Deep Learning Lab

Exercise 4:Hyperparameter Optimization

Manav Madan (02.01.2019)

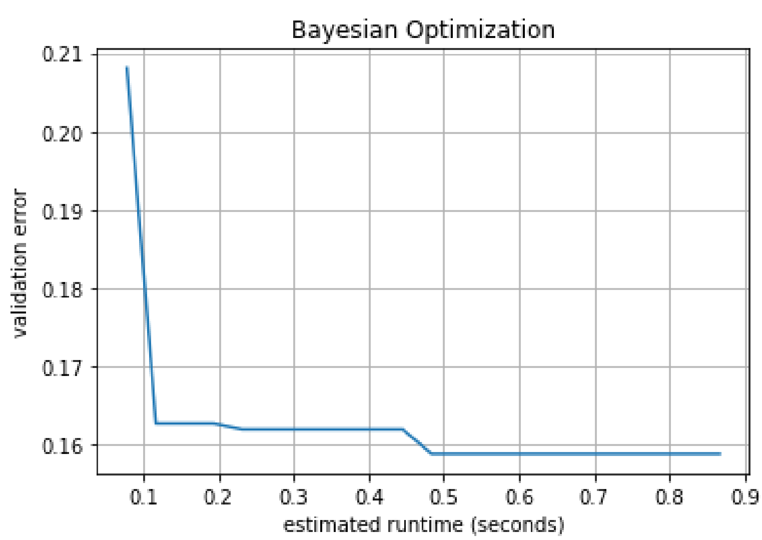
Introduction:

In this exercise we would going through Bayesian optimization, Hypendand and combination of both of hyperparameter optimization(BOHB) techniques. Instead of Evaluating Neural Network directly, we would be using surrogate benchmark(regression-model) as random forest to evaluate the next hyperparameter configuration set instead of running our CNN for evaluating the new loss.

For bayesian Optimization:

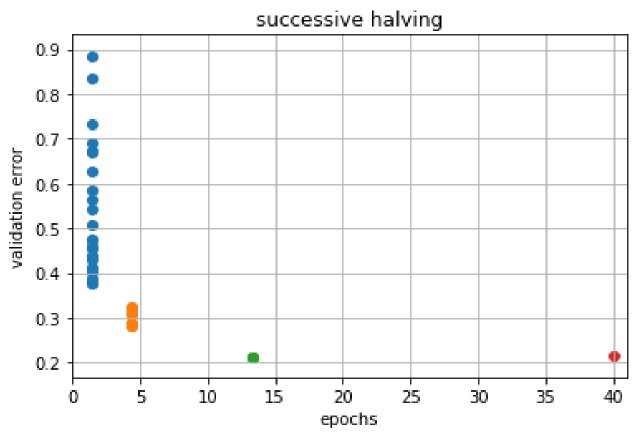
Parameters which will be optimized are the learning rate(logarithmic scale), batch size and the number of filters in each of the 3 convolutional layers and for convenience all hyperparameters as treated as continuous variables. Some data is collected by drawing and evaluating some N random configurations.In this step we are just collecting some initial random points in the parameter space of the objective. This is done from the Emukit RamdonDesing class.

[https://nbviewer.jupyter.org/github/amzn/emukit/blob/master/notebooks/Emukit-tutorial-basic-use-of-the-library.ipynb]

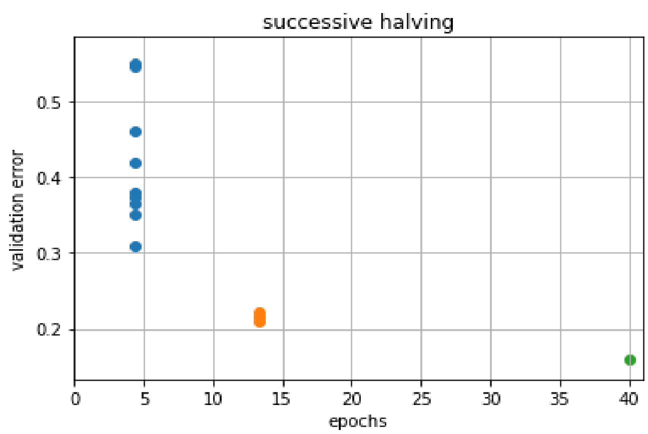
 	Best incumbent_value: 0.15873131034364588

