Deep Learning Lab WSE 2018 Exercise 4

Assignment: Implementing a Bayesian, Hyperband and BOHB hyperparameter optimization

Objective
The goal of this exercise to learn how Bayesian optimization and Hyperband works and how we can combine them to optimize the hyperparameters of a convolutional neural network (CNN) on CIFAR-10. The architecture consists of 3 convolutional layers (with RELU activations and batch norm) and one final fully-connected layer. We use Adam to optimize the weights of the network

Implementation
Configuration Space:

Hyperparameter	Range
Learning Rate	[-6, -1]
Batch Size	[32,512]
Number of filters 1st layer	[4,10]
Number of filters 2nd layer	[4,10]
Number of filters 3rd layer	[4,10]

First Section BO: Bayesian optimization internally uses a model to guide the search. However, in order to train a model, we first need data. Use an initial design to collect some data points before we start with the actual Bayesian optimization loop. Draw and evaluate N random configurations.
Before implementing the main loop, we define the ingredients:

- a Gaussian process as a probabilistic model for the objective function
- Expected improvement as the acquisition function
- an optimizer for the acquistion function

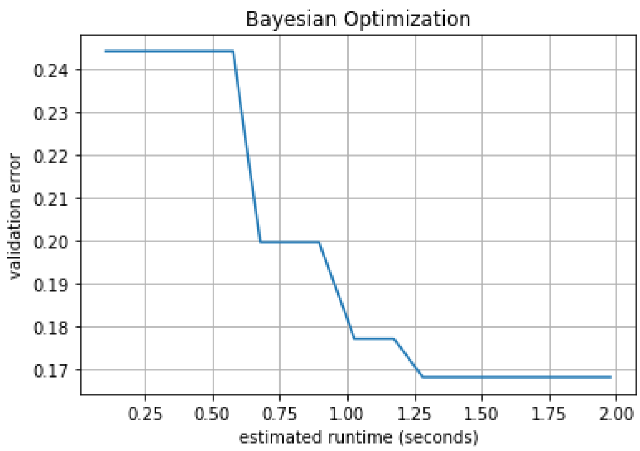
Next step, implementation of the main loop which 1. optimizes the acquisition function, 2. evaluates the objective function 3. augments our dataset and updates our model. Keep track of the current best solution

Second Section HB: We use the validation error after each epoch, as fidelity to speed up the optimization process.
Then implement a function that returns a random hyperparameter configuration sampled from a uniform distribution over the same configuration space as Bo optimization.
Implement successive halving.

Third Section BOHB:
We combine Hyperband with a kernel density estimator that models the distribution of the good and the bad configurations in the input space. By sampling from this model instead of a uniform distribution we can find good configurations much faster.

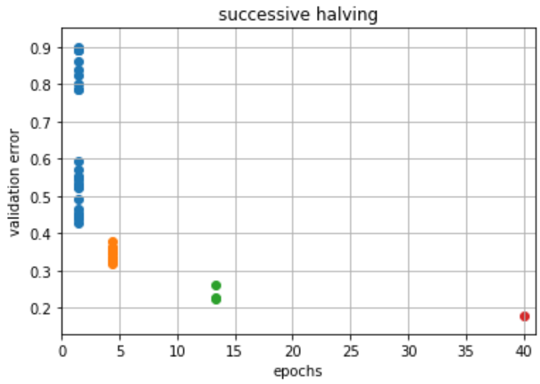
To implement the kernel density estimator, we use statsmodel and a multivariate kernel density estimator that fits a distribution over the whole input space.

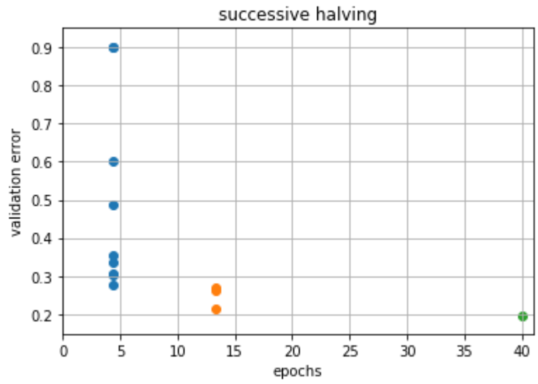
Results
Bayesian:

