DivAug: Plug-in Automated Data Augmentation with Explicit Diversity Maximization

Zirui Liu[*1], Haifeng Jin[*2], Ting-Hsiang Wang[2], Kaixiong Zhou[1], and Xia Hu[1]

[1]Rice Univeristy
[2]Texas A&M University
{zl105, kz34, xia.hu}@rice.edu
{jin, thwang1231}@tamu.edu

Abstract

*Human-designed data augmentation strategies have been replaced by automatically learned augmentation policy in the past two years. Specifically, recent work has empirically shown that the superior performance of the automated data augmentation methods stems from increasing the diversity of augmented data [4, 5]. However, two factors regarding the diversity of augmented data are still missing: 1) the explicit definition (and thus measurement) of diversity and 2) the quantifiable relationship between diversity and its regularization effects. To bridge this gap, we propose a diversity measure called Variance Diversity and theoretically show that the regularization effect of data augmentation is promised by Variance Diversity. We validate in experiments that the relative gain from automated data augmentation in test accuracy is highly correlated to Variance Diversity. An unsupervised sampling-based framework, **DivAug**, is designed to directly maximize Variance Diversity and hence strengthen the regularization effect. Without requiring a separate search process, the performance gain from DivAug is comparable with the state-of-the-art method with better efficiency. Moreover, under the semi-supervised setting, our framework can further improve the performance of semi-supervised learning algorithms compared to RandAugment, making it highly applicable to real-world problems, where labeled data is scarce. The code is available at* `https://github.com/warai-0toko/DivAug`.

1. Introduction

Data augmentation is a technique to create synthetic data from existing data with controlled perturbation. For example, in the context of image recognition, data augmentation refers to applying image operations, *e.g.*, cropping and flip-ping, to input images to generate augmented images, which have labels the same as their originals. In practice, data augmentation has been widely used to improve the generalization in deep learning models and is thought to encourage model insensitivity towards data perturbation [19, 14, 16]. Although data augmentation works well in practice, designing data augmentation strategies requires human expertise, and the strategy customized for one dataset often works poorly for another dataset. Recent efforts have been dedicated to automating the design of augmentation strategies. It has been shown that training models with a learned data augmentation policy may significantly improve test accuracy [20, 28, 5, 15, 13].

However, we do not yet have a good theory to explain how data augmentation improves model generalization. Currently, the most well-known hypothesis is that data augmentation improves generalization by imposing a regularization effect: it regularizes models to give consistent outputs within the vicinity of the original data, where the vicinity of the original data is defined as the space that contains all augmented data after applying operations that do not drastically alter image features [27, 6, 23]. Meanwhile, previous automated data augmentation works claim that the performance gain from applying learned augmentation policies arises from the increase in diversity [4, 5, 15]. However, the "diversity" in the claims remains a hand-waving concept: it is evaluated by the number of distinct sub-policies utilized during training or visually evaluated from a human perspective. Without formally defining diversity and its relation to regularization, the augmentation strategies can only be evaluated indirectly by evaluating the models trained on the augmented data, which may cost thousands of GPU hours [4]. It motivates us to explore the possibility of using an explicit diversity measure to quantify the regularization effect of the augmented data may have on the model. Thus, in this way we can directly maximize the diversity of the augmented data to strengthen the regularization effect to improve the generalization of the model.

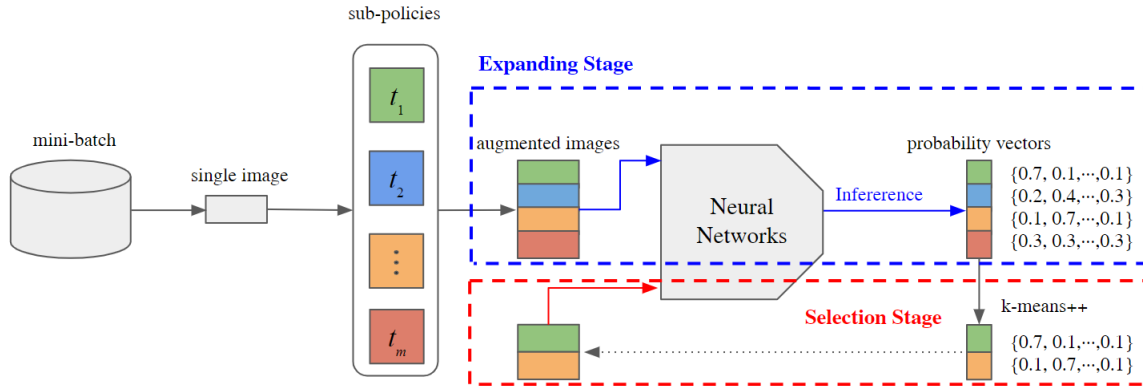
*The first two authors contributed equally to this paper.

Figure 1: The DivAug framework overview. At the expanding stage, each data in the mini-batch is augmented by multiple randomly generated sub-polices. Notice the probability vectors of these augmented data are also obtained. At the selection stage, k-means++ seeding algorithm is used to sub-sample a subset of augmented data whose probability vectors are far apart from each other and thus diversifies the augmented data. Then the sampled data is used to train the model.

