  nthreads:               1
  custom_sbatch_args:
  mem_per_cpu:            7
  dl_backend:             keras
  supp_modules:
  python_bin_path:
  exec_python_module:
  supp_pythonpath:
  extra_script_args:
  exec_r_module:
  supp_r_libs:
  run_workflow:           1
  dry_run:                0
  default_model_file:
  param_space_file:
NOTE: Keyword "model_script" has a valid value of /data/weismanal/vae_with_pytorch/repo/
vae/main.py
NOTE: Keyword "workflow" has a valid value of grid
WARNING: No error-checking done on "walltime" keyword
NOTE: Keyword "walltime" has a valid value of 00:10:00
NOTE: Keyword "worker_type" has a valid value of k80
NOTE: Keyword "nworkers" has a valid value of 3
```

```
mkdir: created directory '/gpfs/gsfs10/users/weismanal/vae_with_pytorch/candle_job/candl
e_generated_files/experiments/X000/run'
/usr/local/OpenMPI/4.0.4/CUDA-10.2/gcc-9.2.0/bin/mpicc
/data/BIDS-HPC/public/software/distributions/candle/main/swift-t-install/stc/bin/swift-t

Currently Loaded Modules:
  1) candle/main   4) openmpi/4.0.4/cuda-10.2/gcc-9.2.0   7) pcre2/10.21
  2) python/3.7    5) ant/1.10.3                          8) GSL/2.6_gcc-9.2.0
  3) gcc/9.2.0     6) java/1.8.0_211



'/gpfs/gsfs10/users/weismanal/vae_with_pytorch/candle_job/candle_generated_files/grid_wo
rkflow.txt' -> '/gpfs/gsfs10/users/weismanal/vae_with_pytorch/candle_job/candle_generate
d_files/experiments/X000/grid_workflow.txt'
WARN  obj_app.swift:13:3: variable called turbine_output already defined at swift-t-work
flow.mhQ.swift:53
WARN  obj_app.swift:36:3: variable called turbine_output already defined at swift-t-work
flow.mhQ.swift:53
TURBINE-SLURM SCRIPT
NODES=2
PROCS=4
PPN=2
TURBINE_OUTPUT=/gpfs/gsfs10/users/weismanal/vae_with_pytorch/candle_job/candle_generated
_files/experiments/X000
TURBINE_HOME=/data/BIDS-HPC/public/software/distributions/candle/2020-11-23/swift-t-inst
all/turbine
wrote: /gpfs/gsfs10/users/weismanal/vae_with_pytorch/candle_job/candle_generated_files/e
xperiments/X000/turbine-slurm.sh
JOB_ID=6393706
Input file submitted successfully
```

(6) Inspect the job results

- **After the job completes, observe the final directory structure:**

```
weismanal@biowulf:/data/weismanal/vae_with_pytorch/candle_job $ ls *
vae.in

candle_generated_files:
default_model.txt  grid_workflow.txt  preprocessed_vars_to_export.sh
experiments        metadata.json      submit_candle_job.sh

last-candle-job:
cfg-sys-biowulf.sh  output.txt  turbine.log       workflow.sh.log
grid_workflow.txt   run         turbine-slurm.sh  workflow.tic
jobid.txt           submit.sh   vae.in

no_workflow:
candle_generated_files       results
candle_value_to_return.json  run_candle_model_standalone.sh
data                         subprocess_out_and_err.txt
Output                       vae.in
params.json                  wrapped_model.py
```

- `last-candle-job` **is an always-updated symbolic link**

(6) Inspect the job results ctd.

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

- **Enter `last-candle-job` directory and observe the contents of `output.txt`:**

```
weismanal@biowulf:/data/weismanal/vae_with_pytorch/candle_job/last-candle-job $ cat output.txt
stdin: is not a tty
TURBINE-SLURM.SH
START: 2021-01-18 21:27:58

+ /usr/local/slurm/bin/srun --ntasks=4 --distribution=cyclic --mpi=pmix --mem=0 /data/BIDS-HPC/p
ublic/software/builds/tcl/bin/tclsh8.6 /gpfs/gsfs10/users/weismanal/vae_with_pytorch/candle_job/
candle_generated_files/experiments/X000/workflow.tic -expid=X000 -benchmark_timeout=3600 -f=/gpf
s/gsfs10/users/weismanal/vae_with_pytorch/candle_job/candle_generated_files/grid_workflow.txt
srun: Warning: can't honor --ntasks-per-node set to 2 which doesn't match the requested tasks 4
with the number of requested nodes 3. Ignoring --ntasks-per-node.
--------------------------------------------------------------------------
WARNING: There is at least non-excluded one OpenFabrics device found,
but there are no active ports detected (or Open MPI was unable to use
them).  This is most certainly not what you wanted.  Check your
cables, subnet manager configuration, etc.  The openib BTL will be
ignored for this job.

