AlphaMatch: Improving Consistency for Semi-supervised Learning with Alpha-divergence

Chengyue Gong, Dilin Wang, Qiang Liu

University of Texas, Austin

Abstract

Semi-supervised learning (SSL) is a key approach toward more data-efficient machine learning by jointly leverage both labeled and unlabeled data. We propose AlphaMatch*, an efficient SSL method that leverages data augmentations, by efficiently enforcing the label consistency between the data points and the augmented data derived from them. Our key technical contribution lies on: 1) using alpha-divergence to prioritize the regularization on data with high confidence, achieving similar effect as FixMatch [32] but in a more flexible fashion, and 2) proposing an optimization-based, EM-like algorithm to enforce the consistency, which enjoys better convergence than iterative regularization procedures used in recent SSL methods such as FixMatch, UDA, and MixMatch. AlphaMatch is simple and easy to implement, and consistently outperforms prior arts on standard benchmarks, e.g. CIFAR-10, SVHN, CIFAR-100, STL-10. Specifically, we achieve 91.3% test accuracy on CIFAR-10 with just 4 labelled data per class, substantially improving over the previously best 88.7% accuracy achieved by FixMatch.*

1. Introduction

Semi-supervised learning (SSL) [7] is a powerful paradigm for leveraging both labeled and unlabeled data jointly in machine learning (ML). Effective SSL methods can help build accurate prediction out of very limited labeled data, and can also boost state-of-the-art performance and robustness in typical supervised learning by leveraging extra unlabeled data [see e.g., 20, 39, 41]. Due to the high cost of collecting labels and the availability of a vast amount of unlabeled data, breakthroughs in SSL can dramatically advance the application of ML in countless fields.

Recently, data augmentation has been shown a powerful tool for developing state-of-the-art SSL methods, including unsupervised data augmentation (UDA) [38], FixMatch [32], MixMatch [2], ReMixMatch [3] and Π-model [30]. All these methods are based on the similar idea of enforcing the consistency between the label of a data point and that of its perturbed version generated by data augmentation. This encourages the learned models to be invariant under given data augmentation transforms, hence incorporating inherent structures of the data into semi-supervised learning.

The performance of these algorithms can be critically influenced by what matching objective and matching algorithm are used to enforce the consistency. For the objective, UDA applies a KL divergence penalty to enforce the consistency uniformly on all the data points. More recently, FixMatch shows that it is useful to focus on matching the consistency on the high confidence data points with a *hard thresholding* approach, by applying regularization only on data with confidence higher than a threshold and use the hard label as the target. In terms of the matching algorithm, most of the existing methods, including UDA and FixMatch, are based on a similar iterative regularization procedure that uses the label distribution predicted from the previous iteration as the target for the next step. Although being simple and intuitive, a key problem is that this iterative procedure does not correspond to optimizing a fixed objective function, and hence may suffer from instability and non-convergence issues.

This work proposes two key algorithmic advances to improve the objective and algorithm for consistency matching in SSL: 1) we propose to use alpha-divergence to measure the label consistency. We show that, by using a large value of α in alpha-divergence, we can focus more on high confidence instances in a way similar to the hard-thresholded regularization of FixMatch, but in a more "soft" and flexible fashion. 2) We propose an optimization-based framework for consistency matching, which yields an EM-like algorithm with better convergence property than the commonly used iterative regularization procedures. By combining these two key techniques, our main algorithm *AlphaMatch* yields better SSL with more effective and stable consistency regularization.

Empirically, we find that AlphaMatch consistently outperforms recently-proposed SSL methods such as Fix-

Match, ReMixMatch, MixMatch, and UDA both in terms of accuracy and data efficiency, on various benchmarks including CIFAR-10, SVHN CIAFR-100, and STL-10. In particular, our method improves over the state-of-the-art method, FixMatch, across all the settings we tested. Our improvement is particularly significant when the labels are highly limited. For example, on CIFAR-10, we improve the 88.71%±3.35% accuracy of FixMatch to 91.35%±3.38% when only 4 labelled images per class are given.

2. Background: Semi-Supervision with Data Augmentation

We give a brief introduction to unsupervised data augmentation (UDA) [38] and FixMatch [32], which are mostly related to our work. Denote by $\mathcal{D}_s$ and $\mathcal{D}_u$ the labeled and unlabeled datasets, respectively. For a data point x in $\mathcal{D}_u$, let $\mathcal{P}_x$ be a distribution that prescribes a random perturbation or augmentation transformation on x that (with high probability) keeps the label of x invariant, such as rotation, shift, and cutout [12]. For learning a prediction model $p_\theta(y|x)$, UDA works by iteratively updating the parameter θ via

$$\theta_{t+1} \leftarrow \arg\min_\theta \Big\{ \mathcal{L}(\theta;\ \mathcal{D}_s) + \lambda \Phi(\theta; \theta_t, \mathcal{D}_u) \Big\} \text{ with}$$
$$\Phi(\theta;\ \theta_t, \mathcal{D}_u) = \mathbb{E}_{x\sim\mathcal{D}_u,\ x'\sim\mathcal{P}_x} \Big[\mathrm{KL}(p_{\theta_t}(\cdot \mid x) \ \|\ p_\theta(\cdot \mid x')) \Big], \tag{1}$$

where θ_t denotes the value at the t-th iteration, $\mathcal{L}(\mathcal{D}_s;\ \theta)$ is the typical supervised loss, e.g. the cross entropy loss, and $\mathrm{KL}(\cdot \ \|\ \cdot)$ denotes the Kullback–Leibler (KL) divergence. Here $\Phi(\theta; \theta_t, \mathcal{D}_u)$ can be viewed as a consistency regularization that enforces the label distribution of the augmented data $x' \sim \mathcal{P}_x$ to be similar to that of the original data x (based on the parameter θ_t at the previous iteration); λ is a regularization coefficient.

In practice, the optimization in (1) can be approximated by applying one step of gradient descent initialized from θ_t, yielding

$$\theta_{t+1} \leftarrow \theta_t - \epsilon \nabla_\theta \Big(\mathcal{L}(\theta;\ \mathcal{D}_s) + \lambda \Phi(\theta;\ \theta_t,\ \mathcal{D}_u) \Big) \Big|_{\theta=\theta_t}, \tag{2}$$

where ϵ is the step size. This procedure is closely related to VAT [26], in which augmented data x' is replaced by an adversarial example in a neighboring ball of x.

