BAYESIAN PERSPECTIVE ON VISUAL DATA AUGMENTATION FOR EFFICIENT UTILIZATION OF SUB-SAMPLED DATA

Joonhyun Jeong[1] **Sungmin Cha**[2] **Youngjoon Yoo**[1,2] **Sangdoo Yun**[2] **Jongwon Choi**[3]

Naver Clova[1], Naver AI Lab[2], Chung-Ang University[3]
{joonhyun.jeong,sungmin.cha,youngjoon.yoo,sangdoo.yun}@navercorp.com,
choijw@cau.ac.kr

ABSTRACT

This paper proposes a Bayesian formulation of the widely used image-mixing augmentations typically using two images, to general K-image cases. Based on the proposed formulation, we introduce a new uncertainty measuring method focusing on the image-mixing augmentations' randomness. We acquire the sub-sampled data pool that can efficiently represent the overall data distribution based on the estimated uncertainty. For supplementing the sub-sampling method, we propose a new data generation mechanism filling in the crucial blanks of the sub-sampled data pool to represent the overall distribution.

1 INTRODUCTION

The importance of data augmentation becomes further emphasized after the advent of deep learning networks (DeVries & Taylor, 2017; Yun et al., 2019; Zhang et al., 2017). Based on the necessity of augmented data, many studies have proposed various augmentation methods to improve the data efficiency for deep learning network. Among the various methods of data augmentation widely used, the image-mixing augmentation methods, such as Mixup Zhang et al. (2017) and CutMix Yun et al. (2019), recently show the impressive performance to train the large-scaled deep learning networks.

In this paper, we derive a novel Bayesian formulation that can generalize the image-mixing augmentation. Based on the proposed formulation, we successfully obtain the results even after expanding the augmenting variation of the image-mixing augmentation methods. Especially, we find that the mixture of three or more images can further improve the performance from the baseline methods using only the paired images. The proposed derivation is also important since the uncertainty of the augmented samples can be estimated by Bayesian formulation. Based on the estimated uncertainty, we can acquire the sub-sampled data pool that can efficiently represent the overall data distribution. To resolve the data aliasing problem occurred by the sub-sampling, we additionally propose a data generation mechanism filling in the crucial blanks of sub-sampled data pool to represent the overall distribution, showing the state-of-the-art performance compared to sub-sampling competitors.

2 TASK FORMULATION

2.1 PRELIMINARY: BAYESIAN DEEP LEARNING

Bayesian typically formulates the classification problem of the labeled samples $\{x_n, l_n\} \in \mathcal{D}, n = 1, ..., N$ as the inference problem of the posterior distribution $p(l|x)$. Here, we term x and l, which are the input image and label, to be the general samples in $\mathcal{D}$ for simplicity. By the derivation from Monte-carlo Dropout Gal & Ghahramani (2016), we can think the distribution $p(l|x)$ as the marginalized form of random variable W, which is network parameters here:

$$\begin{aligned} p(l|x) &= \int p(l|x, W)p(W)dW, \\ &\cong \frac{1}{N_s}\sum f_W(l|x)p(W), \end{aligned} \tag{1}$$

where the function $f_W(l|x)$ is the variational realization of the distribution $p(l|x, W)$, which is typically defined as the neural network with parameter W with final Softmax layer. N_s denotes the number of samples drawn by $p(W)$. The promising aspect of this derivation is that the usual stochastic gradient based (SGD) optimization of the model, with the weight jittering by $p(W)$, makes to mimic the variational function $f_W(\cdot)$ to the real distribution $p(l|x, W)$. Typically, the prior distribution $p(W)$ is defined by random perturbation of the weight W, i.e., dropout (Srivastava et al., 2014). In inference time, we can numerically approximate the distribution by the sampling from the prior distribution $p(W)$, as in (1). In this paper, we argue that the uncertainty can be defined either from weight parameter W, but from the data sample x.

2.2 Model Uncertainty from Composited Data

Here, we define a classification model using image-mixing augmentation in a probabilistic framework. Assume that the model input x_c are composed of $x_1, ..., x_K$, as in:

$$x_c = f_c(x_1, ..., x_K; \phi_1, ..., \phi_K), \tag{2}$$

where the function $f_c(\cdot)$ denotes the composite function, and the term $\boldsymbol{\phi} = \{\phi_1, ..., \phi_K\}$ is a mixing parameter denoting the portion of each sample x_k on the composite sample x_c. We note that it is a generalization of the popular image-mixing augmentations, CutMix and Mixup using two images, to K-image case. Specifically, for the Mixup case, $f_c(\cdot)$ is defined as the weighted summation as:

$$x_c = \Sigma_{k=1}^{K} \phi_k x_k. \tag{3}$$

The mixing parameter $\boldsymbol{\phi}$ is defined by Beta distribution in usual two-image cases (Yun et al., 2019; Zhang et al., 2017), and naturally be expanded to Dirichlet distribution $\boldsymbol{\phi} \sim Dir(\boldsymbol{\alpha}), \boldsymbol{\alpha} \in \mathcal{R}^K$. Obviously, the composite sample x_c is random variable given the hyper-parameter $\boldsymbol{\alpha}$.

