Semi-Supervised Learning with Variational Bayesian Inference and Maximum Uncertainty Regularization

Kien Do, Truyen Tran, Svetha Venkatesh
Applied Artificial Intelligence Institute (A2I2), Deakin University, Geelong, Australia
{k.do, truyen.tran, svetha.venkatesh}@deakin.edu.au

Abstract

We propose *two* generic methods for improving semi-supervised learning (SSL). The first integrates *weight perturbation* (WP) into existing "consistency regularization" (CR) based methods. We implement WP by leveraging variational Bayesian inference (VBI). The second method proposes a novel consistency loss called "maximum uncertainty regularization" (MUR). While most consistency losses act on perturbations in the vicinity of each data point, MUR actively searches for *"virtual"* points situated beyond this region that cause the most uncertain class predictions. This allows MUR to impose smoothness on a wider area in the input-output manifold. Our experiments show clear improvements in classification errors of various CR based methods when they are combined with VBI or MUR or both.

Introduction

Recent success in training deep neural networks is mainly attributed to the availability of large, labeled datasets. However, annotating large amounts of data is often expensive and time-consuming, and sometimes requires specialized expertise (e.g., healthcare). Under these circumstances, semi-supervised learning (SSL) has proven to be an effective means of mitigating the need for labels by leveraging unlabeled data to considerably improve performance. Among a wide range of approaches to SSL (van Engelen and Hoos 2019), *"consistency regularization"* (CR) based methods are currently state-of-the-art (Bachman, Alsharif, and Precup 2014; Sajjadi, Javanmardi, and Tasdizen 2016; Laine and Aila 2016; Tarvainen and Valpola 2017; Miyato et al. 2018; Verma et al. 2019; Xie et al. 2019; Berthelot et al. 2019b; Sohn et al. 2020). These methods encourage neighbor samples to share labels by enforcing consistent predictions for inputs under perturbations.

Although the perturbations can be created in either the input/feature space (data perturbation) or the weight space (weight perturbation), existing CR based methods focus exclusively on the former and leave the latter underexplored. Despite being related, *weight perturbation* (WP) is inherently different from data perturbation (DP) in the sense that WP directly reflects different views of the classifier f on the original data distribution (e.g., $f_w(x)$ v.s. $f_{w'}(x)$) while DP indirectly causes the classifier to adjust its view to adapt to different data distributions (e.g., $f_w(x)$ v.s. $f_w(x')$). Therefore, we hypothesize that the two types of perturbations are complementary and could be combined to increase the classifier's robustness. To implement WP, we treat the classifier's weights w as random variables and perform variational Bayesian inference (VBI) on w. This approach has several advantages. First, perturbations of w can be generated easily via drawing samples from an explicit variational distribution $q_\phi(w|\mathcal{D})$. We also take advantage of the local reparameterization trick and variational dropout (VD) (Kingma, Salimans, and Welling 2015; Molchanov, Ashukha, and Vetrov 2017) to substantially reduce the sampling cost and variance of the gradients w.r.t. the weight samples, making VBI scalable for deep neural networks. Second, since VBI is a realization of the Minimum Description Length (MDL) framework (Hinton and von Cramp 1993; Honkela and Valpola 2004), a classifier trained under VBI, in principle, often generalize better than those trained in the standard way.

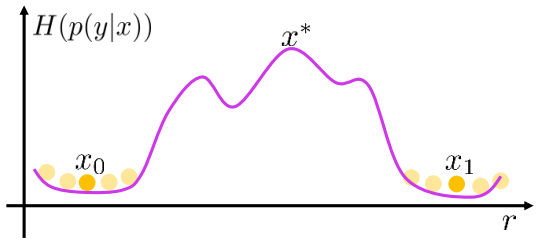


Figure 1: The most uncertain *"virtual"* point x^* usually lies outside the vicinity of x_0 and is the most disruptive.

Standard DP methods (e.g., Gaussian noise, dropout) often generate perturbations in the vicinity of each data point and ignore those in the vacancy among data points, which means consistency losses equipped with standard DPs can only train *locally smooth* classifiers that do not generalize well in general. To overcome this limitation, we propose a novel consistency loss called *"maximum uncertainty regularization"* (MUR) of which the key component is finding for each real data point x_0 a neighbor *"virtual"* point x^* that has the most uncertain class prediction given by the classifier f. During training, f becomes increasingly confident about

its predictions of the real data points and their neighborhood (otherwise, the training cannot converge). Thus, by choosing x^* with the above properties, we can guarantee, with high probability, that i) x^* is situated *outside the vicinity* of x_0, and ii) x^* causes the biggest disruption to the classifier's predictions (Fig. 1). This observation suggests that MUR enforces smoothness on a wider and rougher area in the input space than conventional consistency losses, hence, making the classifier generalize better. As MUR operates in the input space not the weight space, it is complementary to WP and in some cases, both can be used together.

In our experiments, we show that when strong data augmentation is not available, WP and MUR significantly boost the performance of existing CR based methods on various benchmark datasets.

Preliminaries

Consistency Regularization Based Methods for Semi-supervised Learning

We briefly present two representative CR based methods namely Π-model (Laine and Aila 2016) and Mean Teacher (Tarvainen and Valpola 2017). Other methods are discussed in the related work.