FixMatch improves UDA with ideas similar to the classical *pseudo-labeling* method [15]. FixMatch replaces the "soft label" $p_{\theta_t}(\cdot|x)$ with the corresponding "hard label" $\hat{y}_{\theta_t}(x) = \arg\max_y p_{\theta_t}(y \mid x)$ (a.k.a. pseudo-label), and turns on the regularization only when the confidence of the pseudo-label, estimated by $p_{\theta_t}(\hat{y}_{\theta_t}(x) \mid x)$, is sufficiently large:

$$\Phi(\theta;\ \theta_t, \mathcal{D}_u) := \mathbb{E}_{x\sim\mathcal{D}_u, x'\sim\mathcal{P}_x} \Big[\mathbb{I}\Big(p_{\theta_t}(\hat{y}_{\theta_t}(x) \mid x) \ge \tau \Big) \times \mathrm{KL}\Big(\hat{p}_{\theta_t}(\cdot \mid x) \ \|\ p_\theta(\cdot \mid x') \Big) \Big], \tag{3}$$

where $\hat{p}_{\theta_t}(y \mid x) := \delta(y = \hat{y}_{\theta_t}(x))$ and $\mathbb{I}(\cdot)$ is the indicator function and τ a threshold parameter (e.g., $\tau = 0.95$). This regularization has two important effects: 1) it upweights the hard label $\hat{y}_{\theta_t}(x)$ while discarding all the other labels from the regularization, and 2) it skips the data points with low confidence (i.e., $p_{\theta_t}(\hat{y}_{\theta_t}(x) \mid x) < \tau$).

Remark Note that the iterative procedure in (1)-(2) does not in general correspond to optimizing a fixed objective function, and hence does not guarantee to converge theoretically and may suffer from non-convergence practically. For example, we empirically observe that the performance of UDA tends to degenerate significantly when trained for many iterations when few labelled data is given, and FixMatch is sensitive to the choice of threshold τ. An alternative is to directly optimize the following objective function:

$$\min_\theta \Big\{ \mathcal{L}(\theta;\ \mathcal{D}_s) + \lambda \Phi(\theta;\ \theta,\ \mathcal{D}_u) \Big\}, \tag{4}$$

whose gradient descent yields

$$\theta_{t+1} \leftarrow \theta_t - \epsilon \nabla_\theta \Big(\mathcal{L}(\theta;\ \mathcal{D}_s) + \lambda \Phi(\theta;\ \theta,\ \mathcal{D}_u) \Big) \Big|_{\theta=\theta_t}. \tag{5}$$

The difference with (2) is that the gradient of $\Phi(\theta;\ \theta_t, \mathcal{D}_u)$ through θ_t is detached and dropped in (2), while the gradient of $\Phi(\theta;\ \theta,\ \mathcal{D}_u)$ in (5) needs to be taken for both θ. In Miyato et al. [26], Sohn et al. [32], Xie et al. [38] and all the related works, (2) is chosen over (5) for better empirical performance, likely because stopping the gradient encourages the information to pass from the clean data x to the augmented data x', but not the other way, so that the supervised objective is less interfered by the consistency regularization than in the direction optimization approach (5). A complete theoretical understanding of the benefit of stopping gradient is still an open question.

3. Our Method

We introduce our main method *AlphaMatch* (see Algorithm 1), which consists of two key ideas: i) we leverage alpha-divergence to enforce the label consistency between augmented and original data in SSL, which can benefit from

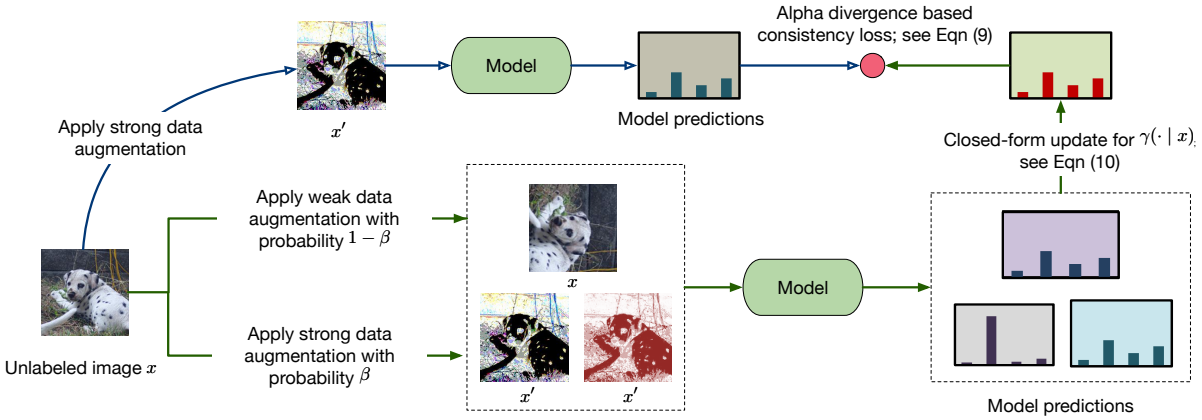


Figure 1. Diagram of our proposed semi-supervised learning algorithm. When augmentation, we use a hyper-parameter β to control a mixture of strong and weak augmentation. When calculating the loss, we use a EM-style update. 1) Solving a closed-from solution of averaging logits with alpha divergence. 2) Using alpha divergence to construct a label-consistency loss. Images are from ImageNet [11].

focusing more on high confidence data like FixMatch, but in a more smooth and flexible fashion (Section 3.1); ii) we introduce a convergent EM-like algorithm based on an optimization-based framework to replace the iterative procedure in (1)-(2), which allows us to enforce the label consistency much efficiently.

