Improving Adaptive Hyperparameter Optimization with Genealogical History and HEBO

Kevin Blin [1] **Antoine Scardigli** [1] **Daniel Schoess** [2] **Steven H. Wang** [1]

Abstract

Current state-of-the-art adaptive hyperparameter optimisation algorithms like PB2 or GPBT reuse search algorithms designed for non-adaptive hyperparameter optimisation algorithms and therefore make assumptions that are not met in practice. In particular, the measured hyperparameter performances fulfill none of the necessary conditions for stationarity: noise is heteroscedastic, and both mean and variance are non-constant as a function of the number of training iterations. Finally, the insight from observations is sparser due to the additional temporal dimension that is added. We then found a state-of-the-art search algorithm (HEBO) that specifically circumvents those issues. Lastly, we adapted HEBO for state-of-the-art adaptive hyperparameter optimisation algorithms and compared our new results to state-of-the-art benchmarks. As hoped, our sounder approach is more robust and outperforms the current state-of-the-art hyperparameter optimization algorithms.

1. Introduction

Hyperparameter optimization is an important part of the machine learning process. Especially in deep learning, careful hyperparameter optimization is essential for strong model performance (Albelwi & Mahmood, 2016; 2017; Hinz et al., 2018; Loshchilov & Hutter, 2016; Talathi, 2015). Additionally, hyperparameter optimization can often be a finicky and unpredictable process. The most commonly used hyperparameter optimization processes are the simplest – grid search and random search (Bergstra & Bengio, 2012). However, several automated hyperparameter optimization algorithms have appeared on the horizon. For example, these algorithms include Hyperband (Li et al., 2018), which improves upon SuccessiveHalving (Jamieson & Talwalkar, 2016) algorithm by stopping inferior solutions early. Many of these algorithms are showcased in the black-box optimization challenge at NeurIPS, including the 2020 winner HEBO (Heteroscedastic and Evolutionary Bayesian Optimisation solver (Cowen-Rivers et al., 2021)) (Turner et al., 2021).

1.1. Adaptive Hyperparameter Tuning

Some hyperparameter tuning strategies are adaptive, meaning that they can change the hyperparameters of the model mid-run. Adaptive hyperparameter tuning strategies train several models in parallel, with each model trained for a fixed number of updates every iteration. At the end of each iteration, each (hyperparameter, model loss) pair is recorded, and this performance history is used to assign new hyperparameters to each model for the next training iteration. Algorithm 1 shows an example (see (Jaderberg et al., 2017; Parker-Holder et al., 2020)).

Algorithm 1 Adaptive Hyperparameter Optimization

Input: Model space $\mathcal{M}$, hyperparameter space $\mathcal{H}$, training function $\mathcal{T}$, loss function $loss$, search function $\mathcal{S}$, number of models n, training iterations $t_{\max}$

▷ P_m contains the pairs of HPs and their losses
$P_0 \leftarrow \emptyset$
$m \leftarrow$ the $(t_{\max} \times n)$ matrix of models
▷ We note $m_i^j = m[i][j]$
for $i \in \{0, \dots, t_{\max} - 1\}$ **do**
  for $k \in \{0, \dots, n-1\}$ **do**
    $h \leftarrow \mathcal{S}(P_i)$
    $m_i^k \leftarrow \mathcal{T}(m_{i-1}^z, h)$ for some $z < n$
    $loss \leftarrow \ell(m_i^k)$
    $P_{i+1} \leftarrow P_i \cup (h, loss)$
  end
end

Population-Based Training (PBT). PBT (Jaderberg et al., 2017) is an evolutionary adaptive hyperparameter tuning technique that maintains a population of models trained in parallel. Poorly scoring models and hyperparameters are overwritten by better performing models and variants of their hyperparameters mid-training (after every N model updates) so that computational resources can focus on promising solutions. PBT uses a random hyperparameter proposal function. Before overwriting, hyperparameters of the better performing models are perturbed via multiplication by a random number (e.g. a random number between 0.8 and

1.2).