To bridge the gap, in this paper we propose a new diversity measure, called Variance Diversity, to quantify the diversity of augmented data. We show that the regularization effect of data augmentation is promised by Variance Diversity. Our measure is motivated by the recent theoretical result that after applying augmented data to train the model, the loss implicitly contains a data-driven regularization term that is in proportion to the variance of probability vectors, where probability vectors are the outputs from models trained with the augmented data [6]. Specifically, we measure the diversity of a set of augmented data by the variance of their corresponding probability vectors. Based on the measure, we propose a plug-in automated data augmentation framework named **DivAug**, which can plug in the standard training process without requiring a separate search process. As illustrated in Figure 1, the framework has two stages: the expanding stage, where we randomly generate several augmented data for each original input data, and the selection stage, where we sub-sample a subset of augmented data and feed them to train the model. Specifically, at the selection stage, for each image, we sub-sample a subset of augmented images with high diversity by applying the k-means++ seeding algorithm [1], where the augmented data accompanied with probability vector which is far away from that of the original data is sampled with high probability. Following the mathematical derivation, the regularization effect increases with the diversity of the augmented data. Consequently, the stronger regularization effect can lead to better model generalization, which is observed in terms of improved model performance. Our main contributions can be summarized as follows:

- We propose a new measure for quantifying the diversity of augmented data. We validate in our experiments that the relative gain in the accuracy of a model after applying data augmentation is highly correlated to our

proposed measure.

- Based on the proposed measure, we design a sampling-based framework to explicitly maximize diversity. Without requiring a separate search process, the performance gain from DivAug is comparable to the state-of-the-art method with better efficiency.

- Our method is unsupervised and can plug in the standard training process. We show that our method can further boost the performance of the semi-supervised learning algorithm, making it highly applicable to real-world problems, where labeled data is scarce.

2. Related Work

Recently, AutoAugment (AA) [4] has been proposed to automatically search for augmentation policies from a dataset. Specifically, AutoAugment utilizes a recurrent neural network (RNN) as the controller to find the best policy in a separate search process on a small proxy task (smaller model size and dataset size). Once the search process is over, the learned policies are transferred to the target task and fixed during the whole training process. These learned augmentation policies significantly improve the generalization of deep models [4]. However, its search time is huge: it costs roughly 5,000 GPU hours to search for the best policies on a smaller dataset they call "reduced CIFAR-10", which consists of 4,000 randomly chosen images.

Most of the following works adopted the AutoAugment search space and formulation with improved optimization algorithms [28, 20, 15, 13]. Population-based augmentation (PBA) [15] replaces the fixed policy with a dynamic schedule of policies evolving along with the training process. Fast AutoAugment (Fast AA) [20] proposes a "density match" method to accelerate the search process and

Table 1: Summary of automated data augmentation.

Method	non-fixed	without the separate search process	unsupervised	without proxy tasks
AA [4]	✗	✗	✗	✗
Fast AA [20]	✗	✗	✗	✓
PBA [15]	✓	✗	✗	✗
Adv. AA [28]	✓	✗	✗	✓
RA [5]	✗	✓	✓	✓
DivAug (this paper)	✓	✓	✓	✓

treats the augmented data as missing points in the training set. RandAugment (RA) [5] eliminates the separate search process by randomly applying augmentation sub-policies, which best resembles our work. Adversarial AutoAugment (Adv. AA) [28] achieves state-of-the-art results by utilizing an RNN controller to learn policies that could generate augmented data with higher loss. As shown in Table 1, we outline a general taxonomy of automated data augmentation methods, characterized by four core properties. *Non-fixed*: augmentation policies are dynamically changed along with the training process; *without the separate search process*: methods do not require a separate search process; *unsupervised*: methods do not require label information to find the best policy; and *without proxy tasks*: methods perform the search directly on target tasks.

3. Methodology

In this section, we introduce the design and implementation of DivAug. First, we describe our search space in Section 3.1. Then we mathematically show that after employing augmented data, the training loss implicitly contains a data-driven regularization term that is in proportion to the variance of probability vectors (Section 3.2). Subsequently, we propose to measure the diversity of a set of augmented data by the variance of their corresponding probability vectors. Based on the measure, we derive a sampling-based automated data augmentation method to explicitly maximize the diversity of augmented data (Section 3.3).