3.1. Matching with Alpha Divergence

We propose to use alpha-divergence in SSL consistency matching. Under the general framework of (1), this amounts to replace the consistency regularization with

$$\Phi(\theta;\ \theta_t, \mathcal{D}_u) = \mathbb{E}_{x\sim\mathcal{D}_u, x'\sim\mathcal{P}_x}\Big[D_\alpha(p_{\theta_t}(\cdot \mid x) \parallel p_\theta(\cdot \mid x'))\Big],$$

where $D_\alpha(\cdot \parallel \cdot)$ is the alpha divergence with $\alpha \in (0,1)\cup(1,\infty)$, defined as:

$$D_\alpha(p_{\theta_t}(\cdot|x) \,||\, p_\theta(\cdot|x')) = \frac{1}{\alpha(\alpha-1)}\Big(\mathbb{E}_{y\sim p_{\theta_t}(\cdot|x)}\left[\rho_{D_\alpha}(y|x)\right] - 1\Big),$$
$$\text{with } \rho_{D_\alpha}(y|x) := \left(\frac{p_{\theta_t}(y|x)}{p_\theta(y|x')}\right)^{\alpha-1}.$$

It is well-known that $D_\alpha(\cdot||\cdot)$ reduces to KL divergence when $\alpha \to 0$ or 1, as follows,

$$\lim_{\alpha\to 1} D_\alpha(p_{\theta_t}(\cdot|x) \parallel p_\theta(\cdot|x')) = \mathrm{KL}(p_{\theta_t}(\cdot|x) \parallel p_\theta(\cdot|x')),$$
$$\lim_{\alpha\to 0} D_\alpha(p_{\theta_t}(\cdot|x) \parallel p_\theta(\cdot|x')) = \mathrm{KL}(p_\theta(\cdot|x') \parallel p_{\theta_t}(\cdot|x)).$$

In general, the value of α critically influences the result of the algorithm. The regime of $\alpha > 1$ is of particular interest for our purpose, because it allows us to achieve a FixMatch-like effect but in a "soft way". This is because when α is large, the power term $\rho_{D_\alpha}(y|x)$ in $D_\alpha(\cdot||\cdot)$ tends to put a higher weight on the (x,y) pairs with large values of $p_{\theta_t}(y|x)$, and hence upweighting the importance the instances x with high confidence as well as their dominating labels y. This is similar to what FixMatch attempts to achieve in (3), except that the regularization is enforced in a different and more "soft" fashion, so that the instances with lower confidence and the less dominant labels still contribute to the loss, except with a lower degree.

It is useful to get further insights by examining the gradient of $D_\alpha(\cdot||\cdot)$, which equals

$$\nabla_\theta D_\alpha(p_{\theta_t}(\cdot|x) \parallel p_\theta(\cdot|x')) = -\frac{1}{\alpha}\mathbb{E}_{y\sim p_{\theta_t}(\cdot|x)}\Big[\rho_{D_\alpha}(y|x)\nabla_\theta \log p_\theta(y|x')\Big]. \quad (6)$$

When $\alpha = 1$ (corresponding to UDA), we have $\rho_{D_\alpha}(y|x) = 1$. The gradient of FixMatch is also similar but with $\rho_{D_\alpha}(y|x)$ replaced by

$$\rho_{\text{FixMatch}}(y|x) = \mathbb{I}\Big(\max_{y'} p_{\theta_t}(y'|x) \geq \tau \& y\ = \arg\max_{y'}(p_{\theta_t}(y'|x))\Big). \quad (7)$$

We can again see that both $\rho_{D_\alpha}(y|x)$ and $\rho_{\text{FixMatch}}(y|x)$ favor the data and label (x,y) with high confidence $p_{\theta_t}(y|x)$, but $\rho_{\text{FixMatch}}(y|x)$ does it in a more aggressive fashion. In addition, note that $\rho_{D_\alpha}(y|x)$ depends on both $p_{\theta_t}(y|x)$ and $p_\theta(y|x)$, while $\rho_{\text{FixMatch}}(y|x)$ only depends on $p_{\theta_t}(y|x)$.

Remark Alpha divergence provides a general framework for distribution matching. It's more flexible compared to other divergences, e.g., Jensen–Shannon divergence. By choosing different values of α, our alpha divergence generalizes a number of well-known SSL approaches, including UDA ($\alpha = 1$) and FixMatch ($\alpha \to \infty$). Furthermore,

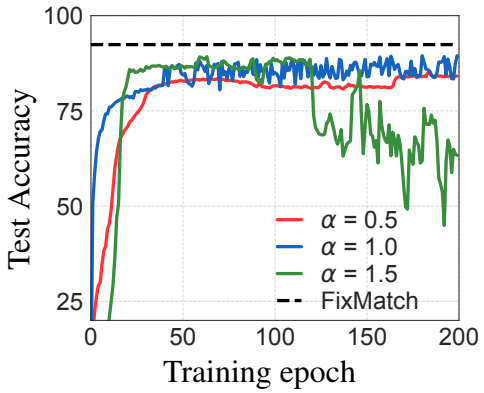


Figure 2. The learning curve of update Eqn (2) with alpha-divergence with different α on SVHN. The x-axis denotes the training epoch y-axis is the test accuracy. The labelled data includes 4 randomly picked mages per class. The dashed line shows the result of FixMatch. We follow all training settings in FixMatch.

note that alpha divergence is proportional to the α-th moment of density ratio $p(\cdot|x)/p(\cdot|x')$ (x clean image and x' augmented image). Therefore, when α is large and positive, large $p(\cdot|x)/p(\cdot|x')$ ratio is strongly penalized, preventing the case of $p(\cdot|x') \ll p(\cdot|x)$. This offers a natural and flexible way to propagate high confidence label on x to low-confidence examples x'.

3.2. An EM-like Optimization Framework for Label Matching

Despite the attractive property of alpha-divergence, we observe that directly incorporating alpha-divergence into the iterative update in (2) like UDA and FixMatch tends to cause instability in convergence when α is large, which is the regime of main interest. We illustrate this with an experiment on SVHN shown in Figure 2, which shows that using large α (e.g., $\alpha = 1.5$) can potentially obtain better results than smaller α but is much more unstable during training and may eventually diverge to worse results (note that the case of $\alpha = 1$ (blue curve) is UDA). This is because the iterative update in (1) does not correspond to optimizing a well-defined objective and does not guarantee to converge theoretically.