Population-Based Bandits (PB2). PB2 (Parker-Holder et al., 2020) improves the efficiency of PBT and adds theoretical guarantees by using time-based Gaussian Process bandit algorithms instead of random heuristics to guide the hyperparameter search. The proposal function in PB2 first models the mean and variance of model performance given the update time step and the hyperparameter, and then uses an acquisition function such as Upper Confidence Bound to suggest promising hyperparameters.

Geneological Population Based Training (GPBT). Scardigli et al. (2021) present another extension of PBT based on an insight that in many adaptive algorithms (including PB2) there is a mistaken "isomodels" assumption.

Such algorithms use proposal functions that are trained on the history of hyperparameters chosen at each time step, but fail to take into account the effect of the changing model parameters at each training step on the effect of the hyperparameters selected at each time step. For example, in PB2, the Gaussian Process Bandit proposal function is trained to predict the model performance from a time-varying hyperspace input, but the model performance also depends on the model weights which are not included in the input.

GPBT addresses the "isomodels" assumption problem by masking out parts of the proposal function training data that could be harmful for future predictions. This is the same as keeping only the genealogical history associated with the parents and grandparents of each model, and reduces bias inside the training data handed to the hyperparameter proposal function.

The standard implementation of GPBT, which achieves the highest performance on benchmark tasks, is GPBT-TPE, which uses Tree Parzen Estimators (Bergstra et al., 2011) as its proposal function (Scardigli et al., 2021).

1.2. Issues in Adaptive Hyperparameter Tuning

The main rationale for our work is the following: we notice that state-of-the-art adaptive hyperparameter optimisation algorithms like PB2 or GPBT use hyperparameter proposal functions based on predictions, which make assumptions that are not met in practice.

(1) We can not assume that the noise in the proposal function's model performance prediction is homoscedastic or independent as a function of the number of iterations, because the noise of future iterations may be dependent on the noise from the current iteration, for example, a family of models with high mean will have proportionally bigger variance (heteroscedastic noise).

(2) The mean validation accuracy is expected to increase as a function of the number of training iteration, and so is the variance. Hence the data's moments are also non-stationary.

(3) The search space is much bigger because one dimension (number of iteration) is added. For this reason, exploration should be weighted than for standard search algorithms.

HEBO. HEBO is a search algorithm that addresses the three issues mentioned above and additionally circumvents the problem of different acquisition functions leading to different predictions (thus, the acquisition function itself becomes a design decision). Cowen-Rivers et al. (2021) noted that many Bayesian Optimization algorithms often fail to take the complexity of data into account by assuming homoscedastic noise and stationarity. HEBO avoids these wrong assumptions with two adaptations.

First, HEBO accounts for heteroscedasticity and non-stationarity through non-linear input-output transformations. Second, to retrieve new hyperparameters, HEBO utilizes an ensembled acquisition function based on a combination of multiple objective functions. Such an approach has been shown to lead to superior results (Lyu et al., 2018). An evolutionary optimizer is then used which increases HEBO's exploration capabilities through mutations (Cowen-Rivers et al., 2021). Therefore, HEBO is able to handle heteroscedastic non-stationary data, exploration in vast search spaces, and dependence on acquisition functions.

2. Models and Methods

In our work, to address the issues inherent to vanilla GPBT as described in Section 1.2, we combine HEBO and Genealogical Population-Based Training, and compare its performance to several baselines, including PB2 and vanilla GPBT.

Challenge in combining GPBT and HEBO The main challenge is to make the GPBT implementation use HEBO in a meaningful way. HEBO is a Bayesian Optimization algorithm that has no sense of a time dimension. So to adapt HEBO we must add a time dimension, and enforce exploration observations to be constrained only in the specific time hyperplane corresponding to the current time iteration. The exact problem of adapting Gaussian processes for time varying tasks is described in Bogunovic et al. (2016). We will therefore follow their recommendations (see implementation details).