  Local host: cn4175
--------------------------------------------------------------------------
--------------------------------------------------------------------------
WARNING: There is at least non-excluded one OpenFabrics device found,
but there are no active ports detected (or Open MPI was unable to use
them).  This is most certainly not what you wanted.  Check your
cables, subnet manager configuration, etc.  The openib BTL will be
ignored for this job.

  Local host: cn4176
--------------------------------------------------------------------------
--------------------------------------------------------------------------
WARNING: There is at least non-excluded one OpenFabrics device found,
```

```
result(hpset_00002): 105.67711689453125
108.63141457519531;105.67711689453125;111.45938181152344;106.94484123535156;117.5833169921875;10
9.8838830078125
  230.397 turbine finalizing
  230.397 turbine finalizing
  230.397 turbine finalizing
  230.397 ADLB_Finalize_Cmd() start
  230.397 ADLB_Finalize_Cmd() start
turbine finalizing at: 230.397
  230.397 ADLB_Finalize_Cmd() start
  230.398 MPI_Finalize start
  230.398 MPI_Finalize start
  230.397 MPI_Finalize start
ADLB_DEBUG_RANKS: rank: 3 nodename: cn4174
ADLB_PAR_MOD: 1
ADLB Total Elapsed Time: 230.401
  230.379 turbine finalizing
  230.379 ADLB_Finalize_Cmd() start
  230.390 MPI_Finalize start
  230.474 MPI_Finalize stop
  230.474 ADLB_Finalize_Cmd() stop
  230.475 MPI_Finalize stop
  230.475 ADLB_Finalize_Cmd() stop
  230.456 MPI_Finalize stop
  230.456 ADLB_Finalize_Cmd() stop
  230.479 MPI_Finalize stop
  230.479 ADLB_Finalize_Cmd() stop

MPIEXEC TIME: 233.405
EXIT CODE: 0
COMPLETE: 2021-01-18 21:31:51
```

(6) Inspect the job results ctd.

- Enter `run` directory and observe the directory structure

- See the final reconstruction loss on the test set for each hyperparameter set:

```
weismanal@biowulf:/data/weismanal/vae_with_pytorch/candle_job/las
t-candle-job/run $ cat */result.txt
108.63141457519531
105.67711689453125
111.45938181152344
106.94484123535156
117.5833169921875
109.8838830078125
```

```
weismanal@biowulf:/data/weismanal/vae_with_pytorch/candle_job/last-candle-job/run $ ls *
hpset_00001:
candle_value_to_return.json  parameters.txt  run_candle_model_standalone.sh
data                         params.json     subprocess_out_and_err.txt
history.txt                  rank.txt        wrapped_model.py
model.log                    results
Output                       result.txt

hpset_00002:
candle_value_to_return.json  parameters.txt  run_candle_model_standalone.sh
data                         params.json     subprocess_out_and_err.txt
history.txt                  rank.txt        wrapped_model.py
model.log                    results
Output                       result.txt

hpset_00003:
candle_value_to_return.json  parameters.txt  run_candle_model_standalone.sh
data                         params.json     subprocess_out_and_err.txt
history.txt                  rank.txt        wrapped_model.py
model.log                    results
Output                       result.txt

hpset_00004:
candle_value_to_return.json  parameters.txt  run_candle_model_standalone.sh
data                         params.json     subprocess_out_and_err.txt
history.txt                  rank.txt        wrapped_model.py
model.log                    results
Output                       result.txt

hpset_00005:
candle_value_to_return.json  parameters.txt  run_candle_model_standalone.sh
data                         params.json     subprocess_out_and_err.txt
history.txt                  rank.txt        wrapped_model.py
model.log                    results
Output                       result.txt

hpset_00006:
candle_value_to_return.json  parameters.txt  run_candle_model_standalone.sh
data                         params.json     subprocess_out_and_err.txt
history.txt                  rank.txt        wrapped_model.py
model.log                    results
Output                       result.txt
```

(6) Inspect the job results ctd.

Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

- **Enter the submission directory again and generate and observe the aggregated results:**