To address this problem, we provide an optimization-based framework for consistency regularization, which yields an EM-like algorithm when solved with a coordinate descent procedure. Our method enjoys better convergence and allows us to achieve better performance than directly combining alpha-divergence with (2). Our loss function is

$$\min_{\theta} \left\{ \mathcal{L}(\mathcal{D}_s;\, \theta) + \lambda \mathbb{E}_{x\sim\mathcal{D}_u} \left[\min_{\gamma(\cdot \mid x)} \Psi_{\alpha,\beta}(\theta, \gamma, x) \right] \right\}, \quad (8)$$

where $\Psi_{\alpha,\beta}(\theta, x)$ is a consistency regularization on x that we define as

$$\begin{aligned} &\Psi_{\alpha,\beta}(\theta, \gamma, x) = (1-\beta) D_\alpha(\gamma(\cdot \mid x) \,||\, p_\theta(\cdot \mid x)) \\ &+ \beta \mathbb{E}_{x'\sim\mathcal{P}_x} \left[D_\alpha(\gamma(\cdot \mid x) \,||\, p_\theta(\cdot \mid x')) \right] \\ &= \mathbb{E}_{x'\sim\mathcal{P}_x^\beta} \left[D_\alpha(\gamma(\cdot \mid x) \,||\, p_\theta(\cdot \mid x')) \right], \end{aligned} \quad (9)$$

where $\mathcal{P}_x^\beta(x') \stackrel{\text{def}}{=} (1-\beta)\delta(x' = x) + \beta \mathcal{P}_x(x')$ is a mixture of the random perturbation $\mathcal{P}_x$ and the original data x and $\beta \in [0, 1]$ is the ratio between the augmented and original data in $\mathcal{P}_x^\beta$. Here $\gamma(\cdot \mid x)$ is an auxiliary variable optimized in the space of all distributions. It is introduced to serves as a bridge for comparing $p_\theta(\cdot|x)$ and $p_\theta(\cdot|x')$, without having θ appearing on sides of the divergence like $\Phi(\theta; \theta, \mathcal{D})$ in (4). When $\beta = 0.5$, the regularization in (8) is a symmetrized version of alpha-divergence that generalizes Jensen-Shannon divergence.

We optimize our objective function in (8) by altere-naively optimizing θ and γ:

Updating γ With $\theta = \theta_t$ fixed, we update $\gamma(\cdot|x)$ for each x:

$$\gamma_t(\cdot \mid x) \leftarrow \arg\min_{\gamma(\cdot|x)} \mathbb{E}_{x'\sim\mathcal{P}_x^\beta} \Big[D_\alpha(\gamma(\cdot \mid x) \;\|\; p_{\theta_t}(\cdot \mid x')) \Big]. \quad (10)$$

Updating θ With $\gamma = \gamma_t$ fixed, we update θ by performing gradient descent on

$$\theta_{t+1} \leftarrow \arg\min_{\theta} L(\mathcal{D}_s;\, \theta) + \lambda \mathbb{E}_{x\sim\mathcal{D}_u} \big[\Psi_{\alpha,\beta}(\theta, \gamma, x) \big]. \quad (11)$$

Critically, for alpha-divergence, the optimal γ_t in Eqn (10) equals a simple powered expectation of $p_{\theta_t}(\cdot|x')$ as $x' \sim \mathcal{P}_x^\beta$,

$$\begin{aligned} &\gamma_t(\cdot \mid x) \propto \left(\mathbb{E}_{x'\sim\mathcal{P}_x^\beta} \left[p_{\theta_t}(\cdot \mid x')^{1-\alpha} \right] \right)^{\frac{1}{1-\alpha}} = \\ &\left((1-\beta) p_{\theta_t}(\cdot \mid x)^{1-\alpha} + \beta \mathbb{E}_{x'\sim\mathcal{P}_x} \left[p_{\theta_t}(\cdot \mid x')^{1-\alpha} \right] \right)^{\frac{1}{1-\alpha}}. \end{aligned} \quad (12)$$

Proof. Eqn (10) could be viewed as a general optimization of form:

$$\min_{\gamma} \sum_i w_i [D_\alpha(\gamma \,||\, p_i)],$$

where w_i denotes the importance score for each distribution

p_i. With simple calculations, we have,

$$\sum_i w_i[D_\alpha(\gamma \,||\, p_i)] = \sum_i w_i \sum_x \frac{\gamma^\alpha(x)}{p_i^{\alpha-1}(x)}$$
$$= \sum_x \sum_i w_i p_i^{1-\alpha}(x)\gamma^\alpha(x),$$
$$= \sum_x \bar{p}^{1-\alpha}(x)\gamma^\alpha(x), \text{with } \bar{p}(x) = \left(\sum_i w_i p_i^{1-\alpha}(x)\right)^{\frac{1}{1-\alpha}}$$
$$= D_\alpha(\gamma||z\bar{p}),$$

where z is the normalization constant. Therefore, we have $\gamma^* \propto \bar{p}$. □

This formulation reduces to the typical averaging when $\alpha \to 0$, and the reduces to geometric mean when $\alpha \to 1$. On the other hand, when $\alpha \geq 1$, the regime of our interest, $\gamma_t(\cdot|x)$ becomes a (powered) harmonic mean of $p_{\theta_t}(\cdot|x')$ when $x' \sim \mathcal{P}_x^\beta$. Consider the limit $\alpha \to +\infty$, in which case we have $\gamma_t(y|x) \approx \operatorname{ess\,inf}_{x' \sim \mathcal{P}_x^\beta} p_{\theta_t}(y \mid x')$, which suggests that $\gamma_t(y|x)$ is large only when $p_{\theta_t}(y \mid x')$ are large for both the original data $x' = x$ and all the augmented data $x' \sim \mathcal{P}_x$. This is again consistent with the idea of emphasizing high confidence instances.

Our method Combining these two updates yields a simple and practical algorithm shown in Algorithm 1, in which we apply one-step of mini-batch gradient descent to update θ at each iteration and approximate $\mathbb{E}_{x' \sim \mathcal{P}_x}[\cdot]$ in (9) and (12) by drawing a number of n random samples $\{x_i'\}_{i=1}^n$ from $\mathcal{P}_x$ for practical efficiency.

Our method converges theoretically because it is a coordinate descent by design. It is similar in style to expectation maximization (EM), especially the α-EM [e.g., 25] which uses alpha-divergence in EM. However, our method is designed from an optimization perspective for enforcing the label consistency on augmented data, rather than a generative modeling perspective underlying EM. Empirically, we approximate $\mathbb{E}_{x' \sim \mathcal{P}_x}[\cdot]$ (see Eqn. 9) by drawing a number of n random samples $\{x_i'\}_{i=1}^n$ from $\mathcal{P}_x$. See Algorithm 1 for details.