3.1. Search Space

We adopt the basic structure of the well-designed search space introduced in AutoAugment [4]. There are totally 16 image operations in our search space, including `Sharpness`, `ShearX/Y`, `TranslateX/Y`, `Rotate`, `AutoContrast`, `Invert`, `Equalize`, `Solarize`, `Posterize`, `Color`, `Brightness`, `Cutout` [8], `Sample Pairing` [17], and `Contrast`. Let $\mathcal{O} = \{\texttt{Sharpness}, \cdots, \texttt{Contrast}\}$ be the set of all available operations. Each operation $\mathrm{op} \in \mathcal{O}$ has two parameters: p, the probability of applying the operation; and m, the magnitude of the operation. To avoid creating confusion in notations, we use $\overline{\mathrm{op}}(\cdot; m)$ to represent image transformation specified by op, with magnitude m. Given an image x, the operation $\mathrm{op}(x; p, m)$ is defined as:

$$\mathrm{op}(x; p, m) = \begin{cases} \overline{\mathrm{op}}(x; m), & \text{with probability } p. \\ x, & \text{with probability } 1 - p. \end{cases}$$

Each operation comes with a maximum range of magnitudes to avoid extreme image transformations. For example, `Rotate` operation is only allowed to rotate images at most 30 degrees. The maximum range of magnitude for each operation is set to be the same as those reported in the AutoAugment. Meanwhile, we normalize the magnitude parameter m to within $[0, 1]$, where 1 stands for the maximum acceptable magnitude. One example for illustrating the operation is shown in Figure 2.

In general, previous automated data augmentation methods search for the top augmentation policy, which is a set of five sub-policies, with each sub-policy consisting of two operations to be applied to the original images in sequence. Let t be the sub-policy that consists of two consecutive operations, namely, $t(x) = \mathrm{op}_2(\mathrm{op}_1(x; p_1, m_1); p_2, m_2)$. For the sake of description convenience, we simplify the notation as $t := \mathrm{op}_2 \circ \mathrm{op}_1$. Given the search space, previous automated data augmentation methods explore and rank the possible policy candidates in a separate search process. Once the search process is over, the top five policies are collected to form a single final policy, which is a set containing 25 distinct sub-policies. The final policy is fixed throughout the training process. For each image in a mini-batch, only one sub-policy will be randomly selected to be applied [4].
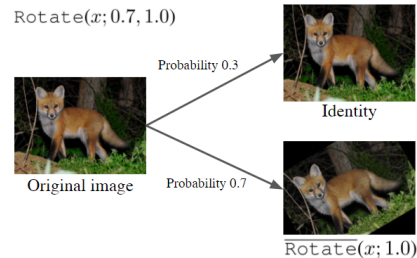

Figure 2: The schema of operation `Rotate`($\cdot; 0.7, 1.0$), where 1.0 is the normalized magnitude of the operation. Notice `Rotate`($\cdot; 0.7, 1.0$) denotes rotating the image by 30 degrees with the probability of 0.7.

However, the fixed policy may be sub-optimal due to

the following two factors. First, there does not exist a sub-policy universally better than all other sub-policies throughout the training process [10, 15, 28]. For example, sub-policies that can reduce generalization error at the end of training is not necessarily a good sub-policy at the initial phase [11]. Second, the choices (hence diversity) of the augmented data is limited by the fixed set of unique sub-policies. From the above analysis, we design our search space similar to the AutoAugment's search space with two differences. First, inspired by Fast AutoAugment [20], to introduce more stochasticity, we relax both the probability p and magnitude m as continuous parameters with value range $[0, 1]$. Second, the final policy in our search space is defined as the universal set that contains all the possible sub-policies. In contrast, the final policy in other work's search space is set to a fixed set of 25 unique sub-policies. We note that RandAugment [5] samples the sub-policies uniformly over the search space similar to ours. The major distinctions in RandAugment are 1) the magnitude parameter m is fixed discrete integer value, 2) the probability parameter p is fixed to $\frac{1}{K}$. That means RandAugment *always* applies operations on the original data*.

3.2. Regularization Effects of Data Augmentation

We derive the regularization effect of data augmentation following from the theoretical analysis in [6]. We start by introducing the setting and notations of representation learning. Consider a neural network $f_\theta(x)$ parameterized by θ. f_θ map the input x into a vector representation $f_\theta(x) \in \mathbb{R}^D$ with D output dimensions. We aim to minimize loss functions $l : \mathbb{R}^D \times \mathbb{R} \to \mathbb{R}$ over a dataset $\{(x_i, y_i)\}_{i=1}^N$, where $y_i \in \{1, \cdots D\}$. Let $\hat{p}(y|x) = \texttt{Softmax}(f_\theta(x))$ be the probability vector, where the $\texttt{Softmax}$ function is used to normalize $f_\theta(x)$ into a probability distribution. We denote the loss function to be minimized as $L = \sum_{i=1}^N L_i$, where $L_i = l(\hat{p}(y|x_i), y_i)$. We denote the gradient of l with respect to the first argument as $l' \in \mathbb{R}^D$. Similarly, we use $l'' \in \mathbb{R}^{D \times D}$ to represent the Hessian matrix of l with respect to the first argument. We use t to represent the sub-policy, and T is the set of all available sub-policies. x_i^t is the augmented data in the vicinity of x_i obtained by applying t to x_i. We use $\langle \cdot, \cdot \rangle$ to denote inner-product. For a set $\mathcal{S}$, we use $|\mathcal{S}|$ to represent its cardinality. With these notations, after applying data augmentation, the new loss function becomes:

$$\hat{L}_i = \mathbb{E}_{t \sim \mathrm{T}}[l(\hat{p}(y|x_i^t), y_i)]. \tag{1}$$