Algorithm 2 GPBT-HEBO Algorithm

Input: n the number of children per generation step, $\mathcal{T}$: trainer, $t_{\max}$: maximum number of generation steps, T_g: number of training iterations between two generation steps, c: number of children per parent over number parents ratio, m_0: initial model, l: loss function.

▷ P_m contains the pairs of HPs and their losses
$P_0 \leftarrow \emptyset$

for $i \in \{1, \dots, t_{\max -1}\}$ **do**
  for $z \in \left\{0, \dots, \sqrt{n/c}\right\}$ **do** ▷ For each parent
    $m_i \leftarrow$ copied model from the z-th parent
    $P_i \leftarrow$ copied history from the z-th parent
    for $d \in \left\{0, \dots, \sqrt{n \cdot c}\right\}$ **do** ▷ For each child
      $j \leftarrow \sqrt{n \cdot c} \cdot z + d$ ▷ Fixed index for each child
      $h^j_{i+1} \leftarrow \text{HEBO}(P_i)$ ▷ Use HEBO for hp suggestion, conditioned on genealogical history
      $m^j_{i+1} \leftarrow \mathcal{T}(h^j_{i+1}, m_i)$ ▷ train child
      $_k \leftarrow \ell(m^j_{i+1})$ ▷ evaluate child
      $P_i \leftarrow P_i \cup (m^j_{i+1}, h^j_{i+1,j})$ ▷ Extend search history
    end
  end
end

Output: the best $m^j_{t_{\max}}$ and its HP schedule $(h^j_i)_{i<t_{\max}}$

2.1. Implementation Details

We start from the public code repositories of HEBO and GPBT. To make HEBO compatible with GPBT, we make the following modifications.

(1) We add the time dimension to HEBO, following the recommendations from Bogunovic et al. (2016). They propose several ways to solve this problem. For simplicity, we choose the first, their easiest, solution: the time becomes a parameter like any other, and the time dimension is simply treated like any other dimensions. The active learning process, therefore, tries to create a surrogate in a space with one additional dimension to explore from. Concretely, we need to add a dimension to the search space, add the iteration number to the hyperparameter observation tuple, and search for optimal new hyperparameter configuration, thanks to the acquisition function, only in the hyperspace with the time dimension equal to the iteration number.

(2) We modify the original HEBO implementation to allow swapping of the experience buffer according to the observed data points that are inherited from the parent.

(3) We adapt GPBT to be compatible with our modified HEBO by implementing an Oracle class. The oracle makes the link between the search algorithm (HEBO) and the scheduler (GPBT): The scheduler asks the Oracle for a batch of optimal hyperparameter configuration, and the Oracle returns these following the search algorithm's recommendation.

2.2. Baselines

As baselines we use both adaptive and non-adaptive hyperparameter tuning approaches. We will include both state-of-the-art in adaptive algorithms: PBT (Jaderberg et al., 2017), PB2 (Parker-Holder et al., 2020), GPBT (Scardigli et al., 2021) (see (Wan et al., 2022; Parker-Holder et al., 2021)), and the commonly used non-adaptive algorithm: BOHB (Falkner et al., 2018) (see (Cowen-Rivers et al., 2021; Turner et al., 2021; White et al., 2021; Lindauer et al., 2022; Li et al., 2021)). We will also compare our approach against vanilla HEBO (that is, HEBO without Genealogical Population-Based Training).

We use the Ray Tune (Liaw et al., 2018) implementations of PBT and PB2, and the Hyperopt implementation of BOHB.

2.3. Datasets

Evaluating hyperparameter optimisation is costly because many models need to be trained in parallel to evaluate a single hyperparameter optimization algorithm. We follow the approach of Scardigli et al. (2021) and train on relatively easy computer vision datasets such as MNIST (LeCun & Cortes, 2010) and FMNIST (Xiao et al., 2017), as this is what have been recommended after the proposal feedback.