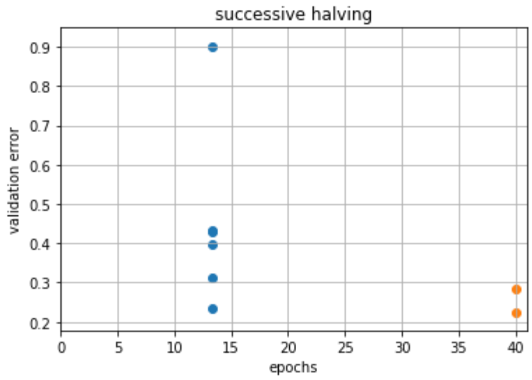


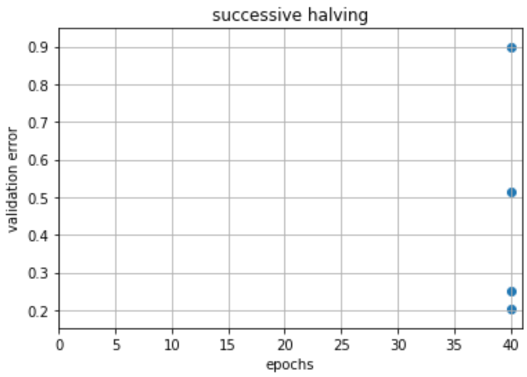
We can see from BO that at the end we get a good solution for the configuration but is intuitive because of the nature of the implementation that a real model would have a high computational cost that we can translate it in great amount of time to reach a good solution

<u>Hyperband:</u>



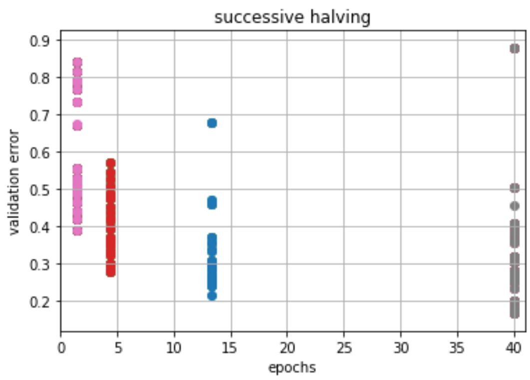


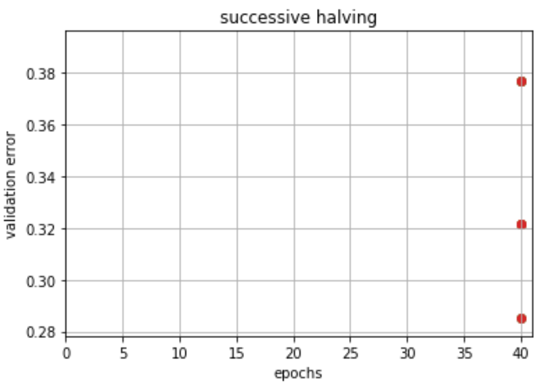


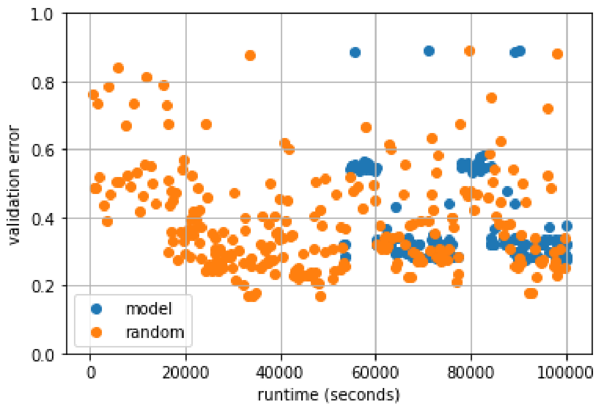


Images were printed by the model in consecutive order from left to right, top to bottom.
These graphs illustrate how the algorithm works, getting the hyperparameters randomly and evaluating configurations based on one budget, continue the best half to two units of budget, then best half of four units of budget until it gets the best unit to perform random search. We can see in this method that the implementation is completely random, so getting to a good solution could take some time since there is no track of a good solution founded by the algorithm in the first evaluations of the run.

<u>BOHB:</u>







We can observe the comparison between configurations selected randomly against the configurations following the model implementation. The implementation was really similar to Hyperband, but the difference is that this model uses probabilistic to get the density of good configurations and with this "keep track" of the best solutions. We can see that the blue points are way less disperse than the orange points.

Conclusion:
BO and HB individually can perform good but will take computational power and longer time to reach a optimum solution, while combining both methods can easily outcome the result from the individual methods by reaching an optimum solution faster and cheaper.

Feedback:
Really interesting and useful topic. Code implementation approximately 25 hours. Running time and debugging approximately 5 hours.
Implementation/Solution of the code around 30 hours.