Suppose data augmentation does not significantly modify the feature map. Using the first order Taylor approxima-

tion, we can expand Equation (1) around point ψ_i:

$$\hat{L}_i \approx l(\psi_i, y_i) +$$
$$\mathbb{E}_{t \sim \mathrm{T}}[\langle \hat{p}(y|x_i^t) - \psi_i, l'(\psi_i, y_i)\rangle]. \tag{2}$$

The second term in Equation (2) can be cancelled by picking $\psi_i = \mathbb{E}_{t \sim \mathrm{T}} \hat{p}(y|x_i^t)$, *i.e.*, ψ_i is the averaged probability vector of all samples within the vicinity of x_i. If we further expand Equation (1) around point $\psi_i = \mathbb{E}_{t \sim \mathrm{T}} \hat{p}(y|x_i^t)$ by considering the second order term, we have:

$$\hat{L}_i \approx l(\psi_i, y_i) + \frac{1}{2}\mathbb{E}_{t \sim \mathrm{T}}[\Delta_i^\top l''(\psi_i, y_i)\Delta_i]. \tag{3}$$

$\Delta_i := \hat{p}(y|x_i^t) - \psi_i$ is the difference between the probability vector $\hat{p}(y|x_i^t)$ referring to the augmented data x_i^t, and the averaged probability vector ψ_i. The second term in Equation (3) is so called the "data-driven regularization term", which is exact the variance of the probability vector $\hat{p}(y|x_i^t)$, weighted by $l''(\psi_i, y_i)$. That means employing augmented data imposes a regularization effect by implicitly controlling the variance of model's outputs.

3.3. The DivAug Framework

To establish the relationship between the diversity of augmented data and their regularization effect, we propose a new diversity measure, called Variance Diversity, for the augmented data whose regularization effect can be quantified. Based on this, we derive a sampling-based framework that explicitly maximizes the Variance Diversity of the augmented data.

3.3.1 Diversity Measure of Augmented Data

We start by proposing a new diversity measure for augmented data, whose regularization effect can be quantified.
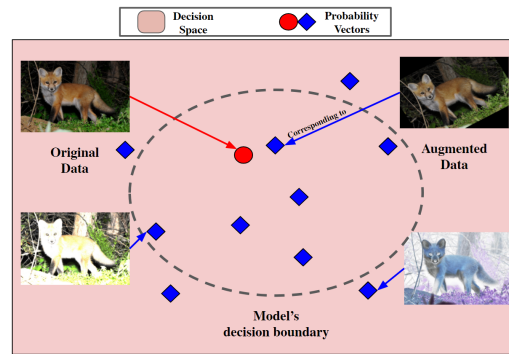

Figure 3: An example to illustrate the diversity between augmented data. DivAug explicitly looks for augmented data whose corresponding probability vectors are far away from each other in the decision space.

From Equation (3), after training models on augmented data, a data-driven regularization term can be decomposed

---

*Although RA always applies operations, RA may keep the original image unchanged since its search space contains an $\texttt{identity}$ operation.

Algorithm 1: DivAug

Input: input image x; model f_θ; all possible operations $\mathcal{O}=\{\texttt{Sharpness},\cdots,\texttt{Contrast}\}$

Parameters: the number of augmented images per input image E; the number of selected augmented images per input image used for training S

Output: $\mathcal{S}$:= a set of S augmented images of input image x

1 for $j = 1,\cdots,E$ **do**
2   Sample operations $\text{op}_1, \text{op}_2 \sim \mathcal{O}$ uniformly at random
3 $p_1 \sim \text{Uniform}(0,1); p_2 \sim \text{Uniform}(0,1)$
4 $m_1 \sim \text{Uniform}(0,1); m_2 \sim \text{Uniform}(0,1)$
5   Get sub-policy $t_j := \text{op}_1(\,\cdot\,; p_1, m_1) \circ \text{op}_2(\,\cdot\,; p_2, m_2)$
6   Generate $x^{t_j} = t(x)$
7   Compute $\hat{p}(y|x^{t_j}) = \texttt{Softmax}(f_\theta(x^{t_j}))$
8 end
9 Generate a set of augmented images $\mathcal{S}$ of size S, which is a random subset of $\{x^{t_j}, j = 1,\cdots,E\}$, using k-means++ seeding algorithm on $\{\hat{p}(y|x^{t_j}) : j = 1,\cdots,E\}$
10 return $\mathcal{S}$

from the loss function. From above, we quantify the diversity of a set of augmented data by the variance of their corresponding probability vectors. Formally, given a model f_θ, for a set of augmented data $\mathcal{S} = \{x^{t_j}\}_{j=1}^{S}$, where x^{t_j} is generated from the same original data x by applying different sub-policy t_j, we define the diversity of $\mathcal{S}$ as:

$$\mathcal{D}(\mathcal{S}) = \mathbb{E}_{x^{t_j} \in \mathcal{S}}\Delta^\top \Delta. \tag{4}$$

