Deep Learning Lab: Assignment #4

Francisco Rivera Valverde
francisco.rivera@students.uni-freiburg.de

***Abstract*—The present work discusses a Bayesian Optimization with Hyperband, called by their original authors BOHB. To conceptualize the benefits of this automatic machine learning aid tool, Bayesian optimization (BO) and Hyperband (HB) are analyzed separately. Emukit is used to facilitate Bayesian optimization with a gaussian process. For both BO and HB a surrogate of a CNN for CIFAR10 is used. BOHB can perform faster than traditional hyperband by using a multivariate kernel density estimator that defines the mean and std of a multivariate normal.**

I. Introduction

The surge of neural networks has enabled a wide range of domains to achieve a new state of the art in computer vision [1], natural language processing [2] and others. The flexibility of neural networks has allowed widespread use in multiple domains, yet there is a common problem during the design of the same. The finetuning of hyperparameters, required to make the network work, imply costly manual work and in extreme cases the need of costly expert to guide the model parameters.

Nevertheless, works like [3] has provided designers with a useful way to automatically fine tune hyperparameter. With such automated methods, Bayesian Optimization and Hyperband, a company can reduce cost and more efficiently deliver a product based on neural network. Under this motivation, the goal of this document is to revisit the benefits of combining Bayesian Optimization with Hyperband.

On this endeavor, section II discusses about Bayesian Optimization and section III about Hyperband. Consecutively, section IV discusses about BOHB (the combination of Bayesian Optimization and Hyperband). Section V presents some results of the practical implementations of the previous section and section VI concludes the most relevant remarks.

II. Bayesian Optimization

The first component of BOHB is the Bayesian optimization, described by the algorithm below [4]. The first step is to take a initial design of the form (X, Y) where X for this document is a set of hyperparameters that can be fed to a surrogate to generate a validation error Y that indicate how good X performed. A surrogate of a CNN for CIFAR10 is used for runtime constrains.

Algorithm 1 Bayesian Optimization

1: Initialize data $\mathcal{D}_0$ using an initial design.
2: **for** $t = 1, 2, \ldots$ **do**
3: Fit probabilistic model for $f(\boldsymbol{x})$ on data $\mathcal{D}_{t-1}$
4: Choose $\boldsymbol{x}_t$ by maximizing the acquisition function $a_p(\boldsymbol{x})$
5: Evaluate $y_t \sim f(\boldsymbol{x}_t) + \mathcal{N}(0, \sigma^2)$, and augment the data: $\mathcal{D}_t = \mathcal{D}_{t-1} \cup \{(\boldsymbol{x}_t, y_t)\}$
6: Choose incumbent $\hat{\boldsymbol{x}}_t \leftarrow \arg\min\{y_1, \ldots y_t\}$

Such initial design can be obtained by using the random design method of Emukit[1], a toolkit for decision making under uncertainty. The next step of Bayesian optimization proposes to fit a probabilistic model, which in the case of this document is a gaussian process with Matern52 kernel using GPy [5]. To choose the next point of the hyperparameter space, Emukit provides a AcquisitionOptimizer class which in our case optimizes an ExpectedImprovement class. Finally, the new point is exercised on the model and the gaussian process wrapper is adjusted accordingly.

III. Hyperband

"Even though Bayesian optimization is sample efficient, it still requires tens to hundreds of function evaluations" [6]. Therefore, the work of [7] proposes hyperband (described in the algorithm below) as a speedup of random search through adaptative resource allocation combined with early stopping.

Algorithm 3 Hyperband

Require: budgets b_{min} and b_{max}, η
1: $s_{max} = \lfloor \log_\eta \frac{b_{max}}{b_{min}} \rfloor$
2: **for** $s \in \{s_{max}, s_{max} - 1, \ldots, 0\}$ **do**
3: sample $n = \lceil \frac{s_{max}+1}{s+1} \cdot \eta^s \rceil$ configurations
4: run SH on them with $\eta^s \cdot b_{max}$ as initial budget

Hyperband for neural networks bases its functionality on the relative performance of a set of random configurations. Fundamentally, a good configuration is likely to perform better than bad configurations even at early epochs. For hyperparameters that do not follow this principle, like learning rate, [6] the authors "loop over varying degrees of aggressiveness balancing breadth versus depth-based search". Such can be seen from the algorithm above, where SH (successive halving) which comprises the early stopping component, only runs the best seen configurations "s" times to the max budget.

IV. Combining Bayesian optimization with Hyperband

The problem with hyperband, is that it draws configurations randomly, which is inefficient if a model over the neural network can better guide the sample selection. Particularly, the current work uses a multivariate kernel density estimator (KDE) that fits a distribution over the whole input space. The algorithm for BOHB described below [6], samples random configurations until there is enough data to adjust a KDE. Then, training points are taking from a multivariate normal with mean and std taken from the top 15% KDE configs.

[1] Available on https://github.com/amzn/emukit

Algorithm 4 Pseudocode for sampling in BOHB

Require: observations D, fraction of random runs ρ, percentile q, number of samples N_s, minimum number of points N_{min} to build a model, and bandwidth factor b_w

if rand() $\leq \rho$ **then**
    return random configuration
$b = \arg\max \{D_b : |D_b| \geq N_{min} + 2\}$
if $b = \emptyset$ **then**
    return random configuration
fit KDEs as in TPE but for each budget b
draw N_s samples according to $l'(\boldsymbol{x})$
return sample with highest ratio $l(\boldsymbol{x})/g(\boldsymbol{x})$

V. Results

With Bayesian optimization, one can trace how the optimizer can find minimums for the validation error. Figure 1 shows red datapoints product of optimizing the expected improvement over a gaussian process that model our neural network. It then tracks with a blue line the best validation error, where it can be seen that around 2000 seconds the best config is found.