For K-image generalization of CutMix, namely DCutMix, the definition of the function $f_c(\cdot)$ is more complicated to contribute all the images $x_1, ..x_K$ to composite image x_c with respect to their mixing parameters $\boldsymbol{\phi}$. Here, we composite the images by the stick-breaking process (SBP) Sethuraman (1994), one widely used approach of sampling from Dirichlet distribution.

Assume that we sample $\boldsymbol{\phi}$ from the prior distribution $Dir(\boldsymbol{\alpha})$ and want to composite the image with respect to the proportion $\phi_k \in \boldsymbol{\phi}$, $\Sigma_k \phi_k = 1$. From SBP, the composition can be done by defining the intermediate random variable $\boldsymbol{v} = [v_1, ..., v_K] \in \mathcal{R}^K$ such that:

$$\begin{aligned} v_1 &= \phi_1 \\ v_k &= \phi_k / \prod_{j=1}^{k-1} (1 - v_j), \ k = 2, ..., K. \end{aligned} \tag{4}$$

Now, we define the image fractions $\boldsymbol{r} = \{r_1, ..., r_K\}$ of the sample x; in this case, we set the sample x as an image for simplicity. Let the function $\tilde{r} = d(x|v, \gamma)$ randomly discriminates the image by $\tilde{r} : x \backslash \tilde{r}$ to the ratio $v : 1 - v$, where the term $x \backslash \tilde{r}$ denotes the region of x excluding $\tilde{r}$. Note that the variable v is defined as Beta-distribution by the derivation from SBP. The fraction $\boldsymbol{r}$ is calculated as:

$$r_k \quad = d(\{x \backslash \textstyle\sum_{j=0}^{k-1} r_j\}|v_k, \gamma), \ k = 1, ..., K-1. \tag{5}$$

where the virtual fraction r_0 is set to $\emptyset$, and the last fraction $r_K = x \backslash \sum_{j=1}^{K-1} r_j$. We will set the hyper-parameter γ to denote the randomness in the discrimination function, and hence the composite function f_c of the CutMix will be governed by two hyper-parameters $\boldsymbol{\alpha}$ and γ. See Appendix B for the classification performance enhancement from the generalization.

The probabilistic framework of the classification applying the augmentation is defined as:

$$\begin{aligned} p(l_c|x_c) &= \int p(l_c|x_c, \boldsymbol{\phi}) p(\boldsymbol{\phi}|\boldsymbol{\alpha}) d\boldsymbol{\phi}, \\ &= \int p(\Sigma_k \{\phi_k l_k\}|f_c(\boldsymbol{x}; \boldsymbol{\phi})) p(\boldsymbol{\phi}|\boldsymbol{\alpha}) d\boldsymbol{\phi}, \\ &\cong \frac{1}{N_s} \sum_{\boldsymbol{\phi}^{(j)}} f_W(\{\Sigma_k \phi_k^{(j)} l_k\}|f_c(\boldsymbol{x}; \boldsymbol{\phi}^{(j)})), \end{aligned} \tag{6}$$

where $(\boldsymbol{x}, \boldsymbol{l}) = \{(x_1, l_1), ..., (x_K, l_K)\}$, $l_c = \Sigma_k\{\phi_k l_k\}$, and $\boldsymbol{\phi}^{(j)}$ is the j^{th} sample drawn from the Dirichlet prior distribution $p(\cdot|\boldsymbol{\alpha})$. Hereafter, we define the label $l_i \in \mathcal{R}^C$ to be a one-hot indexing variable denoting one of total C classes. By the derivation from Monte-carlo Dropout Gal & Ghahramani (2016), we can set the distribution $p(l_c|x_c)$ to the variational function $f_W(\cdot)$, which is realized by a classification network having the network parameter W with softmax output. For CutMix case, we consider another variable γ in (6) as:

$$p(l_c|x_c) \cong \sum_{\gamma^{(i)}} \sum_{\boldsymbol{\phi}^{(j)}} f_W(\{\Sigma_k \phi_k^{(j)} l_k\} | f_c(\boldsymbol{x}; \boldsymbol{\phi}^{(j)}, \gamma^{(i)})). \tag{7}$$

Consequently, by (6) and (7), we calculate the uncertainty given the composite samples.