$\hat{p}(y|x^{t_j}) := \texttt{Softmax}(f_\theta(x^{t_j}))$ is the probability vector corresponding to x^{t_j}, and $\Delta = \hat{p}(y|x^{t_j}) - \mathbb{E}_{x^{t_j} \in \mathcal{S}}\hat{p}(y|x^{t_j})$. If CrossEntropy is used as the loss function, then the Hessian matrix $l''(\psi_i, y_i)$ is a diagonal matrix, where the elements on the diagonal are all zero, except for the one corresponding to the true label. This implies that under the supervised setting, only the variance of the probability associated with the true label will be penalized. We can extend this penalty effect to unsupervised domain by setting $l''(\psi_i, y_i)$ in Equation (3) as the identity matrix. In this way, Equation (3) penalizes the variance of the probability associated with any class. We note that this is essentially the consistency regularization, which is one of the key techniques in semi-supervised learning and self-supervised learning, which encourages the model to produce similar probability vectors when the input data is perturbed by noise [24, 2]. Moreover, if $l''(\psi_i, y_i)$ in Equation (3) is set as the identity matrix, the diversity of augmented data is exact the data-driven regularization term in Equation (3).

According to Equation (4), we name our diversity measure "Variance Diversity". We note that this is a unsupervised model-specific measure, which depends only on the model prediction without involving any label information. Intuitively, as illustrated in Figure 3, if a set of augmented data has large Variance Diversity, that means their corresponding probability vectors are far away from each other.

Therefore, it is harder for models to give consistent predictions for diversely augmented data. This forces the models to generalize over the vicinity of original data.

3.3.2 Design of DivAug

According to the definition of Variance Diversity and Equation (3), the increase of Variance Diversity directly strengthens the regularization effect of augmented data. Based on this insight, our DivAug framework generates a set of diversely augmented data and minimizes the loss over them. Specifically, DivAug consists of two stages: the expanding stage and the selection stage. At the expanding stage, for each original data x_i, we first randomly generate a set of sub-policies $\{t_j\}_{j=1}^{E}$, where $\{x_i^{t_j}\}_{j=1}^{E}$ are the set of augmented data $x_i^{t_j}$ corresponding to t_j. The second stage is the selection stage, where we sub-sample a subset of augmented data $\mathcal{S}_i \subset \{x_i^{t_j}\}_{j=1}^{E}$, where $|\mathcal{S}_i| = S < E$. Then we feed the selected augmented data to the model. Our DivAug framework is illustrated in Figure 1. Formally, with the notations introduced in Section 3.2 and Section 3.3.1, given f_θ, we minimize the following objective:

$$\min_\theta \quad \frac{1}{N}\sum_{i=1}^{N}[\frac{1}{S}\sum_{x_i^t \in \mathcal{S}_i} l(\hat{p}(y|x_i^t), y)], \tag{5}$$

$$\text{s.t.} \quad \mathcal{S}_i = \underset{\substack{\mathcal{S}_i \subset \{x_i^{t_j}\}_{j=1}^{E}, \\ |\mathcal{S}_i| = S}}{\arg\max} \; \mathbb{E}_{x_i^{t_j} \in \mathcal{S}_i}\Delta_i^\top \Delta_i. \tag{6}$$

where $\Delta_i = \hat{p}(y|x_i^{t_j}) - \mathbb{E}_{x_i^{t_j} \in \mathcal{S}_i}\hat{p}(y|x_i^{t_j})$. From Equation (6), we target at selecting a subset of augmented data $\mathcal{S}_i$, whose corresponding probability vectors have maximum variance. Unfortunately, getting the solution of Equation (6)

poses a significant computational hurdle. Instead of computing the optimal solution, we efficiently sample $\mathcal{S}_i$ with the k-means++ seeding algorithm [1], which is originally made to generate a good initialization for k-means clustering. k-means++ seeding selects centroids by iteratively sampling points in proportion to their squared distances from the closest centroid that has been chosen. Here, we define the distance between a pair of probability vector as their Euclidean distance. Therefore, k-means++ samples a subset of augmented data where their probability vectors are far apart from each other, which practically leads to a large Variance Diversity. For more details, the k-means++ seeding algorithm is shown in Algorithm 2 in the Appendix A. We show the algorithm of DivAug in Algorithm 1 and remark that the operation is randomly generated. There are two hyperparameters in Algorithm 1. Namely, the number of augmented images per input image E, and the number of selected augmented images per input image used for training S. Moreover, the two hyperparameters S and E do not need to be tuned on proxy tasks and can be chosen according to available computation resources. Similar to RandAugment, DivAug is a sampling-based method that does not require a separate search process. Note that there is no label information involved in Algorithm 1, which means DivAug is suitable for both semi-supervised learning and supervised learning.

4. Experiments

Our experiments aim to answer the following research questions:

- **RQ1**. What is the effect of Variance Diversity on model generalization?

- **RQ2**. How effective is the proposed DivAug compared with other automated data augmentation methods under the supervised settings?