```
weismanal@biowulf:/data/weismanal/vae_with_pytorch/candle_job $ candle aggregate-results $(pwd)/last-candle-job
Aggregating results from experiment directory "/data/weismanal/vae_with_pytorch/candle_job/last-candle-job" using
 result format "%07.3f" into file "candle_results.csv"... done
weismanal@biowulf:/data/weismanal/vae_with_pytorch/candle_job $ cat candle_generated_files/candle_results.csv
result,dirname,id,batch_size,epochs
105.677,/data/weismanal/vae_with_pytorch/candle_job/last-candle-job/run/hpset_00002,hpset_00002,128,10
106.945,/data/weismanal/vae_with_pytorch/candle_job/last-candle-job/run/hpset_00004,hpset_00004,256,10
108.631,/data/weismanal/vae_with_pytorch/candle_job/last-candle-job/run/hpset_00001,hpset_00001,128,5
109.884,/data/weismanal/vae_with_pytorch/candle_job/last-candle-job/run/hpset_00006,hpset_00006,512,10
111.459,/data/weismanal/vae_with_pytorch/candle_job/last-candle-job/run/hpset_00003,hpset_00003,256,5
117.583,/data/weismanal/vae_with_pytorch/candle_job/last-candle-job/run/hpset_00005,hpset_00005,512,5
```

These results, which are sorted by increasing test loss, preliminarily show that a combination of low batch size and high number of epochs results in lower test loss

Wrap-up

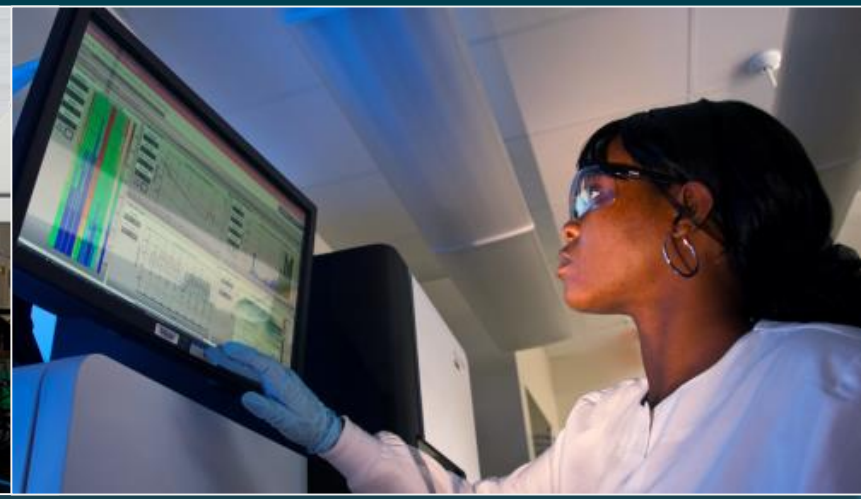
Frederick National Laboratory for Cancer Research
sponsored by the National Cancer Institute

- **Appreciation:**
  - **SDSI team:** George Zaki, Ravi Ravichandran, Lynn Borkon, Petrina Hollingsworth
  - **DOE teams:** Argonne, Los Alamos, Oak Ridge National Labs
  - **Biowulf staff:** Tim Miller, Wolfgang Resch, Susan Chacko
  - **IT support:** Michael Rinaldi, Jose Aragon
- **Links:**
  - **Presentation survey: https://bit.ly/CANDLE_Workshop_Jan21**
  - CANDLE on Biowulf documentation
  - Biowulf user guide
  - This very talk
  - My contact info: andrew.weisman@nih.gov
- **Questions?**

Frederick National Laboratory
for Cancer Research

sponsored by the National Cancer Institute





Presentation Title

Presenter
Title, Affiliation

Date 1, 2018

Secondary Slide Title

- **Bullet point**
- **Bullet point**
- **Bullet point**

Section Title

Presenter