Time Cost As shown in Algorithm 1, our proposed algorithm introduces an additional latent variable γ which has a closed-form solution. Therefore, compared to KL divergence, we introduce almost no additional time cost.

4. Experiments

We test AlphaMatch on a variety of standard SSL benchmarks (e.g. STL-10, CIFAR-10, CIFAR-100 and SVHN) and compare it with a number of state-of-the-art (SOTA) SSL baselines, including MixMatch [2], ReMixMatch [3] and FixMatch [32]. We show that AlphaMatch achieves the best performance in all benchmark settings evaluated.

We use the code-base provided in FixMatch [32][1] for implementation. Throughout our experiments, we simply set $\alpha = 1.5$ and $\beta = 0.5$ without tuning; we found the default setting yields the best performance in almost all the cases. We provide comprehensive ablation studies on the impact of different choices of α and β in section 4.4.

4.1. STL-10

We conduct experiments on the challenging SLT-10 dataset. SLT-10 is a realistic and challenging SSL dataset which contains 5,000 labeled images, and 100,000 unlabeled, which are extracted from a similar but broader distribution of images than labelled data. The unlabelled data contains other types of animals (bears, rabbits, etc.) and vehicles (trains, buses, etc.) in addition to the ones in the labeled set. The distribution shift between the labeled and unlabeled data casts a higher challenge for SSL algorithms, and requires us to learn models with stronger generalizability. AlphaMatch again shows clear advantages over existing methods in this case.

Settings We preprocess the data and split the labeled images into 5 folds with the same data partition as FixMatch [32], with each partition containing 1,000 labels. Specifically, for each dataset, we use 4,000 labelled data and 100,000 unlabelled data to train and use the remained 1,000 labelled data to test. Thus, a total of 5 models will be trained and evaluated. The final performance is averaged over these 5 individual runs.

We use the Wide ResNet(WRN)-28-2 and WRN-16-8 model for our method and all the baselines. We compare AlphaMatch with π-model, unsupervised data augmentation (UDA), MixMatch, ReMixMatch and FixMatch. All baseline results are produced by exactly following the same training setting suggested in [32]. For our method, we use the default setting of $\alpha = 1.5$, $\beta = 0.5$ and $n = 1$.

Results We report the averaged accuracy of all 5 runs in Table 1. AlphaMatch significantly outperforms all other baselines in this setting. In particular, for WRN-28-2, compared with FixMatch, we improve the accuracy from 89.28% to 90.36%. Our performance is also about 1.9% higher than ReMixMatch. For WRN-16-8, we also improve the baselines with a large margin.

4.2. CIFAR-10, SVHN and CIFAR-100

We then test AlphaMatch on three widely-used benchmark SSL datasets: CIFAR-10 [21], SVHN [27] and CIFAR-100. With only 4 labeled data per class, we achieve

[1] https://github.com/google-research/fixmatch

Algorithm 1 AlphaMatch: Improving Consistency for SSL with Alpha-divergence

Input: labeled data $\mathcal{D}_s$; unlabeled data $\mathcal{D}_u$; regularization coefficient λ; alpha-divergence hyper-parameters α and β; number of augmentation samples n; initial model parameter θ_0.
for iteration t **do**
  Randomly sample a labeled batch $\mathcal{B}_s$ from $\mathcal{D}_s$ and an unlabeled batch $\mathcal{B}_u$ from $\mathcal{D}_u$.
  for each x in $\mathcal{B}_u$ **do**
    Apply data augmentation on x for n times, yielding augmented examples $\{x'_i\}_{i=1}^n$
    Fixing $\theta = \theta_t$, update $\gamma(\cdot|x)$ with

$$\gamma_{t+1}(\cdot \mid x) \leftarrow \left((1-\beta) \times p_{\theta_t}(\cdot \mid x)^{1-\alpha} + \frac{\beta}{n} \times \sum_{i=1}^{n} \left[p_{\theta_t}(\cdot \mid x'_i)^{1-\alpha}\right]\right)^{\frac{1}{1-\alpha}}.$$

    Approximate $\Psi_{\alpha,\beta}(\theta, \gamma_{t+1}, x)$ in (9) with augmented examples $\{x'_i\}_{i=1}^n$ accordingly.
  end for
  Update θ_{t+1} by applying one step of gradient descent on (11) over batch $\mathcal{B}_u$:

$$\theta_{t+1} \leftarrow \theta_t - \epsilon \nabla_\theta \Big(\mathcal{L}(\mathcal{B}_s;\ \theta) + \lambda \mathbb{E}_{x \sim \mathcal{B}_u}[\Psi_{\alpha,\beta}(\theta, \gamma_{t+1}, x)]\Big)\Big|_{\theta=\theta_t}.$$

end for

Method	WRN-28-2 (%)	WRN-16-8 (%)
π-model	71.39±1.21	74.92±1.18
UDA	86.57±1.06	89.14±0.73
MixMatch	85.19±1.24	87.52±0.79
ReMixMatch	88.42±0.78	91.07±0.84
FixMatch	89.28±0.63	91.35±0.67
AlphaMatch	**90.36±0.75**	**92.83±0.86**

Table 1. Testing accuracy on STL-10. All averaged over 5 different folds. Results on two different model architectures, WRN-28-2 and WRN-16-8 is reported.

91.28%±3.41% accurcy on CIFAR-10, 97.03%±0.26% on SVHN, and 61.27%±3.13% on CIFAR-100, all of which yield significantly improvement over prior SSL results on this task.

Settings For a fair comparison, we proceed with minimum changes to the code provided by [32]. Specifically, we use the same setting and random seeds as in FixMatch [32] to generate labeled and unlabeled data partitions. We use wide ResNet-28-2 [40] with 1.5M parameters as our prediction model, and then train it using SGD with cosine learning rate decay for 1024 epochs. We use CTAugment [3] for data augmentation as suggested in FixMatch [32].

We evaluate two SSL settings, which use 4 (resp. 25) labeled images per class, yielding 40 (resp. 250) labeled images in CIFAR-10 and SVHN and 400 (resp. 2500) in CIFAR-100. The remaining data in the training set is regarded as unlabelled data. For our method, we set $\alpha = 1.5$ and $\beta = 0.5$ by default and investigate the effect of α and β in section 4.4. In order to maintain the similar training computation cost as FixMatch, We set the number of augmentation applied per image $n = 1$ as default; the effect of n is studied in section 4.4. Following FixMatch, we combine AlphaMatch with an additional distribution alignment loss on CIFAR-100, which is proposed in ReMixMatch [3].