- **RQ3**. How well does DivAug improve the performance of semi-supervised learning algorithms?

4.1. Experimental Settings

Below, we first introduce the datasets and the default augmentation method for them. Then, we will introduce the hyperparameter setting of Divaug (S and E in Algorithm 1) , and the baseline methods for comparison.

We adopt four benchmark datasets for evaluating our proposed method: CIFAR-10, CIFAR-100, SVHN and ImageNet. These four datasets are processed based on the way and codes provided in [4]. The basic statistics of these four datasets and the default data augmentation for them are summarized in Appendix. For DivAug, we set $E = 8$ and $S = 4$ for the experiments in Section 4.2 and 4.3, excluding the ImageNet experiment. For ImageNet, we set $E = 4$ and

$S = 2$ due to limited resources. For the semi-supervised learning experiment, we set $E = 4$ and $S = 2$. We did not tune these two hyperparameters, and we choose them mainly according to the available GPU memory.

The methods for comparison are as below: We compare Algorithm 1 with AutoAugment (AA) [4], Fast AutoAugment (Fast AA) [20], Population Based Augmentation (PBA) [15], RandAugment (RA) [5], and Adversarial AutoAugment (Adv. AA) [28]. For each image, the augmentation policy proposed by different methods and the default augmentation are applied in sequence.w

4.2. Correlation Between Variance Diversity and Generalization

To answer **RQ1**, we calculate the Variance Diversity of augmented data generated by AA, Fast AA, RA, the default augmentation introduced in Section 4.1, and DivAug†. Then, we report the test accuracy of models trained on augmented data generated by different methods.

Because Variance Diversity is an unsupervised, model-specific measure, for a fair comparison, we first train a Wide-ResNet-40-2 model on CIFAR-10 without applying any data augmentation methods. Then we use it as the f_θ in Equation (4) to evaluate all different automated data augmentation methods. To verify the correlation between generalization and Variance Diversity, we calculate the Variance Diversity of augmented data as follows: for each image in the training set, an automated augmentation method is used to randomly generate four augmented images. Then we calculate the Variance Diversity of these four images according to Equation (4). We report the averaged Variance Diversity over the entire training set in Figure 4.
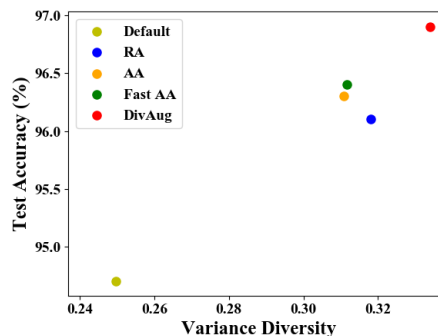

Figure 4: **The performance gain is positively correlated to Variance Diversity.** In general, almost all points lies near the diagonal, and the relative gain in test accuracy increases with larger Variance Diversity.

Figure 4 demonstrates the performance gain and Variance Diversity are positively correlated (the detailed test ac-

---

†We do not include Adv. AA because the official code is not released. For PBA, the official code is based on Ray and hard to migrate our code-base for a fair comparison.

curacy is shown in the first row of Table 2). As shown in the figure, all automated data augmentation methods could improve the Variance Diversity of augmented data over the default augmentation. Specifically, AA and Fast AA has small Variance Diversity. It makes sense because both of them try to minimize the distribution shift of the augmented data from the original distribution. For example, Fast AA treats the augmented data as the missing point in the training set. As a result, for CIFAR-10, all of the reported sub-policy proposed by AA and Fast AA do not contain the counter-intuitive operation `SamplePair` [4, 20], which limits the Variance Diversity of the augmented data generated by them. In contrast, DivAug has the largest Variance Diversity because it tries to explicitly maximize the Variance Diversity of the augmented data. Notice RA has larger Variance Diversity compared to AA and Fast AA. This might be a result of RA randomly sample operations. As a result, RA samples more distinct sub-policies than AA and Fast AA do and leads to larger diversity. Here we remark that although RA has larger Variance Diversity compared to AA and Fast AA, the model's relative gain in accuracy is smaller compared to those of AA and Fast AA. We provided a detailed analysis in the Appendix D.

We also present a simple case study in Appendix C Figure 6, where DivAug's candidate images are obtained by only applying the single transform `Rotate` with fixed probability parameter p (the magnitude parameter remains random). As shown in Figure 6, Variance Diversity and generalization are generally correlated.

4.3. The Effectiveness of DivAug Under the Supervised Settings

The main propose of automated data augmentation is to further improve the generalization of models over traditional data augmentation techniques. To answer **RQ2**, we compare our proposed method with several baselines under the supervised learning settings.

4.3.1 Experiment on CIFAR-10 and CIFAR-100

Following [4, 20, 5], we evaluate our proposed method with the following models: Wide-ResNet-28-10, Wide-ResNet-40-2 [26], Shake-Shake (26 2x96d) [9], and PyramidNet+ShakeDrop [25, 12]. The details of hyperparameters are shown in Appendix Table 5.