3 Dataset Distillation

3.1 Coefficient of Variation based Sub-sampling

The sub-sampling is not a trivial problem even though we can get exact confidence of each sample. Let assume some typical strategies: selecting most certain samples, vice versa, or uniformly distribute the samples over difficulty. We can easily think of some plausible aspects of each strategies. To tackle the sub-sampling problem, based on (7), the numerical approximation of the distribution $p(l_c|x_c)$, we first define the sub-sampling measure, corresponding to the distribution derived in (7):

$$E[L(l_c|x_c)] \cong \sum_{\gamma^{(i)}} \sum_{\boldsymbol{\phi}^{(j)}} L(\{\Sigma_k \phi_k^{(j)} l_k\} | f_c(\boldsymbol{x}; \boldsymbol{\phi}^{(j)}, \gamma^{(i)})), \tag{8}$$

where $\boldsymbol{x} = \{x_1, .., x_K\}$, and the loss $L(l_c|x_c)$ corresponds to the likelihood distribution $p(l_c|x_c)$. The expectation is defined on the space by the random variable $\boldsymbol{\phi}$ and $\boldsymbol{\gamma}$.

Using the expected loss, we select an **anchor sample** $x_i \in \boldsymbol{x}$ with fixed ϕ_i and then jitter $\boldsymbol{\phi} \backslash \phi_i$ related to other samples $\boldsymbol{x} \backslash x_i$ to calculate the uncertainty of the anchor sample x_i. The $\boldsymbol{\phi} \backslash \phi_i$ are drawn from a conditional Dirichlet distribution $D(\boldsymbol{\alpha} \backslash \alpha_i)$, by its definition. We will term $L_i = \{L_{i,m} | m = 1, ...M\}$ as the loss for all the jittered composition given the anchor x_i, and correspondingly calculated by (8). The number M denotes the total number of jittering from $\boldsymbol{\phi} \backslash \phi_i$.

To design the most effective data sub-sampling policy, we introduce a semantic measure function $O(\cdot)$ from the obtained L_i. The function $O(\cdot)$ is designed to indicate how informative the sample x_i is among all the images of same class, inspired by loss-based sub-sampling measures (Lin et al., 2017; Kumar et al., 2010; Kuchnik & Smith, 2018), and defined as $O(L_i) = \frac{\sigma(L_i)}{m(L_i)}$, where $\sigma(\cdot)$ and $m(\cdot)$ is the standard deviation and average of L_i. The definition of $O(\cdot)$ is called coefficient of variation (CV) based measure.

By using the measure $O(\cdot)$, a newly sub-sampled set D for each class is defined as follows:

$$D = S(O(L_k) | k = j_1, ..., j_{N_{intra}}, t), \tag{9}$$

where $S(\cdot)$ denotes a sampling function indicating whether x_k is to be included in D or not by using the sub-sampling ratio t. Here, j denotes the index of N_{intra} number of intra-class images where the class labels are equivalent among the others.

3.2 Sample Generation Model

To supplement the information loss from the sub-sampling, we propose a sample generation method where the generated sample $\tilde{x}$ can possibly lie in the similar manifold of composited samples $x_c = f_c(\boldsymbol{x}; \boldsymbol{\phi})$, given the equivalent soft-label $l_c = \{\Sigma_k \phi_k^{(j)} l_k\}$ for generating $\tilde{x}$.

Here, we define a variational distribution $\tilde{x} \sim q(x_a|x, \rho)$, where $q(x_a|x, \rho) = q_V(x_a|\mu) q_p(\mu|\rho)$, and x_a denotes a random variable approximating x_c. The distributions $q_V(x_a|\mu)$ and $q_p(\mu|\rho)$ are defined as a generation network given μ and a prior distribution sampling μ. The term ρ is defined as a sampling hyper-parameter for μ, where the vector $\mu \in \mathcal{R}^{\mathcal{C}}$ denotes the soft label. Our goal is to draw sample $\tilde{x}$ from the distribution $q(\cdot)$.

