A. Experiments: learning hyperparameters

We train task model with the following hyperparameters for all methods and datasets: SGD optimizer with 0.9 Nesterov momentum, 1e-4 weight decay, 200 train epochs, 256 mini-batch size, cosine learning rate annealing with 5 warm-up epochs. Learning rate is 0.005 for SVHN and 0.1 for CIFAR-10/100 and ImageNet. The HO optimizer is RMSprop with 0.01 learning rate for ImageNet and 0.05 for others, and it uses the same schedule as the task optimizer.

Table 1. Runtime comparison of our and recently published methods, GPU hours. AutoDO results are estimated using V100 GPU.

Alg./Dataset	CIFAR-10	SVHN
AA [3]	5,000	1,000
PBA [5]	5.0	1.0
FAA [8]	3.5	1.5
OHL [9]	83.3	-
DADA [7]	0.1	0.1
AutoDO, 1 epoch	0.036	0.046
AutoDO, $E = 150$	1.8	2.3
AutoDO, $E = 50$	5.4	6.8

We use only $\boldsymbol{\theta}$ of penultimate fully-connected layer in (5) with $T = 5$ Neumann series iterations for $\boldsymbol{H}_{\boldsymbol{\theta}}^{-1}$ approximation. The T hyperparameter is chosen from Fig. 3 ablation study in [10]. The hyperparameter E in Alg. 1 is 100 for ImageNet and 50 for others. The soft-label initialization constant from Section 4.2 is $\alpha = 0.1$. The evaluated architectures are ResNet-18 for ImageNet, Wide-ResNet-28-10 for CIFAR-10/100 and SVHN. We replace ReLU nonlinearities in all networks with CELU [1] to satisfy C^1 requirement in (4). While IFT in (4) is only locally defined ($\|\boldsymbol{\lambda} - \acute{\boldsymbol{\lambda}}\| \leq r_1$ and $\|\boldsymbol{\theta} - \acute{\boldsymbol{\theta}}\| \leq r_2$) for a fixed point $(\acute{\boldsymbol{\lambda}}, \acute{\boldsymbol{\theta}})$, practically, the choice of hyperparameter E and CELU stabilizes optimization process in Alg. 1. In addition, there are an attempts to extend IFT to a global case in [6, 2, 4].

We run all experiments on P100/V100 GPUs and each one takes few hours depending on the dataset setting. Only large-scale ImageNet experiments can take up to several days. Table 1 presents detailed comparison between the estimated AutoDO runtime and the reported runtimes for other recent methods.

B. Experiments: SVHN learning curves

The additional learning curves for SVHN are showed in Figure 1. Our AutoDO optimization starts at epoch $E = \{10, 50, 150, 190\}$. It is evident that $E = 50$ hyperparameter has a minor edge over other settings, but practically $E = 150$ works almost the same with $3\times$ less computing.