CIFAR-10 Results: In Table 2, we report the test accuracy of these models. For all of these models, our proposed method can achieve better performance compared to previous methods. We achieve 0.7%, 0.8%, 0.7%, 0.8% improvement on Wide-ResNet-28-10 compared to AA, Fast AA, PBA and RA, respectively. Overall, DivAug significantly improves the performances over baselines while achieves comparable performances to those of Adv. AA.

The effect of k-means++ : To check the effect of k-means++ in DivAug, we compare the performance of Wide-ResNet-28-10 with DivAug and that with the random baseline in Appendix C Table 6. The random baseline here randomly picks S augmented images from E candidates for training. Also, the magnitude m and probability p are also randomly picked. As shown in Table 6, DivAug is significantly better than the random baseline. Moreover, to understand the effect of k-means++ and how DivAug improves the test accuracy over RA, we further visualize the distribution of sub-policies selected by DivAug with Wide-ResNet-40-2 on CIFAR-10 over the training process. As shown in Figure 5, we observe that the percentages of some operations picked from the sampled sub-policies, such as `TranslateY`, `ShearY`, `Posterize`, and `SampleParing`, gradually increase along with the training process. In contrast, some color-based operation, such as `Invert`, `Brightness`, `AutoContrast`, and `Color`, gradually decrease along with the training process. This behavior is consistent with the discovery that there does not exist an operation beating all other operations throughout the training process [15, 28]. Also, the average probability of applying operations in the selected sub-policies slowly increases with the training process. That means DivAug tends to mildly shift the distribution of augmented images away from the original one over the training process. From above, it suggests that the sub-policies selected by DivAug evolve throughout the training process.
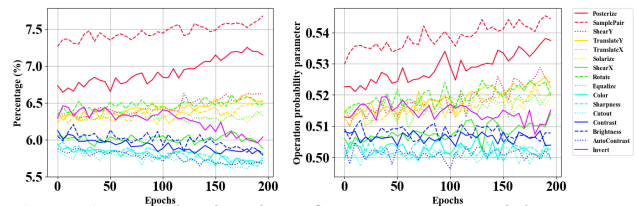

Figure 5: **The distribution of selected sub-policies evolves along with the training process.** (a) The statistics of sub-policies selected by DivAug. (b) The averaged probability of applying operations in sub-policies selected by DivAug.

Training Efficiency Analysis: DivAug is estimated to be significantly faster than Adv. AA for the following reasons. Following the time cost metric in [23], we estimate the inference cost (see Algorithm 1 line 7) equals half of the training cost. Under the setting of $E = 8$ and $S = 4$, DivAug additionally generates four times more augmented data for training. In contrast, Adv. AA needs to generate eight times more augmented data to achieve the results reported in Table 2. Moreover, it also needs a separate phase to search for the best policy. Although the search time for Adv. AA is not reported in [28]. The estimated costs are summarized in Table 3.

CIFAR-100 Results: As shown in Table 2, DivAug generally achieves non-trivial performance gain over all other

Table 2: Test accuracy (%) on CIFAR-10 and CIFAR-100. For ImageNet, we report the validation accuracy (%). We compare our method with the default data augmentation (Baseline), AA, Fast AA, PBA, RA, and Adv. AA. Our results are averaged over four trials except ImageNet.

Dataset	Model	Baseline	AA	Fast AA	PBA	RA	Adv. AA	DivAug
CIFAR-10	Wide-ResNet-40-2	94.7	96.3	96.4	-	96.1	-	**96.9**±.1
	Wide-ResNet-28-10	96.1	97.4	97.3	97.4	97.3	**98.1**	**98.1**±.1
	Shake-Shake (26 2x96d)	97.1	98.0	98.0	98.0	98.0	**98.1**	**98.1**±.1
	PyramidNet+ShakeDrop	97.3	98.5	98.3	98.5	98.5	**98.6**	98.5±.1
CIFAR-100	Wide-ResNet-40-2	74.0	79.3	79.4	-	-	-	**81.3**±.3
	Wide-ResNet-28-10	81.2	82.9	82.7	83.3	83.3	**84.5**	84.2±.2
	Shake-Shake (26 2x96d)	82.9	85.7	85.1	84.7	-	**85.9**	85.3±.2
SVHN	Wide-ResNet-28-10	96.9	98.1	-	-	**98.3**	-	**98.3**±.3
ImageNet	ResNet-50	76.3	77.6	77.6	-	77.6	**79.4**	78.0

Table 3: Comparison of the total cost of DivAug and Adv. AA on CIFAR-10 relative to RA. The training cost of Adv. AA is cited from [28].

	RA	Adv. AA	DivAug
Training($\times$)	1.0	8.0 + Search Cost	4.5

methods excluding Adv. AA. However, we note that DivAug does not require label information or a separate search process. Also, DivAug is significantly faster than Adv. AA.