By substituting the distribution $q(\cdot)$ for (6), the distribution $p(l_c|x_c)$ is approximated by $\tilde{p}(\mu|x_a)$ as:

$$\begin{aligned}\tilde{p}(\mu|x_a) &= \int p(\mu|q_V(x_a|\mu))q_p(\mu|\rho)d\rho, \\ &\cong \frac{1}{N_s}\sum_{\mu^{(j)}} f_W(\mu^{(j)}|q_V(x_a|x,\mu^{(j)})),\end{aligned} \tag{10}$$

where the variable $\mu^{(j)}$ is drawn from $q_p(\cdot|\rho)$. Following (10), we train the generation network parameter V through minimizing cross-entropy loss between the softmax output of $f_W(\cdot)$ given the generated image $\tilde{x}$, and its soft label μ. In implementation, the variational prior $q_p(\mu|\rho)$ sets $\mu_{i=g}$ as $1-\epsilon$ for the ground truth class g of x and uniformly samples $\mu_{i\neq g}$ for background classes $i \neq g$, where μ_i denotes the label for class i and ϵ denotes a label smoothing factor as in Szegedy et al. (2016). The function $q_V(x_a|x,\mu)$ is defined as an auto-encoder structure: including an encoder $z = q_{V_E}(x)$ and a decoder $\tilde{x} = q_{V_D}(z,\mu)$. Here, the network parameter V is defined as $\{V_E, V_D\}$.

3.3 Experimental Results on Sample Generation

Figure 1 shows the performance and visualization of output images for the sample generation model V depending on the smoothing factor ϵ for the variational prior $q_p(\mu|\rho)$ in (10). As shown in right Figure 1, the generated image from the sample generation model on each ϵ may seem like simply adding a meaningless noise to the original image (i.e. $\epsilon = 0$). However, Ilyas et al. (2019) demonstrated that the small perturbations, the byproducts of adversarial attack for a specific background class, can serve as a *well-generalized* and *transferable* feature to train a new model. Through this findings, we want to argue that the noise in the generated image is not a meaningless noise but is a *feature* of the background classes, and we believe that it can help to supplement the information loss of the sub-sampling. We trained the sample generation model on five cases of ϵ. As a result, large ϵ diminishes the perceptible information of the original input image x, showing significantly deteriorated performance. On the other hand, small ϵ showed too small reflection of other background classes on x, hence showing little enhancement from the baseline case.Notably, utilizing generator with optimal $\epsilon = 0.5$ clearly outperformed the random sampling and the baseline case.

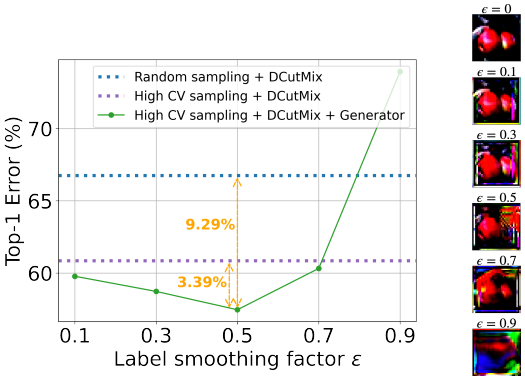

Figure 1: Performance (Left) and the output image visualizations (Right) from the generators, from various smoothing factor ϵ.

4 Conclusion

This paper introduced a new Bayesian perspective on expanding the image-mixing augmentations into general K-image cases. Based on this generalized formulation, we newly derived the data given Bayesian uncertainty of the model and showed its effectiveness on the data sub-sampling method, also proposing a sample generation method backing-up the information loss from the sub-sampling.

References

Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. *arXiv preprint arXiv:1812.00332*, 2018.

Sungmin Cha, Hsiang Hsu, Flavio P Calmon, and Taesup Moon. Cpr: Classifier-projection regularization for continual learning. *arXiv preprint arXiv:2006.07326*, 2020.

Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient

descent into wide valleys. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12): 124018, 2019.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. Dropblock: A regularization method for convolutional networks. In *Advances in Neural Information Processing Systems*, pp. 10750–10760, 2018.

Dongyoon Han, Jiwhan Kim, and Junmo Kim. Deep pyramidal residual networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5927–5935, 2017.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Weinberger. Deep networks with stochastic depth. In *ECCV*, 2016.

Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *arXiv preprint arXiv:1905.02175*, 2019.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

JangHyun Kim, Wonho Choo, Hosan Jeong, and Hyun Oh Song. Co-mixup: Saliency guided joint mixup with supermodular diversity. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=gvxJzw8kW4b.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Michael Kuchnik and Virginia Smith. Efficient augmentation via data subsampling. In *International Conference on Learning Representations*, 2018.

M Pawan Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. In *Advances in neural information processing systems*, pp. 1189–1197, 2010.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

Gabriel Pereyra, George Tucker, Jan Chorowski, Łukasz Kaiser, and Geoffrey Hinton. Regularizing neural networks by penalizing confident output distributions. *arXiv preprint arXiv:1701.06548*, 2017.

Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *AAAI*, 2019.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Jayaram Sethuraman. A constructive definition of dirichlet priors. *Statistica sinica*, pp. 639–650, 1994.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, and Quoc V Le. MnasNet: Platform-aware neural architecture search for mobile. *CVPR*, 2019.

Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *International Conference on Machine Learning*, pp. 6438–6447, 2019.

Sirui Xie, Hehui Zheng, Chunxiao Liu, and Liang Lin. Snas: stochastic neural architecture search. *arXiv preprint arXiv:1812.09926*, 2018.

Yuhui Xu, Lingxi Xie, Xiaopeng Zhang, Xin Chen, Guo-Jun Qi, Qi Tian, and Hongkai Xiong. Pc-darts: Partial channel connections for memory-efficient architecture search. In *International Conference on Learning Representations*, 2019.

Zhewei Yao, Amir Gholami, Kurt Keutzer, and Michael Mahoney. Pyhessian: Neural networks through the lens of the hessian. *arXiv preprint arXiv:1912.07145*, 2019.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 6023–6032, 2019.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4320–4328, 2018.

Hongpeng Zhou, Minghao Yang, Jun Wang, and Wei Pan. BayesNAS: A Bayesian approach for neural architecture search. In *ICML*, 2019.

A SAMPLE GENERATION MODEL

Implementation details: Let x and μ denote the original input image and a soft label where the ground truth label of x is smoothed by ϵ, respectively. For training the generation model V, we minimize the cross entropy loss L between μ and the output of the pretrained classifier W' as follows:

$$V^* = \arg\min_V L(\mu|W'(q_V(x_a|x,\mu))), \tag{11}$$

where $\tilde{x} = q_V(x_a|x,\mu)$ is defined as an auto-encoder structure which consists of an encoder $z = q_{V_E}(x)$ and a decoder $\tilde{x} = q_{V_D}(z,\mu)$. In order to generate the sample $\tilde{x}$ considering the distribution of μ, inspired by Mirza & Osindero (2014), the output of the encoder z is concatenated with the soft label μ to be the input of decoder q_{V_D}. Here, we employed U-Net (Ronneberger et al., 2015) as the backbone model for V. For training the generation model V, we used Adam optimizer (Kingma & Ba, 2014), and set the mini-batch size, momentum and weight decay as 64, 0.9 and 5e-4, respectively. We set the number of training epochs and initial learning rate as 600 and 0.0001, and the learning rate is decayed by the factor of 0.1 at 450 and 540 epoch.

Subsequently, for training a classifier W from the scratch using the pretrained generator V trained by (11), we minimize the cross entropy loss with the generated samples from V as follows:

$$W^* = \arg\min_W L(\mu|W(q_V(x_a|x,\mu))). \tag{12}$$

Note that this auxiliary loss from generated samples can be jointly optimized with the cross entropy loss from image mixing augmentation methods such as DCutMix.

Ablation study: Table 1 shows the ablation study for the components of the generator. The result shows that additional applying of the smoothed label with ϵ to the model without image generation showed higher error rate compared to the case where generator with smoothed label was applied. This result implies the effectiveness of using generator and the importance of reflecting the label information on x, which were not considered in Szegedy et al. (2016). Furthermore, the generator which employs the one-hot distribution as the prior distribution $q_p(\mu|\rho)$ did not prevail the generator where the smoothed label μ was employed as the prior distribution, since the generated sample $\tilde{x}$ resembles x too closely. This result informs the effectiveness of utilizing the soft label rather than one-hot label, regarding the design choice of generator.

Method	Top-1 Err (%)
Baseline (High CV sampling + DCutMix)	60.85
+ Label smoothing ($\epsilon = 0.5$)	59.46
+ generator (one-hot label)	59.73
+ generator ($\epsilon = 0.5$)	57.45

Table 1: Ablation study on the sample generation model.

B DISCUSSION: K-IMAGE GENERALIZATION OF IMAGE-MIXING AUGMENTATION METHODS

Our supposition is that the proposed probabilistic framework (6) well generalizes the data distribution: intuitively, the randomly composited samples will densely cover the data spaces between the training samples. In this section, to verify the advantage of our method, we conduct the experiments on K-image generalization of CutMix and Mixup in the classification task. By the observations from the experiments, we demonstrate that the blessing of the proposed generalization comes from making a model converge to a more lower and wider local minima.