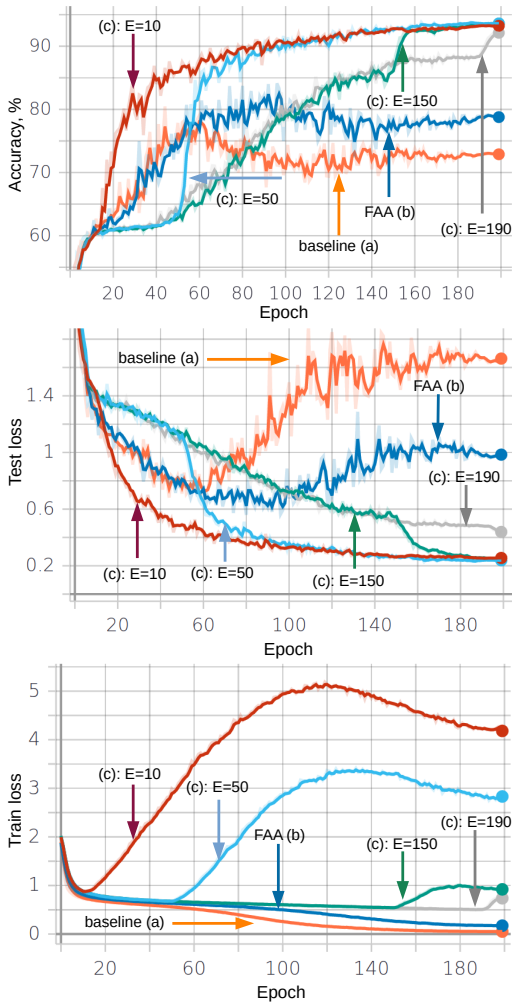

Figure 1. SVHN top-1 test accuracy (top), test loss (middle) and train loss (bottom) learning curves for the following models trained on a dataset with $100\times$ class imbalance and 10% label noise: (a) baseline, (b) FAA [8], and (c) our AutoDO with $\boldsymbol{\lambda}^{A,W,S}$. Our HO starts at epoch $E = \{10, 50, 150, 190\}$, which prevents overfitting to the distorted train data and improves generalization to test data.

C. Theoretical background: implicit differentiation meets density matching

The gradient of expectation in (7) over two independent distributions $\hat{Q}_{\boldsymbol{x}}^{\text{val}}$ and $\hat{Q}_{\boldsymbol{x}}$ is a product

$$\nabla_{\boldsymbol{\lambda}} \mathbb{E}_{\hat{Q}_{\boldsymbol{x}}^{\text{val}}, \hat{Q}_{\boldsymbol{x}}} \left[\mathcal{L}_{\text{v}}\right] = -\mathbb{E}_{\hat{Q}_{\boldsymbol{x}}^{\text{val}}} \left[\nabla_{\boldsymbol{\theta}} \mathcal{L}_{\text{v}}\right] \mathbb{E}_{\hat{Q}_{\boldsymbol{x}}} \left[\boldsymbol{H}_{\boldsymbol{\theta}}^{-1} \nabla_{\boldsymbol{\theta}} \nabla_{\boldsymbol{\lambda}}^T \mathcal{L}\right].$$

The Hessian $\boldsymbol{H}_{\boldsymbol{\theta}}$ itself is calculated as expectation over $\hat{Q}_{\boldsymbol{x}}$ due to computational and stability reasons. It leads to a connection to the Fisher information metric $\mathcal{I}_{\boldsymbol{\theta}}$ [11] as

$$\mathbb{E}_{\hat{Q}_{\boldsymbol{x}}} \left[\boldsymbol{H}_{\boldsymbol{\theta}}\right] = \mathbb{E}_{\hat{Q}_{\boldsymbol{x}}} \left[\frac{\partial^2 \mathcal{L}}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^T}\right] = \mathbb{E}_{\hat{Q}_{\boldsymbol{x}}} \left[\boldsymbol{u}_i(\boldsymbol{\theta}) \boldsymbol{u}_i(\boldsymbol{\theta})^T\right] = -\mathcal{I}_{\boldsymbol{\theta}},$$

where $\boldsymbol{u}_i(\boldsymbol{\theta}) = -\partial \mathcal{L}(i) / \partial \boldsymbol{\theta} = \nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{y}_i | \boldsymbol{x}_i, \boldsymbol{\theta}(\boldsymbol{\lambda}))$ are

the Fisher scores [12] for log-likelihood loss function.

The expectation $\mathbb{E}_{\hat{Q}_{\boldsymbol{x}}}\left[\nabla_{\boldsymbol{\theta}}\nabla_{\boldsymbol{\lambda}}^{T}\mathcal{L}\right]$ contains a second-order derivative of the train loss $\mathcal{L}(i) = -\log p(\boldsymbol{y}_i|\boldsymbol{x}_i, \boldsymbol{\theta}(\boldsymbol{\lambda})) = -\log p$ from (6) that can be similarly simplified to

$$\mathbb{E}_{\hat{Q}_{\boldsymbol{x}}}\left[\frac{\partial^2\mathcal{L}}{\partial\boldsymbol{\theta}\partial\boldsymbol{\lambda}^T}\right] = -\mathbb{E}_{\hat{Q}_{\boldsymbol{x}}}\left[\frac{\partial}{\partial\boldsymbol{\theta}}\left(\frac{\partial\log p}{\partial\boldsymbol{\lambda}^T}\right)\right] =$$
$$-\mathbb{E}_{\hat{Q}_{\boldsymbol{x}}}\left[-\frac{1}{p^2}\frac{\partial p}{\partial\boldsymbol{\theta}}\frac{\partial p}{\partial\boldsymbol{\lambda}^T} + \frac{1}{p}\frac{\partial^2 p}{\partial\boldsymbol{\theta}\partial\boldsymbol{\lambda}^T}\right] = \mathbb{E}_{\hat{Q}_{\boldsymbol{x}}}\left[\frac{\partial\mathcal{L}}{\partial\boldsymbol{\theta}}\frac{\partial\mathcal{L}}{\partial\boldsymbol{\lambda}^T}\right],$$