The general way of comparing search algorithms from the literature is by comparing the mean and standard deviation of the quality curves as a function of running time.

2.4. Experimental Details

Training and Search Space details. For both MNIST and FMNIST experiments, we train LeNet-style Convolutional Neural Networks using the Adam optimizer (Kingma & Ba, 2017) with weight decay and dropout.

Our hyperparameter search space is over the Adam optimizer's learning rate η, Adam's β_1 and β_2 hyperparameters, the weight decay, and the dropout probability. For the hyperparameter ranges explored in this search space, see Table 1.

For every hyperparameter optimization algorithm we perform 10 trials of training 25 models to completion, with 10 epochs of training, and hyperparameter adaptation be-

hyperparameter	lower bound	upper bound
η	10^{-5}	1
β_1	10^{-4}	10^{-1}
β_2	10^{-5}	10^{-2}
dropout p	0	1
weight decay	0	1

Table 1. Hyperparameter search space for MNIST and FMNIST tasks, shared between all hyperparameter tuning methods.

tween each epoch if the hyperparameter tuning algorithm is adaptive.

Metrics. We compare the test accuracy of each hyperparameter optimization algorithm on each dataset against wall clock time on the same machine.

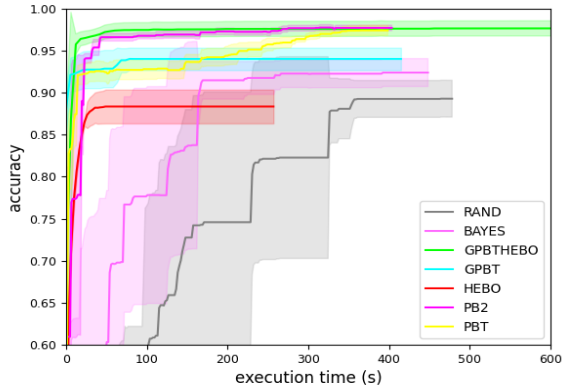

Figure 1. Test accuracy on MNIST

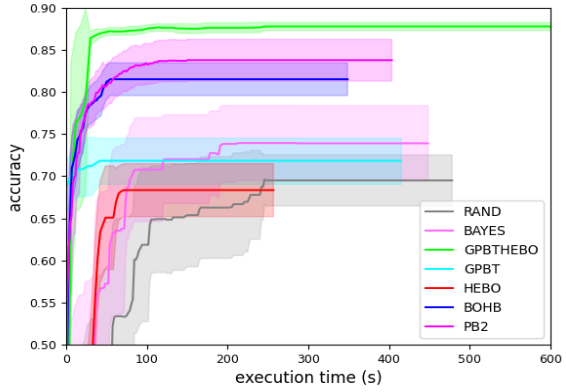

Figure 2. Test accuracy on FMNIST

3. Results

In this section we will compare the performance of different state-of-the-art hyperparameter optimization algorithms on our data sets.

Performance on MNIST data set Figure 1 shows that GPBT-HEBO performed best on the MNIST data set. PB2 and PBT also performed well in the end, however it took longer for these algorithms to achieve the same accuracy as a combination GPBT-HEBO. Both vanilla GPBT and vanilla HEBO had steep improvement curves, but leveled out below the algorithms above. The performance of vanilla GPBT was worse than expected. In the paper of Scardigli et al. (2021) the model outperformed both PB2 and PBT.

Performance on FMNIST data set In Figure 2, we can see that GPBT-HEBO performed significantly better than the other algorithms on the FMNIST data set. The algorithm outperformed the others both in terms of final accuracy achieved and speed of convergence. Besides GPBT-HEBO, BOHB and PB2 also performed well on this data set. Both vanilla HEBO and GPBT did not perform well. somewaht surprisingly since GPBT was supposed to outperform the other methods.