Fully Supervised Baselines As a reference, we train fully-supervised baselines, i.e., training the models with all the training labels available. The test accuracy is 96.14%±0.03%, 97.89%±0.02% and 79.17%±0.01% on CIFAR-10, SVHN and CIFAR-100, respectively.

Results We report the test accuracy of all baselines [2, 3, 24, 32, 38] along with AlphaMatch in Table 2. All results are averaged over 5 random trials with different data partitions. We can see that the models trained with AlphaMatch achieve the best performance in all the settings. The gain is especially significant when the number of labeled data examples is limited (e.g. 4 labeled image per class). In particular, compared with FixMatch, our method achieves a 2.64% improvement on CIFAR-10 and 4.58% on SVHN for test accuracy. On CIFAR-100, when combined with the distribution alignment loss as suggested in [3], AlphaMatch yields the best performance, achieving $61.26\% \pm 3.13\%$ accuracy with 4 labels per class and and $74.98\% \pm 0.27\%$ with 25 labels per class.

4.3. Point Cloud Classification

To further verify the effectiveness of our method, we conduct experiments for semi-supervised 3D point cloud

Method	CIFAR-10		SVHN		CIFAR-100	
	40 labels	250 labels	40 labels	250 labels	400 labels	2500 labels
Pseudo-Labeling	-	50.22±0.43	-	79.79±1.09	-	42.62±0.46
UDA	70.95±5.93	91.18±1.08	47.37±20.51	94.31±2.70	40.72±0.88	66.87±0.22
MixMatch	52.46±11.50	88.95±0.86	57.45±14.53	96.02±0.23	33.39±1.32	60.06±0.37
ReMixMatch	80.90±9.64*	94.56±0.05*	96.64±0.30*	97.08±0.48*	55.72±2.06*	73.57±0.31*
FixMatch	88.71±3.35	94.93±0.33	92.35±7.65	97.36±0.64	59.79±2.94*	74.63±0.22*
AlphaMatch	**91.35±3.38**	**95.03±0.29**	**97.03±0.26**	**97.56±0.32**	**61.26±3.13***	**74.98±0.27***

Table 2. Testing accuracy (%) of different methods on CIFAR-10, SVHN, and CIFAR-100. All averaged over 5 different folds. The '*' indicates the results are achieved by combining the distribution alignment loss proposed in ReMixMatch [3].

classification. We test a number of prior art SSL baselines and our algorithm on ModelNet40 [6] using the SOTA Dynamic graph convolution neural network (DGCNN) [36].

Dataset ModelNet40 is the most widely adopted benchmark for point-cloud classification. It contains objects from 40 common categories. There are 9840 objects in the training set and 2468 in the test set. In the experiment, we randomly select 100 objects for each category as labelled data and treat the other objects in the training set as unlabelled data. It means, we use 4,000 labelled data and 5,420 unlabelled data during training and then evaluate the performance on the original test set.

Settings For all the baselines and our method, we use Gaussian blur $\mathcal{N}(0, 0.02)$ as weak augmentation, and using additional randomized jittering as strong augmentation. For the DGCNN model, we use 2048 number of particles for each object and set the number of neighbours to 20. We train the model with 2,00 epochs with SGD with cosing learning rate decay and 0.9 momentum.

Result We report the averaged accuracy of all 5 runs in Table 3. AlphaMatch significantly outperforms all other baselines in this setting. In particular, compared with FixMatch, we improve the accuracy from 86.5% to 88.3%. Compared to other baselines, the proposed methods can also boost the performance by a large margin. The result shows that our method can also be applied to other settings except image classification.

4.4. Ablation Studies

Impact of α and β We follow all training settings as Section 4.2. All models are trained with 4 labeled examples per class. On this dataset, FixMatch achieves 92.4% test accuracy (see the dashed black line). In Figure 3 (a), we fix $\alpha = 1.5$ and study the impact of β. We find a smaller β (e.g. $\beta = 0.2$ or 0.5) often yields more stabilized training; in the contrary, a larger β (e.g. $\beta = 0.8$) focuses less on clean (or weakly augmented) data and performs worse than small β in general.

Method	ModelNet40 Accuracy (%)
π-model	81.82±1.18
UDA	86.15±1.13
MixMatch	85.34±1.05
ReMixMatch	85.66±0.92
FixMatch	86.47±0.79
AlphaMatch	**88.32±0.84**

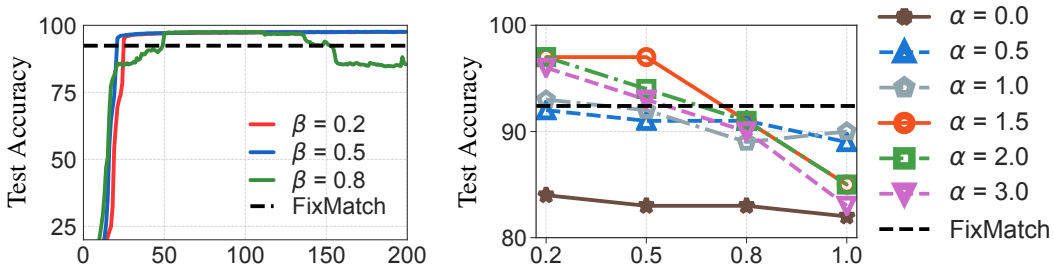
Table 3. Testing accuracy on ModelNet40. All averaged over 5 different folds.

We plot in Figure 3 (b) the testing accuracy of AlphaMatch with different α and β on SVHN dataset. As we can see from Figure 3, a larger α (e.g. $\alpha \geq 1.5$) and a smaller β (e.g. $\beta \leq 0.5$) normally achieves better performance than FixMatch. This is expected, as using alpha-divergence with large α values help propagate high confidence labels and a smaller β helps to stabilize the training. On the other hand, if α is too large, the performance may diminish because numerical instability increases.

We observe that $\alpha = 1.5$ and $\beta = 0.5$ yields the best performance, achieving a good balance between consistency regularization and training stability.