Regarding hyperband optimization, one can plot the successive halving optimization to have a notion of how many runs where exercise to a high epoch. Figure 2 shows the first iteration of successive halving, where only one run with best performance reaches 40 epochs.

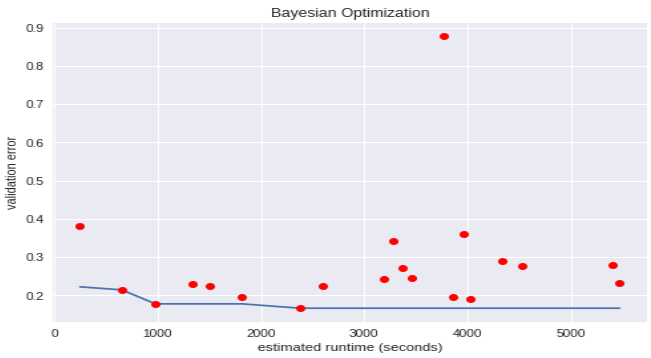

Fig. 1. Incumbent trajectory while Bayesian Optimization find new datapoints while optimizing a expected improvement over a gaussian process

One can see that at lower epochs samples are not taken from the model, but from a random distribution as there is not enough datapoints to build the model. It is possible to notice how random samples are also added.

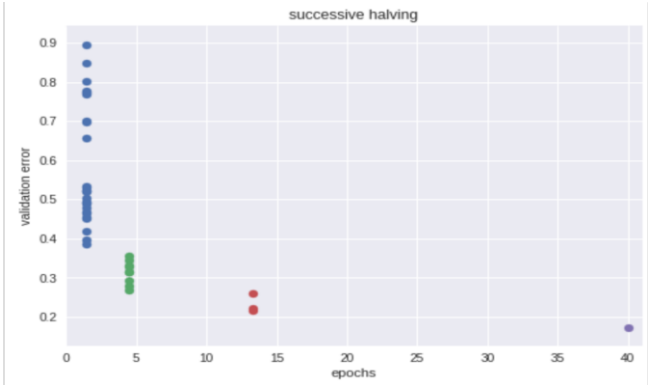

Fig. 2. HyperBand optimization for the first successive halving with 40 max epochs and downsampling of ETA=3

The equivalent plot for BOHB is shown in figure 3, where one has to account for random search data points (coming from initial runs as well as a random fraction of runs) as well as model generated samples. This can be seen more intuitively in figure 4. At the beginning there is no speedup compared to hyperband, but as soon as a good model is created, BOHB is able to reach faster than random search a better validation error.

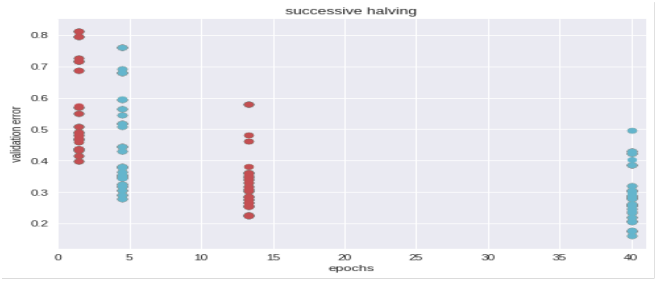

Fig. 3. Successive halving for BOHB with 40 max epochs and eta=3.

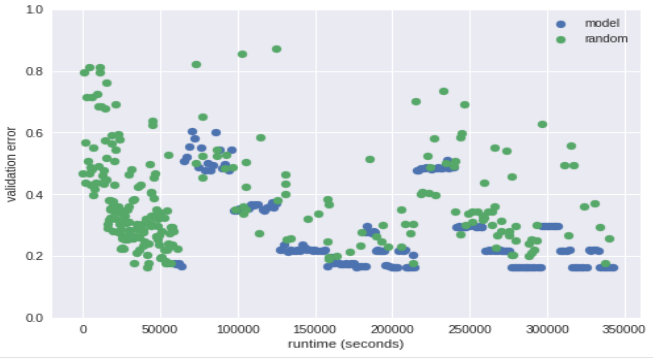

Fig. 4. Datapoints for BOHB hyperparameter optimization. In green one can see the random samples coming from both original datapoint as well as random points added 1/3 of the time. In blue one can see the samples coming from the multivariate normal model

VI. Conclusion

Hyperband is a technique that relies on fidelity models to explore for a minimum using random search. Such random search might take a lot of time to reach a minimum, and a model based on the random samples can help speed up the optimization. This work highlights how BOHB justifies extra complexity in the design to achieve a faster hyperparameter automation.

References

[1] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.

[2] Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." Advances in neural information processing systems. 2014.

[3] Falkner, Stefan, Aaron Klein, and Frank Hutter. "BOHB: Robust and efficient hyperparameter optimization at scale." arXiv preprint arXiv:1807.01774 (2018).

[4] Jones, Donald R., Matthias Schonlau, and William J. Welch. "Efficient global optimization of expensive black-box functions." Journal of Global optimization 13.4 (1998): 455-492.

[5] Gpy. "Gaussian Process Framework in Python". Taken from http://github.com/SheffieldML/GPy (2012).

[6] Albert Ludwigs Universitat Freiburg. Masterpraktikum Deep Learning lab. Taken from http://dl-lab.informatik.uni-freiburg.de(2018)

[7] Li, Lisha, et al. "Hyperband: A novel bandit-based approach to hyperparameter optimization." The Journal of Machine Learning Research 18.1 (2017): 6765-6816.