4. Discussion

Our method has smaller test variance and higher test mean accuracy. This is probably due to it being more robust because it is conceptually more sound. The computational time is unchanged in practice compared to the vanilla GPBT, because the training time overwhelms the HP optimization time. Still we propose a complexity analysis: if training one model for one epoch takes time T, the training time of our approach is $O(n \cdot m \cdot T)$, with m the number of iterations, and n the number of models trained per iteration. However, HEBO is in $O(number\_of\_data\_points^3)$ (Cowen-Rivers et al., 2021), with $number\_of\_data\_points = \sqrt{i} * n$ for iteration i (Scardigli et al., 2021). Given that the search algorithm is used n times per iteration, the overall complexity of the search algorithm is in $O(n^4 \cdot \sum_{i=1}^{m} i^{1.5}) = O(n^4 m^{2.5})$. Therefore the total complexity of GPBT-HEBO is $O(nmT + n^4 m^{2.5})$. In practice, T is huge and dominates the running time in all of our experiments, but our analysis shows that for huge n or huge m, GPBT-HEBO might be slowed down by the search algorithm side. This is a limitation of our work.

5. Conclusion

We observe that although Genealogical Population-Based Training achieves higher performance than other adaptive hyperparameter tuning algorithms on several computer vi-

sion benchmarks, GPBT originally uses hyperparameter proposal methods that still implicitly make erroneous assumptions which we describe in Section 1.2. HEBO is a recent Bayesian Optimization algorithm that addresses these erroneous assumptions, and we combine GPBT and HEBO to achieve higher performance on MNIST and FMNIST than vanilla GPBT and various adaptive hyperparameter optimization baselines.

References

Albelwi, S. and Mahmood, A. Automated optimal architecture of deep convolutional neural networks for image recognition. In *2016 15th IEEE International conference on machine learning and applications (ICMLA)*, pp. 53–60. IEEE, 2016.

Albelwi, S. and Mahmood, A. A framework for designing the architectures of deep convolutional neural networks. *Entropy*, 19(6):242, 2017.

Bergstra, J. and Bengio, Y. Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2), 2012.

Bergstra, J., Bardenet, R., Bengio, Y., and Kégl, B. Algorithms for hyper-parameter optimization. In *25th annual conference on neural information processing systems (NIPS 2011)*, volume 24. Neural Information Processing Systems Foundation, 2011.

Bogunovic, I., Scarlett, J., and Cevher, V. Time-varying gaussian process bandit optimization. In *Artificial Intelligence and Statistics*, pp. 314–323. PMLR, 2016.

Cowen-Rivers, A. I., Lyu, W., Tutunov, R., Wang, Z., Grosnit, A., Griffiths, R. R., Maraval, A. M., Jianye, H., Wang, J., Peters, J., and Ammar, H. B. An empirical study of assumptions in bayesian optimisation. 2021.

Falkner, S., Klein, A., and Hutter, F. Bohb: Robust and efficient hyperparameter optimization at scale. In *International Conference on Machine Learning*, pp. 1437–1446. PMLR, 2018.

Hinz, T., Navarro-Guerrero, N., Magg, S., and Wermter, S. Speeding up the hyperparameter optimization of deep convolutional neural networks. *International Journal of Computational Intelligence and Applications*, 17(02): 1850008, 2018.

Jaderberg, M., Dalibard, V., Osindero, S., Czarnecki, W. M., Donahue, J., Razavi, A., Vinyals, O., Green, T., Dunning, I., Simonyan, K., Fernando, C., and Kavukcuoglu, K. Population based training of neural networks. *CoRR*, abs/1711.09846, 2017. URL http://arxiv.org/abs/1711.09846.

Jamieson, K. and Talwalkar, A. Non-stochastic best arm identification and hyperparameter optimization. In Gretton, A. and Robert, C. C. (eds.), *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pp. 240–248, Cadiz, Spain, 09–11 May 2016. PMLR. URL https://proceedings.mlr.press/v51/jamieson16.html.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization, 2017.

LeCun, Y. and Cortes, C. MNIST handwritten digit database. 2010. URL http://yann.lecun.com/exdb/mnist/.

Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., and Talwalkar, A. Hyperband: A novel bandit-based approach to hyperparameter optimization. *JOURNAL OF MACHINE LEARNING RESEARCH*, 18, 2018. ISSN 1532-4435.

Li, Y., Shen, Y., Zhang, W., Chen, Y., Jiang, H., Liu, M., Jiang, J., Gao, J., Wu, W., Yang, Z., Zhang, C., and Cui, B. OpenBox: A generalized black-box optimization service. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, New York, NY, USA, August 2021. ACM.

Liaw, R., Liang, E., Nishihara, R., Moritz, P., Gonzalez, J. E., and Stoica, I. Tune: A research platform for distributed model selection and training. *arXiv preprint arXiv:1807.05118*, 2018.

Lindauer, M., Eggensperger, K., Feurer, M., Biedenkapp, A., Deng, D., Benjamins, C., Ruhkopf, T., Sass, R., and Hutter, F. Smac3: A versatile bayesian optimization package for hyperparameter optimization. *J. Mach. Learn. Res.*, 23:54–1, 2022.

Loshchilov, I. and Hutter, F. Cma-es for hyperparameter optimization of deep neural networks, 2016.

Lyu, W., Yang, F., Yan, C., Zhou, D., and Zeng, X. Batch bayesian optimization via multi-objective acquisition ensemble for automated analog circuit design. In *International conference on machine learning*, pp. 3306–3314. PMLR, 2018.

Parker-Holder, J., Nguyen, V., and Roberts, S. J. Provably efficient online hyperparameter optimization with population-based bandits. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 17200–17211. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/

c7af0926b294e47e52e46cfebe173f20-Paper.pdf.

Parker-Holder, J., Nguyen, V., and Roberts, S. Provably efficient online hyperparameter optimization with population-based bandits, 2021.

Scardigli, A., Fournier, P., Vilucchio, M., and Naccache, D. Genealogical population-based training for hyperparameter optimization. *arXiv preprint arXiv:2109.14925*, 2021.

Talathi, S. S. Hyper-parameter optimization of deep convolutional networks for object recognition. In *2015 IEEE International Conference on Image Processing (ICIP)*, pp. 3982–3986. IEEE, 2015.

Turner, R., Eriksson, D., McCourt, M., Kiili, J., Laaksonen, E., Xu, Z., and Guyon, I. Bayesian optimization is superior to random search for machine learning hyperparameter tuning: Analysis of the black-box optimization challenge 2020. In Escalante, H. J. and Hofmann, K. (eds.), *Proceedings of the NeurIPS 2020 Competition and Demonstration Track*, volume 133 of *Proceedings of Machine Learning Research*, pp. 3–26. PMLR, 06–12 Dec 2021. URL https://proceedings.mlr.press/v133/turner21a.html.

Wan, X., Lu, C., Parker-Holder, J., Ball, P. J., Nguyen, V., Ru, B., and Osborne, M. Bayesian generational population-based training. In *First Conference on Automated Machine Learning (Main Track)*, 2022. URL https://openreview.net/forum?id=HW4-ZaHUg5.

White, C., Neiswanger, W., and Savani, Y. Bananas: Bayesian optimization with neural architectures for neural architecture search. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12): 10293–10301, May 2021. doi: 10.1609/aaai.v35i12.17233. URL https://ojs.aaai.org/index.php/AAAI/article/view/17233.

Xiao, H., Rasul, K., and Vollgraf, R. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017. URL http://arxiv.org/abs/1708.07747. cite arxiv:1708.07747Comment: Dataset is freely available at https://github.com/zalandoresearch/fashion-mnist Benchmark is available at http://fashion-mnist.s3-website.eu-central-1.amazonaws.com/.