Impact of the number of augmented examples n In our algorithm, n controls on how many augmented examples is generated to approximate $\mathcal{P}_x$ (see Algorithm 1). We can expect that $\{x'_i\}_{i=1}^n$ forms increasingly better approximation for $\mathcal{P}_x$ with a larger n. In this section, we perform an in-depth analysis on the effect of different n values. Intuitively, a smaller n leads to better energy-efficiency while a larger n is more computationally expensive but may yield more robust estimation hence produce better performance.

We test AlphaMatch on CIFAR-10 and SVHN, with the same settings as section 4.2. Table 4 shows the performance of various n. All results are averaged over 5 random trials. We find $n = 1$ often performs competitively and a larger



(a) Training epoch ($\alpha = 1.5$) (b) choices of β

Figure 3. (b) The Learning curve of AlphaMatch on SVHN, when using $\alpha = 1.5$ and varying β. The x-axis denotes the training epoch and y-axis represents the test accuracy. The dashed line shows the result of FixMatch. (b) Testing accuracy of AlphaMatch with various α and β on the SVHN dataset. All models are trained with 4 labeled examples per class.

n augmented examples	1	2	4	10
CIFAR-10 Accuracy (%)	91.35±3.38	91.48±3.41	91.63±3.12	91.59±3.08
SVHN Accuracy (%)	97.03±0.26	97.07±0.19	97.18±0.23	97.18±0.21

Table 4. Comparison on testing accuracy for various n. All results are averaged over 5 random trials with the same settings as section 4.2.

n (e.g. $n = 4$, $n = 10$) yields slightly improvements in general. However, a large n can significantly increase the computation cost since it increases the forward and backward time cost of the training model, and more critically, it requires a larger GPU memory cost. Considering its huge time/memory cost and mild improvement, we recommend to use $n = 1$ in practice.

5. Related Works

Semi-supervised learning (SSL) has been a classical subfield of machine learning with a large literature. Examples of classical methods include transductive models [e.g. 10, 13, 19], co-training [e.g. 5, 28], entropy minimization [e.g. 14, 23], graph-based models [e.g. 1, 4, 16, 35, 37, 43, 44] and many more (see [e.g. 7, 45]).

Due to the recent success of deep learning, combining generative model [9, 17] and adversarial training [26] with SSL achieves meaningful improvement over the classic SSL methods in many real-world problems, e.g. image classification [34, 38], segmentation [29], detection [18]. Most recently, data augmentation has been introduced into SSL and achieved great success. Π-model, for example, shares the similar idea of (1), but replaces the KL divergence with L2 distance and replace x in (1) with another random copy of augmentation x''. MixMatch [2] and ReMixMatch [3], use *mixup* [42] to do data augmentation. Based on MixMatch, ReMixMatch uses some additional loss and new data augmentation method to improve the performance. These methods force the label consistency between weakly augmented examples (or examples without augmentation) and strong augmented examples, and improve the state-of-the-art results on classification tasks by a large margin.

Closely related to our work, a series of recent SSL methods have been proposed based on the general idea of enforcing the label prediction of an image to be consistent with its augmented counterparts. The label consistency has been mostly measured by either KL divergence [e.g., 2, 3, 26, 32, 38], or mean squared error [e.g., 22, 30, 31, 33]. Almost all these methods use iterative regularization processes similar to (1)-(2), with the stop-gradient trick (see Section 2). Compared to these works, our method first introduces alpha-divergence as consistency measure, and equip it with a convergent EM-like matching algorithm to achieve better results.

Most recently, Chen & He [8] presents a similar EM-like approach for self-supervised learning. For each step, Chen & He [8] first closed-form optimizes a local latent variable (similar to our γ) and then updates the model parameters with a fixed γ. This motivates us to further explore our method to more general topics.

6. Conclusion

In this paper, we propose to use alpha-divergence and a new optimization-based framework to build a semi-supervised learning algorithm based on data augmentation. The proposed AlphaMatch is simple yet powerful. With only a few lines of extra code to implement alpha-divergence and the EM-like update, it achieves the state-of-the-art performance on various benchmarks.

For future work, we will apply our algorithm to more practical tasks, e.g. 2D/3D segmentation, machine translation, object detection. By further extending and improving our framework, it is promising to push the frontier of ML with limited data to close the gap between few-shot learn-

ing and SSL. Finally, we plan to extend our algorithm to unsupervised learning, out-of-distribution detection and other related topics.

References

[1] Belkin, Mikhail, Matveeva, Irina, and Niyogi, Partha. Regularization and semi-supervised learning on large graphs. In *International Conference on Computational Learning Theory*, pp. 624–638. Springer, 2004. 8

[2] Berthelot, David, Carlini, Nicholas, Goodfellow, Ian, Papernot, Nicolas, Oliver, Avital, and Raffel, Colin A. Mixmatch: A holistic approach to semi-supervised learning. In *NeurIPS*, pp. 5050–5060, 2019. 1, 5, 6, 8

[3] Berthelot, David, Carlini, Nicholas, Cubuk, Ekin D, Kurakin, Alex, Sohn, Kihyuk, Zhang, Han, and Raffel, Colin. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. In *ICLR*, 2020. 1, 5, 6, 7, 8

[4] Blum, Avrim and Chawla, Shuchi. Learning from labeled and unlabeled data using graph mincuts. 2001. 8

[5] Blum, Avrim and Mitchell, Tom. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pp. 92–100, 1998. 8

[6] Chang, Angel X, Funkhouser, Thomas, Guibas, Leonidas, Hanrahan, Pat, Huang, Qixing, Li, Zimo, Savarese, Silvio, Savva, Manolis, Song, Shuran, Su, Hao, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 7

[7] Chapelle, Olivier, Scholkopf, Bernhard, and Zien, Alexander. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009. 1, 8

[8] Chen, Xinlei and He, Kaiming. Exploring simple siamese representation learning, 2020. 8

[9] Dai, Zihang, Yang, Zhilin, Yang, Fan, Cohen, William W, and Salakhutdinov, Russ R. Good semi-supervised learning that requires a bad gan. In *Advances in neural information processing systems*, pp. 6510–6520, 2017. 8

[10] Demiriz, Ayhan, Bennett, Kristin P, and Embrechts, Mark J. Semi-supervised clustering using genetic algorithms. *Artificial neural networks in engineering (ANNIE-99)*, pp. 809–814, 1999. 8

[11] Deng, Jia, Dong, Wei, Socher, Richard, Li, Li-Jia, Li, Kai, and Fei-Fei, Li. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009. 3