where the term $-\mathbb{E}_{\hat{Q}_{\boldsymbol{x}}}\left[\frac{1}{p}\frac{\partial^2 p}{\partial\boldsymbol{\theta}\partial\boldsymbol{\lambda}^T}\right] = -\frac{\partial^2}{\partial\boldsymbol{\theta}\partial\boldsymbol{\lambda}^T}\int_x p d\boldsymbol{x} = 0$.

By substituting the above derivations, the final form of (7) and its practical variant with equal-probability data points can be obtained as

$$\nabla_{\boldsymbol{\lambda}}\mathbb{E}_{\hat{Q}_{\boldsymbol{x}}^{\mathrm{val}},\hat{Q}_{\boldsymbol{x}}}\left[\mathcal{L}_{\mathrm{v}}\right] = \mathbb{E}_{\hat{Q}_{\boldsymbol{x}}^{\mathrm{val}}}\left[\boldsymbol{u}^{\mathrm{v}}(\boldsymbol{\theta})\right]\mathcal{I}_{\boldsymbol{\theta}}^{-1}\mathbb{E}_{\hat{Q}_{\boldsymbol{x}}}\left[\boldsymbol{u}(\boldsymbol{\theta})\boldsymbol{u}(\boldsymbol{\lambda})^T\right]$$
$$= \left[\frac{1}{M}\sum_{j\in\mathbb{M}}\boldsymbol{u}_j^{\mathrm{v}}(\boldsymbol{\theta})\right]\mathcal{I}_{\boldsymbol{\theta}}^{-1}\left[\frac{1}{N}\sum_{i\in\mathbb{N}}\boldsymbol{u}_i(\boldsymbol{\theta})\boldsymbol{u}_i^T(\boldsymbol{\lambda})\right].$$

References

[1] Jonathan T. Barron. Continuously differentiable exponential linear units. *arXiv:1704.07483*, 2017. 1

[2] Mihai Cristea. On global implicit function theorem. *Journal of Mathematical Analysis and Applications*, 2017. 1

[3] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. AutoAugment: Learning augmentation policies from data. *arXiv:1805.09501*, 2018. 1

[4] M. Galewski and M. Rădulescu. On a global implicit function theorem for locally Lipschitz maps via nonsmooth critical point theory. *arXiv:1704.04280*, 2017. 1

[5] Daniel Ho, Eric Liang, Ion Stoica, Pieter Abbeel, and Xi Chen. Population based augmentation: Efficient learning of augmentation policy schedules. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2019. 1

[6] Dariusz Idczak. A global implicit function theorem and its applications to functional equations. *Discrete and Continuous Dynamical Systems - Series B*, 2014. 1

[7] Yonggang Li, Guosheng Hu, Yongtao Wang, Timothy Hospedales, Neil M. Robertson, and Yongxin Yang. DADA: Differentiable automatic data augmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 1

[8] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast AutoAugment. In *Advances in Neural Information Processing Systems*, 2019. 1

[9] Chen Lin, Minghao Guo, Chuming Li, Xin Yuan, Wei Wu, Junjie Yan, Dahua Lin, and Wanli Ouyang. Online hyper-parameter learning for auto-augmentation strategy. In *The IEEE International Conference on Computer Vision (ICCV)*, 2019. 1

[10] Jonathan Lorraine, Paul Vicol, and David Duvenaud. Optimizing millions of hyperparameters by implicit differentiation. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2020. 1

[11] Alexander Ly, Maarten Marsman, A J Verhagen, Raoul Grasman, and Eric-Jan Wagenmakers. A tutorial on Fisher information. *Journal of Mathematical Psychology*, 2017. 1

[12] Laurens van der Maaten. Learning discriminative Fisher kernels. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2011. 2