Hyperband:

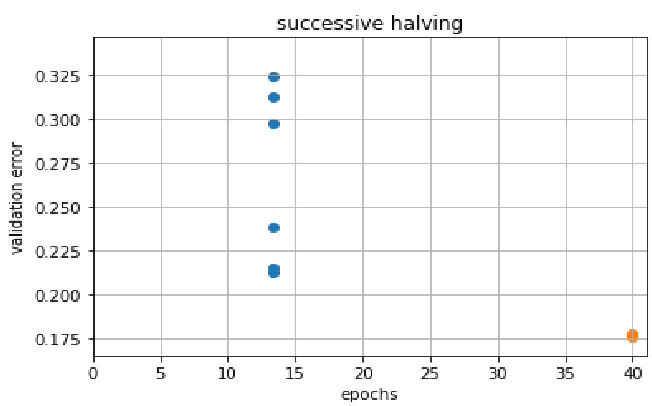
In hyperband we don't have any model, we just sample randomly some set of configurations that are related to the maximum iterations, eta and B. Then we Hedge and loop over varying degrees of the aggressiveness balancing breadth versus depth based search.The outer loop describes the hedge strategy and the innerloop describes the early-stopping procedure that considers multiple configurations in parallel and terminates poor performing configurations leaving more resources for more promising configurations.number of configurations n_i and the number of iterations they are run for r_i within each round of the Successive Halving innerloop for a particular value of (n,s).



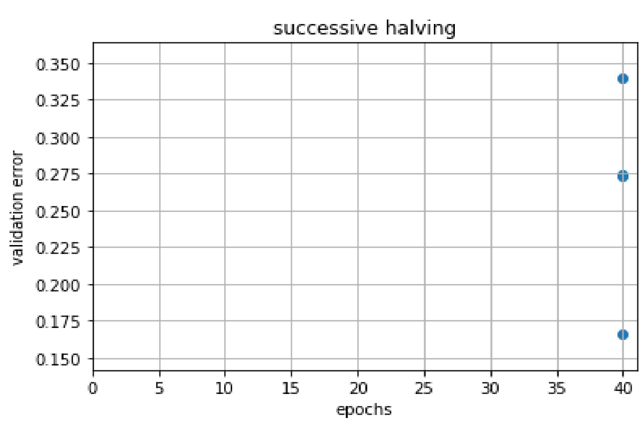
the value of n: 27
the value of r 1.4814814814814814
the value of n_i 27
the value of r_1 1.4814814814814814
Best loss: 0.3777981551228591
the value of n_i 9.0
the value of r_1 4.444444444444445
Best loss: 0.2820273568944227
the value of n_i 3.0
the value of r_1 13.333333333333332
Best loss: 0.21106234739540244
the value of n_i 1.0
the value of r_1 40.0
Best loss: 0.21106234739540244



the value of n: 9
the value of r 4.444444444444445
the value of n_i 9
the value of r_1 4.444444444444445
Best loss: 0.21106234739540244
the value of n_i 3.0
the value of r_1 13.333333333333334
Best loss: 0.20995269870917976
the value of n_i 1.0
the value of r_1 40.0
Best loss: 0.1589820250614489



the value of n: 6
the value of r 13.333333333333332
the value of n_i 6
the value of r_1 13.333333333333332
Best loss: 0.1589820250614489
the value of n_i 2.0
the value of r_1 40.0
Best loss: 0.1589820250614489