[12] DeVries, Terrance and Taylor, Graham W. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 2

[13] Gammerman, Alex, Vovk, Volodya, and Vapnik, Vladimir. Learning by transduction. *arXiv preprint arXiv:1301.7375*, 2013. 8

[14] Grandvalet, Yves and Bengio, Yoshua. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pp. 529–536, 2005. 8

[15] Grandvalet, Yves and Bengio, Yoshua. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pp. 529–536, 2005. 2

[16] He, Yihui, Lin, Ji, Liu, Zhijian, Wang, Hanrui, Li, Li-Jia, and Han, Song. Amc: Automl for model compression and acceleration on mobile devices. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 784–800, 2018. 8

[17] Higgins, Irina, Matthey, Loic, Pal, Arka, Burgess, Christopher, Glorot, Xavier, Botvinick, Matthew, Mohamed, Shakir, and Lerchner, Alexander. beta-vae: Learning basic visual concepts with a constrained variational framework. 2016. 8

[18] Jeong, Jisoo, Lee, Seungeui, Kim, Jeesoo, and Kwak, Nojun. Consistency-based semi-supervised learning for object detection. In *Advances in neural information processing systems*, pp. 10759–10768, 2019. 8

[19] Joachims, Thorsten. Transductive inference for text classification using support vector machines. In *Icml*, volume 99, pp. 200–209, 1999. 8

[20] Kahn, Jacob, Lee, Ann, and Hannun, Awni. Self-training for end-to-end speech recognition. In *ICASSP*, pp. 7084–7088. IEEE, 2020. 1

[21] Krizhevsky, Alex, Hinton, Geoffrey, et al. Learning multiple layers of features from tiny images. 2009. 5

[22] Laine, Samuli and Aila, Timo. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016. 8

[23] Lee, Chi-Hoon, Wang, Shaojun, Jiao, Feng, Schuurmans, Dale, and Greiner, Russell. Learning to model spatial dependency: Semi-supervised discriminative random fields. In *Advances in Neural Information Processing Systems*, pp. 793–800, 2007. 8

[24] Lee, Dong-Hyun. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, pp. 2, 2013. 6

[25] Matsuyama, Y. The α-em algorithm: surrogate likelihood maximization using α-logarithmic information measures. *IEEE Transactions on Information Theory*, 49(3):692–706, 2003. 5

[26] Miyato, Takeru, Maeda, Shin-ichi, Koyama, Masanori, Nakae, Ken, and Ishii, Shin. Distributional smoothing with virtual adversarial training. In *ICLR*, 2016. 2, 8

[27] Netzer, Yuval, Wang, Tao, Coates, Adam, Bissacco, Alessandro, Wu, Bo, and Ng, Andrew Y. Reading digits in natural images with unsupervised feature learning. 2011. 5

[28] Nigam, Kamal and Ghani, Rayid. Analyzing the effectiveness and applicability of co-training. In *Proceedings of the ninth international conference on Information and knowledge management*, pp. 86–93, 2000. 8

[29] Papandreou, George, Chen, Liang-Chieh, Murphy, Kevin P, and Yuille, Alan L. Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *Proceedings of the IEEE international conference on computer vision*, pp. 1742–1750, 2015. 8

[30] Rasmus, Antti, Berglund, Mathias, Honkala, Mikko, Valpola, Harri, and Raiko, Tapani. Semi-supervised learning with ladder networks. In *NeurIPS*, pp. 3546–3554, 2015. 1, 8

[31] Sajjadi, Mehdi, Javanmardi, Mehran, and Tasdizen, Tolga. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *Advances in neural information processing systems*, pp. 1163–1171, 2016. 8

[32] Sohn, Kihyuk, Berthelot, David, Li, Chun-Liang, Zhang, Zizhao, Carlini, Nicholas, Cubuk, Ekin D, Kurakin, Alex, Zhang, Han, and Raffel, Colin. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint arXiv:2001.07685*, 2020. 1, 2, 5, 6, 8

[33] Tarvainen, Antti and Valpola, Harri. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in neural information processing systems*, pp. 1195–1204, 2017. 8

[34] Wang, Dilin, Li, Meng, Gong, Chengyue, and Chandra, Vikas. Attentivenas: Improving neural architecture search via attentive sampling. *arXiv preprint arXiv:2011.09011*, 2020. 8

[35] Wang, Fei and Zhang, Changshui. Label propagation through linear neighborhoods. *IEEE Transactions on Knowledge and Data Engineering*, 20(1):55–67, 2007. 8

[36] Wang, Yue, Sun, Yongbin, Liu, Ziwei, Sarma, Sanjay E, Bronstein, Michael M, and Solomon, Justin M. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. 7

[37] Wu, Mingrui and Schölkopf, Bernhard. Transductive classification via local learning regularization. In *Artificial Intelligence and Statistics*, pp. 628–635, 2007. 8

[38] Xie, Qizhe, Dai, Zihang, Hovy, Eduard, Luong, Minh-Thang, and Le, Quoc V. Unsupervised data augmentation. *arXiv preprint arXiv:1904.12848*, 2019. 1, 2, 6, 8

[39] Xie, Qizhe, Hovy, Eduard, Luong, Minh-Thang, and Le, Quoc V. Self-training with noisy student improves imagenet classification. In *CVPR*, 2020. 1

[40] Zagoruyko, Sergey and Komodakis, Nikos. Wide residual networks. In *BMVC*, 2016. 6

[41] Zhai, Runtian, Cai, Tianle, He, Di, Dan, Chen, He, Kun, Hopcroft, John, and Wang, Liwei. Adversarially robust generalization just requires more unlabeled data. *arXiv preprint arXiv:1906.00555*, 2019. 1

[42] Zhang, Hongyi, Cisse, Moustapha, Dauphin, Yann N, and Lopez-Paz, David. mixup: Beyond empirical risk minimization. In *ICLR*, 2018. 8

[43] Zhou, Dengyong, Bousquet, Olivier, Lal, Thomas N, Weston, Jason, and Schölkopf, Bernhard. Learning with local and global consistency. In *Advances in neural information processing systems*, pp. 321–328, 2004. 8

[44] Zhu, Xiaojin and Ghahramani, Zoubin. Learning from labeled and unlabeled data with label propagation. 2002. 8

[45] Zhu, Xiaojin Jerry. Semi-supervised learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2005. 8