We present the toy-classification test results on CIFAR-10 and 100 (Krizhevsky et al., 2009) dataset, as in Table 2 and 3. The results are obtained from the equivalent training and augmentation specific hyper-parameter setup used in Yun et al. (2019). Firstly, Table 2 shows the superiority of DCutMix and DMixup, which are the proposed generalized version of CutMix and Mixup, over the other augmentation and regularization methods. With light-weight backbone PyramidNet-110 Han et al. (2017), DCutMix and DMixup improve the performance compared to CutMix and Mixup by approximately 1% and 0.32%, respectively. For the deeper neural network PyramidNet-200, DCutMix

Model	# Params	Top-1 Err (%)
PyramidNet-110 ($\tilde{\alpha} = 64$) Han et al. (2017)	1.7 M	19.85
+ Mixup Zhang et al. (2017)	1.7 M	18.92
+ CutMix Yun et al. (2019)	1.7 M	17.97
+ DMixup ($K = 3, \alpha = \frac{1}{3}$)	1.7 M	**18.6**
+ DCutMix ($K = 5, \alpha = 0.2$)	1.7 M	**16.95**
PyramidNet-200 ($\tilde{\alpha} = 240$)	26.8 M	16.45
+ StochDepth Huang et al. (2016)	26.8 M	15.86
+ Label Smoothing Szegedy et al. (2016)	26.8 M	16.73
+ Cutout DeVries & Taylor (2017)	26.8 M	16.53
+ DropBlock Ghiasi et al. (2018)	26.8 M	15.73
+ Mixup Zhang et al. (2017)	26.8 M	15.63
+ Manifold Mixup Verma et al. (2019)	26.8 M	15.09
+ CutMix Yun et al. (2019)	26.8 M	14.47
+ DMixup ($K = 3, \alpha = 1$)	26.8 M	**15.07**
+ DCutMix ($K = 5, \alpha = 1$)	26.8 M	**13.86**

Table 2: Comparison of DCutMix and DMixup against other augmentations and regularization methods for PyramidNet-110, 200 models on CIFAR-100 dataset.

Model	Top-1 Err (%)
PyramidNet-200 ($\tilde{\alpha} = 240$)	3.85
PyramidNet-200 + Cutout DeVries & Taylor (2017)	3.1
PyramidNet-200 + Mixup Zhang et al. (2017)	3.09
PyramidNet-200 + Manifold Mixup Verma et al. (2019)	3.15
PyramidNet-200 + CutMix Yun et al. (2019)	2.88
PyramidNet-200 + DMixup ($K = 5, \alpha = 0.2$)	**2.9**
PyramidNet-200 + DCutMix ($K = 3, \alpha = \frac{1}{3}$)	**2.42**

Table 3: Impact of DCutMix and DMixup on CIFAR-10 dataset. PyramidNet-200, heavier than Pyramid-110, network was used.

and DMixup achieve the enhancement compared to original version and DCutMix shows the lowest top-1 error compared to other baselines. Finally, we evaluate our proposed methods on CIFAR-10 dataset as shown in Table 3. We can see from the result that DCutMix and DMixup both achieved the performance enhancement. Specifically, DCutMix shows the best performance among the baseline augmentation methods again. The overall results demonstrate the effectiveness of the proposed K-image generalization for augmentation methods. For more explicit investigation, we further analyze the generalized CutMix by its loss landscape.

Flatness of the loss-surface near local minima has been considered as one key signal for better generalization of the model, by the number of previous studies (Keskar et al., 2016; Pereyra et al., 2017; Zhang et al., 2018; Chaudhari et al., 2019; Cha et al., 2020). Based on these findings, we plot the **patterns of loss-surfaces** of each model by using PyHessian (Yao et al., 2019) framework, as shown in Figure 2. The plotted result shows that our generalized version of CutMix (i.e., DCutMix) has a flatter loss-surface near local minima among the comparisons. Also, DCutMix enables to have lower losses in overall, denoting well generalization to the unseen test data as well. We believe that this observation can be one supporting signal for the superior result of proposed augmentation. From a more analytical point of view, we can nully-hypothesize that the merit of K-image generalization of CutMix regarding the wide local minima comes from a more softened label than that of CutMix. This is because several papers have reported that a model trained with an *artificially* smoothed label can make a model converge to wide local minima and hence achieve better generalization (Pereyra et al., 2017; Zhang et al., 2018; Cha et al., 2020), for example, Label-Smoothing shown in Figure 2. However, as opposed to the previous methods, note that our softened label *directly* reveals the augmented ratio of several images, and hence, we conjecture that this would be one key factor why the model trained by our approach converges to lower and wider minima.

The previous augmentation methods (Kim et al., 2021; Zhang et al., 2017) reported a conflicting result with ours, where they could not obtain the positive enhancement by the K-image generalization of the augmentation method. However, we argue that we could achieve the advantage of it with

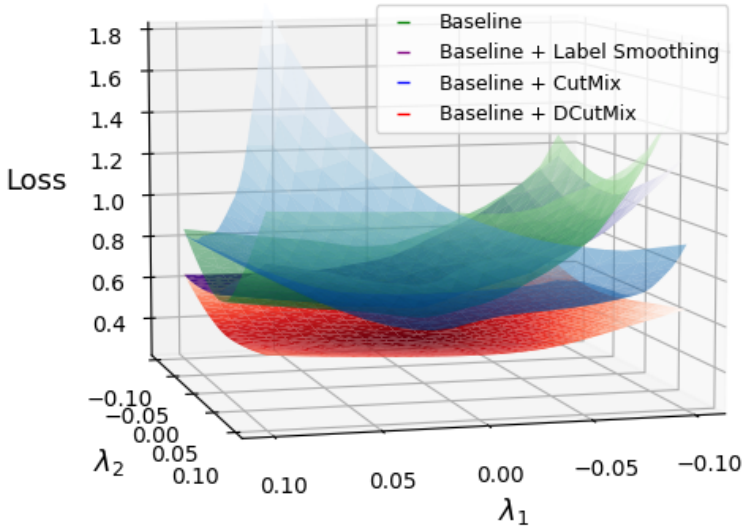

Figure 2: Comparison of image mixing augmentation and regularization methods in perspective of the loss-surface near local minima. We measured the loss-surfaces on CIFAR-100 with PyramidNet Han et al. (2017). λ_1 and λ_2 denote the degree of perturbation across the top-1 and 2 eigen vectors.

above experimental and analytical results. Also, we conducted the experiment with the same setting used in Kim et al. (2021) and we could get a positive improvement after expanding to K-image generalization, as shown in Table 4.

K (# inputs for Mix)	CutMix‡	CutMix
K=2	21.29	21.34
K=3	22.01	21.01
K=4	22.20	20.5
K=5	-	21.01
K=7	-	20.9
K=9	-	20.93

Table 4: Top-1 error rate tendency given mixing multiple images for CutMix on CIFAR-100 and PreActResNet18. ‡ denotes the reported result from Kim et al. (2021).

C Discussion: Dataset Distillation

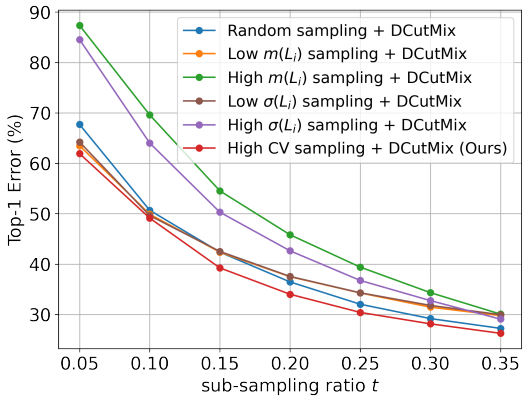

Figure 3: Top-1 test error plot of data sub-sampling methods tested on CIFAR-100. The model were trained with DCutMix.

We investigate the proposed sub-sampling method trained with DCutMix as an augmentation in Figure 3. In the figure, we compare our data sub-sampling method with others using various measures for $O(\cdot)$ and function $S(\cdot)$. The full 10K validation set of CIFAR-100 was used for evaluation, and we report the averaged results experimented by three independent seeds using PyraimdNet (Han et al., 2017).

As shown in the graph in Figure 3, sampling the easy-only or hard-only examples based on $m(L_i)$ shows deteriorated performance compared to the random sub-sampling. The hard-only sub-sampling severely suffered from poor generalization, which indicates that the outliers and noisy examples are not informative for training a model under the constraint of small number of training samples. These results imply that sub-sampling only easy or hard samples extracts the biased data samples that cannot be helpful for better generalization. Also, simply employing standard deviation $\sigma(L_i)$ as measure $O(\cdot)$ showed a similar test error plot compared to the above mean based sampling methods. On the other hand, our high CV based sub-sampling outperforms the random sampling by a significant margin, especially 5.79% lower test error in case of the number of sub-sampled training samples being extremely small (i.e. $t = 0.05$).

D Application: Network Architecture Search

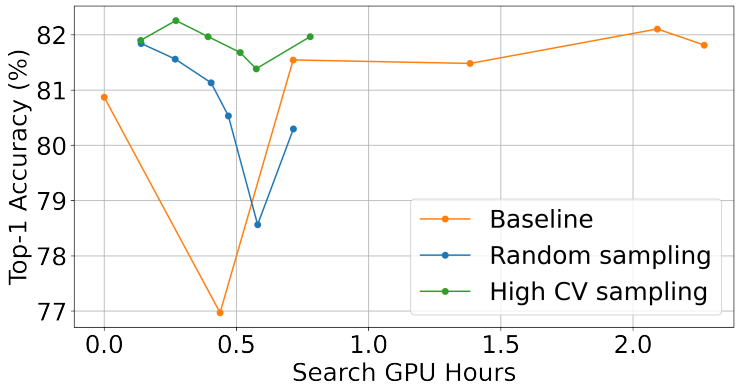

Figure 4: Performance of neural networks searched on the entire CIFAR-100 dataset (baseline), randomly sub-sampled CIFAR-100 dataset, and sub-sampled CIFAR-100 drawn by high CV measure. We adjusted the search epochs from 0 to 50 for baseline, while adjusting the sub-sample ratio t from 0.05 to 0.3 for sub-sampling.

This section shows the practical and effective usage of our proposed data distillation method and sample generation model on another domain, specifically on network architecture search (NAS). Our goal is to reduce the time spent for searching the architectures by searching on the sub-sampled dataset drawn from our framework, rather than on the full training dataset. We adopt one of the most computationally efficient and stabilized NAS method, PC-DARTS (Xu et al., 2019) as our baseline. Here, we used the equivalent searching hyper-parameters proposed in Xu et al. (2019).

Figure 4 shows the outstanding efficiency of our sub-sampling framework in terms of search time and accuracy. Especially, ours (searching on high CV sub-sampled dataset) achieves comparable accuracy with 7.7$\times$ reduced search time compared to other baselines. Furthermore, ours consistently outperforms random sub-sampling given the equivalent number of data samples.

As shown in Table 5, we could observe that ours serves as an effective proxy dataset and the neural network searched by ours is also well-generalized on ImageNet. Note that, ours reduced the search GPU time up to 0.01 days (i.e. 16 minutes), while showing comparable or even higher accuracy compared to the models searched with PC-DARTS on the entire CIFAR-10, CIFAR-100, and randomly sub-sampled ImageNet dataset. Also, compared to the other NAS methods, ours achieves the best performance while enjoying the significantly reduced search computational cost.

As an ablation study of our method on NAS, we conducted the experimental analysis to verify the impact of high CV sub-sampling and sample generator on the search process in Table 6. Here, we searched for 50 epochs for all methods and set the sub-sampling ratio $t = 0.05$ for the cases searching with high CV sub-sampled dataset. Regarding the generator, we optimize the cross entropy loss

Architecture	**Top-1 Err (%)**	**Top-5 Err (%)**	**# Params (M)**	**# FLOPs (M)**	**Search Cost (GPU days)**	**Search method**
AmoebaNet-C (Real et al., 2019)	24.3	7.6	6.4	570	3150	evolution
MnasNet-92 (Tan et al., 2019)	25.2	8.0	4.4	388	-	RL
ProxylessNAS (Cai et al., 2018)	24.9	7.5	7.1	465	8.3	gradient-based
SNAS (Xie et al., 2018)	27.3	9.2	4.3	522	1.5	gradient-based
BayesNAS (Zhou et al., 2019)	26.5	8.9	3.9	-	0.2	gradient-based
PC-DARTS (CIFAR10) Xu et al. (2019)	25.1	7.8	5.3	586	0.1	gradient-based
PC-DARTS (ImageNet) Xu et al. (2019)	24.2	7.3	5.3	597	3.8	gradient-based
PC-DARTS (CIFAR100) Xu et al. (2019)	23.8	7.09	6.3	730	0.1	gradient-based
PC-DARTS (High CV Sub-sampled CIFAR100)	24.3	7.2	5.8	671	**0.01**	gradient-based

Table 5: Comparison of the state-of-the-art NAS methods on ImageNet under comparably small resource constraints. $(\cdot)$ denotes the proxy dataset where the architecture was searched on.

between the generated samples and soft labels subject to the architecture hyper-parameters (i.e. α, β in Xu et al. (2019)). Note that high CV sampling considerably reduced search time of PC-DARTS with comparable accuracy. Utilizing generator for searching further raises accuracy by 1.07% with a small burden of search resources, while still showing significantly reduced search time compared to baseline PC-DARTS.

Method	Search Mem (GB)	Search Cost (GPU hours)	Top-1 Acc (%)
PC-DARTS (baseline)	10.9 (1×)	2.26 (1×)	81.81
+ High CV sampling	10.5 (0.96×)	0.13 (0.05×)	81.89
+ High CV sampling + generator	12.9 (1.18×)	0.17 (0.07×)	82.88

Table 6: Impact of high CV sampling and the generator on the performance of PC-DARTS in terms of GPU usage, search time and the accuracy, where the sub-sampling ratio $t = 0.05$.