4.3.2 Experiment on ImageNet

Following [4, 20, 5], we select ResNet-50 [14] to evaluate our proposed method. The details of the hyperparameters are shown in Appendix Table 5. As shown in Table 2, DivAug outperforms other baselines except Adv. AA. We remark that due to the limited resources, the two hyperparameters in Algorithm 1 are set to $E = 4$ and $S = 2$, respectively. The performance gain from DivAug is expected to be further improved with larger E and S.

4.4. The Effectiveness of DivAug Under the Semi-Supervised Setting

One of the key techniques in semi-supervised learning [3] (SSL) is consistency regularization, which encourages the model to produce similar probability vectors when the input data is perturbed by noise. It has been proven that the augmented data produced by state-of-the-art automated methods can serve as a superior source of noise under the consistency regularization framework [24, 22]. Specifically, UDA [24] utilizes RA as the source of perturbation and achieves non-trivial performance gain. Also, it has been theoretically shown that the success of UDA stems from the diversity of augmented data generated by RA [24].

However, most automated data augmentation methods require label information to search for the best policy. Thus,

this prerequisite limits their application in SSL. In contrast, our proposed method is suitable for SSL because it is unsupervised and tries to explicitly maximize diversity. This leads to the following question: can SSL benefit from our proposed DivAug (**RQ3**)? To answer this question, following UDA, we change the source of perturbation from RA to DivAug (detailed hyperparameters are shown in the Appendix). Here, we report the averaged results over four random trials. As shown in Table 4, DivAug can further boost the performance of UDA under different settings. Moreover, the performance gap grows larger when there is less labeled data available. This might be because, when there is limited labeled data, the regularization effect brought by diversity plays a much bigger role in model performance.

Table 4: Error rate (%) comparison on CIFAR-10 with 1000, 2000, and 4000 labeled data. The architecture is Wide-ResNet-28-2. For fair comparison, we reproduced the UDA(RA)* result by ourselves using the same codebase.

Methods	CIFAR-10		
	1000	2000	4000
UDA(RA)*	7.37±0.15	6.50 ±0.14	5.44±0.15
UDA(DivAug)	**6.94** ±0.12	**6.26**±0.15	**5.40**±0.12

5. Conclusion

In this work, we propose a new diversity measure called Variance Diversity by investigating the regularization effect of data augmentation. We validate in experiments that the performance gain from automated data augmentation is highly correlated to Variance Diversity. Based on this measure, we derive the DivAug framework to explicitly maximize Variance Diversity during training. We demonstrate our proposed method has the practical utility of achieving better performance without the need to search for top policies in a separate phase. Therefore, DivAug can benefit both the supervised tasks and the semi-supervised tasks.

References

[1] David Arthur and Sergei Vassilvitskii. k-means++ the advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035, 2007.

[2] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, pages 5049–5059, 2019.

[3] Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

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

[5] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020.

[6] Tri Dao, Albert Gu, Alexander J Ratner, Virginia Smith, Christopher De Sa, and Christopher Ré. A kernel theory of modern data augmentation. *Proceedings of machine learning research*, 97:1528, 2019.

[7] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

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

[9] Xavier Gastaldi. Shake-shake regularization. *arXiv preprint arXiv:1705.07485*, 2017.

[10] Aditya Sharad Golatkar, Alessandro Achille, and Stefano Soatto. Time matters in regularizing deep networks: Weight decay and data augmentation affect early learning dynamics, matter little near convergence. In *Advances in Neural Information Processing Systems*, pages 10678–10688, 2019.

[11] Raphael Gontijo-Lopes, Sylvia J Smullin, Ekin D Cubuk, and Ethan Dyer. Affinity and diversity: Quantifying mechanisms of data augmentation. *arXiv preprint arXiv:2002.08973*, 2020.

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

[13] Ryuichiro Hataya, Jan Zdenek, Kazuki Yoshizoe, and Hideki Nakayama. Faster autoaugment: Learning augmentation strategies using backpropagation. *arXiv preprint arXiv:1911.06987*, 2019.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[15] Daniel Ho, Eric Liang, Xi Chen, Ion Stoica, and Pieter Abbeel. Population based augmentation: Efficient learning of augmentation policy schedules. In *International Conference on Machine Learning*, pages 2731–2741. PMLR, 2019.

[16] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[17] Hiroshi Inoue. Data augmentation by pairing samples for images classification. *arXiv preprint arXiv:1801.02929*, 2018.

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

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[20] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. In *Advances in Neural Information Processing Systems*, pages 6665–6675, 2019.

[21] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[22] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint arXiv:2001.07685*, 2020.

[23] Sen Wu, Hongyang R Zhang, Gregory Valiant, and Christopher Ré. On the generalization effects of linear transformations in data augmentation. *arXiv preprint arXiv:2005.00695*, 2020.

[24] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*, 2019.

[25] Yoshihiro Yamada, Masakazu Iwamura, Takuya Akiba, and Koichi Kise. Shakedrop regularization for deep residual learning. *IEEE Access*, 7:186126–186136, 2019.

[26] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

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

[28] Xinyu Zhang, Qiang Wang, Jian Zhang, and Zhao Zhong. Adversarial autoaugment. *arXiv preprint arXiv:1912.11188*, 2019.