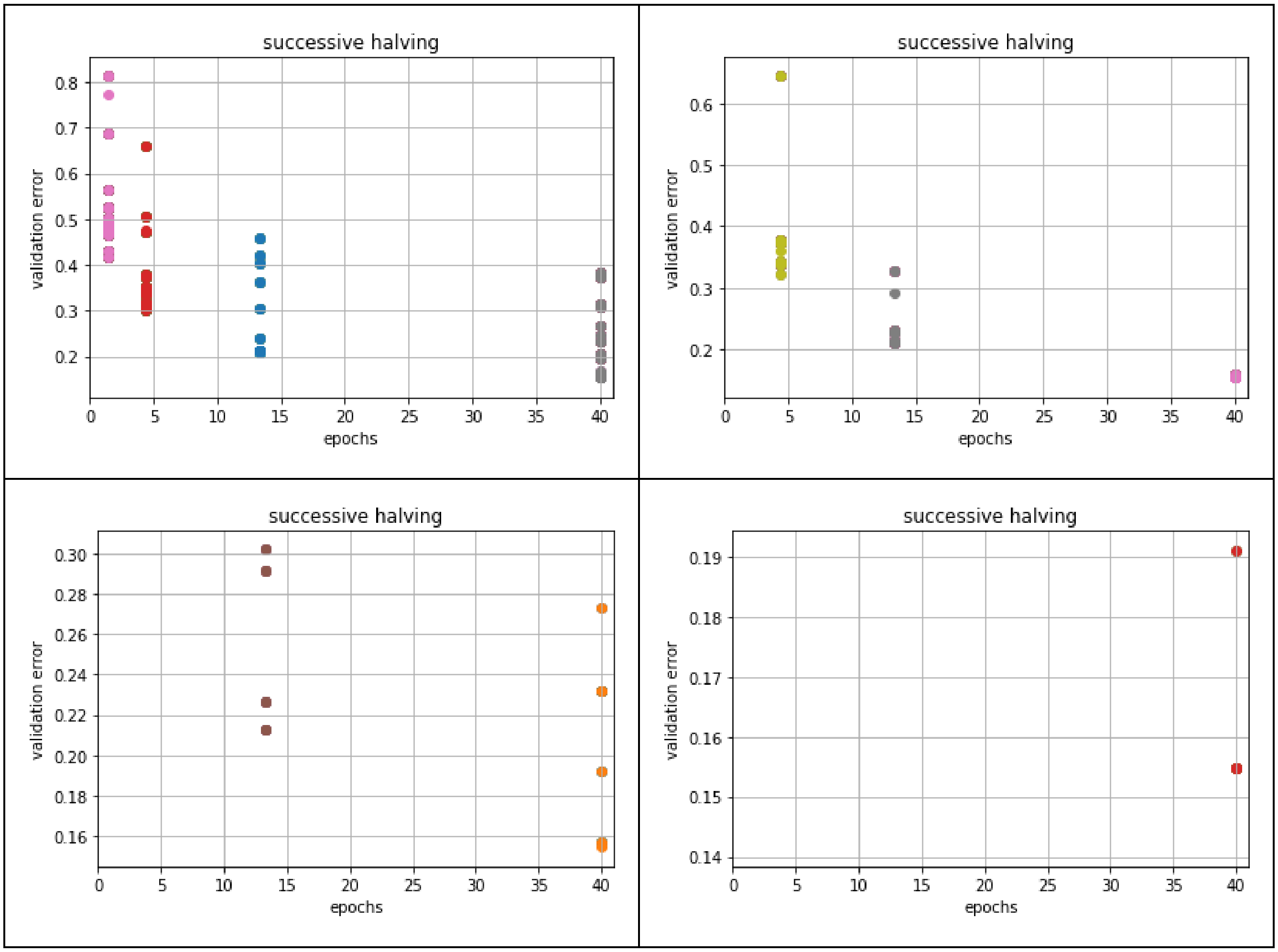


the value of n: 4
the value of r 40
the value of n_i 4
the value of r_1 40
Best loss: 0.1589820250614489

Combining Bayesian Optimization with Hyperband(BOHB):

To overcome the weakness of hyperband, i.e. it draws configuration randomly and hence might take exponentially long to approach the global optimum. So we combine Hyperband with a kernel density estimator that models the distribution of good and bad configurations. Below are the results of different successive halving rounds of configuration sampled in BOHB.

Last set of Successive Halving is depicted below:



FInal result: