VISUALHYPERTUNER: VISUAL ANALYTICS FOR USER-DRIVEN HYPERPARAMTER TUNING OF DEEP NEURAL NETWORKS

Heungseok Park [1] **Jinwoong Kim** [1] **Minkyu Kim** [1] **Ji-Hoon Kim** [1] **Jaegul Choo** [2] **Jung-Woo Ha** [1] **Nako Sung** [1]

ABSTRACT

Deep learning researchers and practitioners often struggle to find an optimal set of hyperparameters to maximize model performance due to a large combinatorial search space. Existing hyperparameter optimization methods, which mostly rely on a fully automatic approach, only made a limited success, leaving room for human intervention via a visual analytic approach. In response, we propose VisualHyperTuner, a web-based visual analytics system that supports user-driven, in-depth analysis and hyperparameter tuning processes in a *model-agnostic* environment. VisualHyperTuner utilizes a new approach to effectively control hyperparameter optimization through an *iterative*, *interactive* tuning procedure allowing users to fine-tune the optimal hyperparameters based on their prior knowledge from the given results. By tightly integrating multiple coordinated views, users can explore the obtained results and get insights into the optimization behavior. To demonstrate the utility of VisualHyperTuner, we present a usage scenario with real-world examples.

1 INTRODUCTION

As deep neural networks evolve with advanced components and their highly modular architectures, the number of hyperparameters has significantly increased. Such increase in the number of hyperparameters hinders researchers and practitioners from finding an optimal hyperparameter setting with a limited amount of human labor and computational resource, often leading to sub-optimal model performance. Therefore, the methods and the interfaces for automated hyperparamter tuning have emerged as a critical task.

Recent studies to address hyperparameter optimization (HyperOpt) problems have employed algorithm-based (Jaderberg et al., 2017; Li et al., 2017) and platform system-based (Golovin et al., 2017; Tsirigotis et al., 2018; Liaw et al., 2018) approaches. These methods have numerous advantages such as parallelism, early stopping, and easy of use, which can significantly improve performance in terms of resource efficiency and the optimization procedure. However, there exist few studies on visual analytic approaches for HyperOpt tasks to effectively leverage human insights by steering the optimization process in a user-driven manner.

In response, we propose VisualHyperTuner, a web-based visual analytics system for analyzing and controlling HyperOpt system, which is a subsystem of NAVER internal machine learning platform (Kim et al., 2018a;b). Visual-

HyperTuner takes a new approach to effectively control HyperOpt through an *iterative* and *interactive* tuning procedure, allowing users to fine-tune the optimal hyperparameters based on their prior knowledge from the given results. Based on parallel coordinates (Inselberg et al.), different combinations of hyperparameters are effectively visualized as high-dimensional vectors, together with a particular objective metric (e.q., a test accuracy), and the results are visualized across various model types. To demonstrate how our proposed system works, we describe its usage scenarios.

2 USAGE SCENARIOS

In order to demonstrate how VisualHyperTuner can effectively support a real-world task, we present a usage scenario of VisualHyperTuner in an image classification application. Specifically, we focus how the core modules of VisualHyperTuner are effectively applied to the task, as follows:

- *Iterative* and *interactive* approaches can effectively steer HyperOpt process.
- Visualization can support in gaining insights and validation from HyperOpt results.

We applied our system in training the ResNet (He et al., 2016) with data augmentation by random erasing (RE) (Zhong et al., 2017) on the CIFAR-100 dataset.

The user tests six different hyperparameter settings of the ResNet-RE models step-by-step following our proposed workflow (Figure 1(b)). First, she runs an initial HyperOpt session (purple color in Figure 1) to tune the learning rate while fixing other hyperparameters in an initial stage (S1 in Figure 1(b)). Afterwards, she selects the ten best performing models from the visualization (S2) and obtains a sense of

---

[1]Clova AI Research, NAVER, Seongnam, Korea [2]Korea University, Seoul, Korea. Correspondence to: Nako Sung <@naver-corp.com>.

(a) Overview of VisualHyperTuner   (b) Fine-tuning scenario with VisualHyperTuner
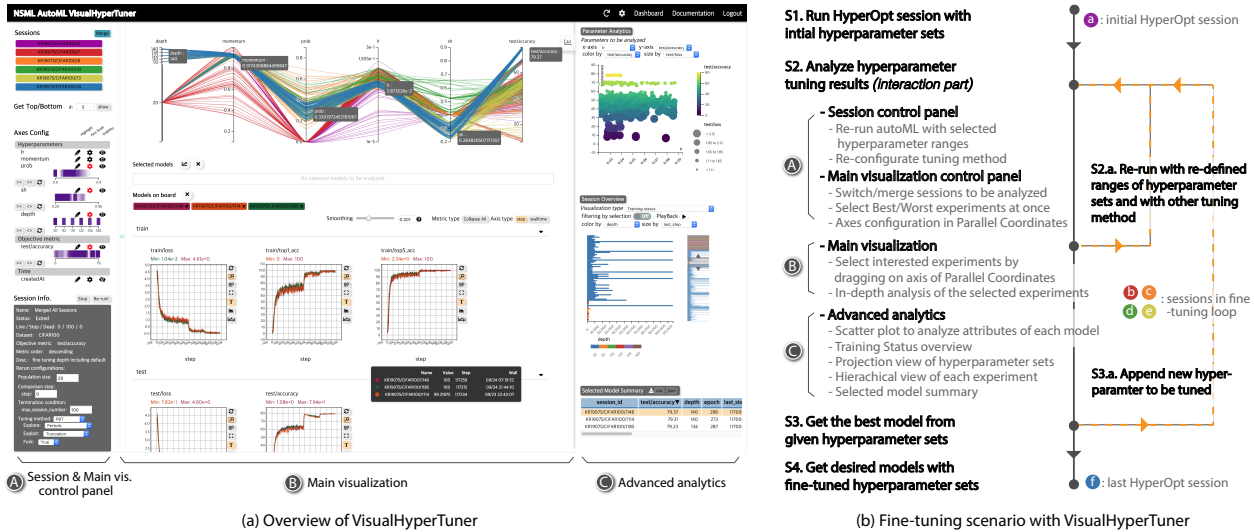
Figure 1. Overview of VisualHyperTuner. On the left, our system shows six different HyperOpt sessions with their different hyperparameter configuration, which are visualized in a parallel coordinates plot with different color encodings. Here, the three best performing models are shown in the detailed analysis panel; On the right, an overall workflow is described using the color-coded circles.

an optimal range of the learning rate (S3). With such an optimal range, she runs the second HyperOpt session to tune an additional hyperparameter in the momentum algorithm (red color). The session explores the space with the optimal range of a learning rate and the initial range of the momentum parameter (S2.a and S3.a). Consequently, she obtains the optimal range that gave high performance and continues with additional hyperparameters, similar to previous steps (S2). In this manner, she gradually adds other hyperparameters sequentially and fine-tunes the optimal range of each hyperparameter (from S2 to S3.a iteratively).

At the last stage, she adds another hyperparameter of the number of layers (yellow color), and she notices that the early stopping method enforces to quickly stop the experiments with deeper layer by looking the training status overview in the right-middle side of Figure 1(a). The horizontal bar graph presents that the session only keeps the experiments with a shallow depth (blue colored bars) because they performed better in the early stage.

Therefore, she reruns the HyperOpt session with the same hyperparameter configurations but without an early stopping

method (S2.a). Finally (S4 and blue color), she obtains the best performing model with the accuracy of 79.37%. Table 1 shows how the configuration changes according to the progress of the above-described steps, along with their corresponding accuracy. We note the ResNet-RE with 110 layers of our study achieved 78.34%, outperforming 77.9% of the original paper.

Conclusion. We presented our visual analytics system called VisualHyperTuner, a *model-agnostic* visual analytics for user-driven hyperparameter optimization for deep neural networks, as well as its usage scenarios. VisualHyperTuner has lots of additional features that was not fully described due to the page limit. We believe our human-in-the-loop approach for HyperOpt tasks can guide users towards optimizing their model performance.

REFERENCES

Golovin, D. et al. Google vizier: A service for black-box optimization. In *ACM SIGKDD*, pp. 1487–1495, 2017.

He, K. et al. Deep residual learning for image recognition. In *IEEE CVPR*, pp. 770–778, 2016.

Inselberg, A. et al. Parallel coordinates for visualizing multi-dimensional geometry. In *Computer Graphics 1987*.

Jaderberg, M. et al. Population based training of neural networks. *arXiv preprint arXiv:1711.09846*, 2017.

Kim, H. et al. Nsml: Meet the mlaas platform with a real-world case study. *arXiv preprint arXiv:1810.09957*, 2018a.

Kim, J. et al. Chopt: Automated hyperparameter optimization framework for cloud-based machine learning platforms. *arXiv preprint arXiv:1810.03527*, 2018b.

Li, L. et al. Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research*, 18(1):6765–6816, 2017.

Liaw, R. et al. Tune: A research platform for distributed model selection and training. *arXiv preprint arXiv:1807.05118*, 2018.

Tsirigotis, C. et al. Oríon: Experiment version control for efficient hyperparameter optimization. 2018.

Zhong, Z. et al. Random erasing data augmentation. *arXiv preprint arXiv:1708.04896*, 2017.

Table 1. Used configurations and and their accuracy

index	Top accuracy	Early stopped	Learning rate	Momentum	Erasing probability p	Max erasing area s_h	No. of layers
1	69.62	Yes	0.001 - 0.2	0.9	0.0	0.4	20
2	69.78	Yes	0.033 - 0.087	0.1 - 0.999	0.0	0.4	20
3	70.4	Yes	0.033 - 0.087	0.9 - 0.938	0.0 - 0.9	0.4	20
4	70.36	Yes	0.033 - 0.087	0.9 - 0.938	0.238 - 0.58	0.2 - 0.9	20
5	70.54	Yes	0.027 - 0.04	0.9 - 0.938	0.211 - 0.38	0.224 - 0.350	[20, 92, 110, 122, 134, 140]
6	79.37	No	0.027 - 0.04	0.9 - 0.938	0.211 - 0.38	0.224 - 0.350	[20, 92, 110, 122, 134, 140]