In Π-model, a classifier f has *deterministic* weights θ but contains random input/feature perturbation layers such as binary dropout (Srivastava et al. 2014) and/or additive Gaussian noise. Thus, if we pass an input sample x through f_θ twice, we will get two different distributions $p(y|x,\theta)$ and $p(y|x',\theta)$ where x' is a perturbation of x in the input/feature space. The loss of Π-model is given by:

$$\mathcal{L}_\Pi(\theta) = \mathbb{E}_{(x_l,y_l)\sim\mathcal{D}_l}\left[-\log p(y_l|x_l,\theta)\right] + \lambda(t)\mathbb{E}_{x\sim\mathcal{D}}\left[\frac{1}{K}\sum_{k=1}^{K}\left(p(k|x,\theta) - p(k|x',\theta_{\text{sg}})\right)^2\right] \quad (1)$$

$$= \mathcal{L}_{\text{xent},l}(\theta) + \lambda(t)\mathcal{L}_{\Pi,\text{cons}}(\theta,\theta_{\text{sg}}) \quad (2)$$

where $\mathcal{D}_l$, $\mathcal{D}_u$ denote the disjoint labeled and unlabeled training datasets; $\mathcal{D} = \mathcal{D}_l \cup \mathcal{D}_u$; K is the number of classes; $\lambda(t)$ is a "ramp" function which depends on the training step t; θ_{sg} denotes θ with no gradient update; $\mathcal{L}_{\text{xent}}(\theta)$ is the cross-entropy loss on labeled samples and $\mathcal{L}_{\Pi,\text{cons}}(\theta)$ is the consistency loss on both labeled and unlabeled samples.

Mean Teacher (MT), on the other hand, introduces another network $f_{\bar\theta}$ called *"mean teacher"* whose weights $\bar\theta$ are the exponential moving averages (EMA) of θ across training steps: $\bar\theta_t = \alpha\bar\theta_{t-1} + (1-\alpha)\theta$ ($\alpha \in [0,1]$ is a momentum). Based on $f_{\bar\theta}$, MT defines a new consistency loss $\mathcal{L}_{\text{MT,cons}}(\theta,\bar\theta)$ between the outputs of f_θ and $f_{\bar\theta}$, which leads to the final loss:

$$\mathcal{L}_{\text{MT}}(\theta) = \mathbb{E}_{(x_l,y_l)\sim\mathcal{D}_l}\left[-\log p(y_l|x_l,\theta)\right] + \lambda(t)\mathbb{E}_{x\sim\mathcal{D}}\left[\frac{1}{K}\sum_{k=1}^{K}\left(p(k|x,\theta) - p(k|x',\bar\theta)\right)^2\right] \quad (3)$$

$$= \mathcal{L}_{\text{xent},l}(\theta) + \lambda(t)\mathcal{L}_{\text{MT,cons}}(\theta,\bar\theta) \quad (4)$$

Variational Bayesian Inference and Variational Dropout

In Bayesian learning, we assume there is a prior distribution of weights, denoted by $p(w)$. After observing the training dataset $\mathcal{D}$, we update our belief about w as $p(w|\mathcal{D}) = \frac{p(w)p(\mathcal{D}|w)}{p(\mathcal{D})} = \frac{p(w)p(\mathcal{D}|w)}{\int_w p(w)p(\mathcal{D}|w)}$. Since $p(w|\mathcal{D})$ is generally intractable to compute, we approximate $p(w|\mathcal{D})$ with a variational distribution $q_\phi(w)$:

$$\begin{aligned}&\min_\phi D_{KL}\left(q_\phi(w)\|p(w|\mathcal{D})\right)\\ &= \min_\phi \mathbb{E}_{w\sim q_\phi(w)}\left[\log\frac{q_\phi(w)}{p(w)p(\mathcal{D}|w)}\right]\\ &= \min_\phi \mathbb{E}_{w\sim q_\phi(w)}\left[-\log p(\mathcal{D}|w)\right] + D_{KL}\left(q_\phi(w)\|p(w)\right)\end{aligned} \quad (5)$$

For discriminative classification tasks with an i.i.d. assumption of data, Eq. 5 is equivalent to:

$$\min_\phi \mathbb{E}_{w\sim q_\phi(w)}\left[\mathbb{E}_{(x,y)\sim\mathcal{D}}\left[-\log p(y|x,w)\right]\right] + \lambda D_{KL}\left(q_\phi(w)\|p(w)\right) \quad (6)$$

where $x \in \mathbb{R}^M$ and $y \in \{1,...,K\}$ denote an input sample and its corresponding label, respectively; λ is a balancing coefficient, usually set to $\frac{1}{|\mathcal{D}|}$.

The loss function in Eq. 6 consists of two terms: the expected negative data log-likelihood w.r.t. $q_\phi(w)$ and the KLD between $q_\phi(w)$ and $p(w)$ which usually acts as a regularization term on the model complexity. In order to minimize this loss, we need to find a model that yields good classification results yet still being as simple as possible. In fact, Eq. 6 can also be viewed as the *bits-back coding* objective under the scope of the Minimum Description Length (MDL) principle (Graves 2011; Hinton and von Cramp 1993; Honkela and Valpola 2004).

Generally, we could assume $q_\phi(w)$ to be a factorized Gaussian distribution $q_\phi(w) = \mathcal{N}(w;\theta,\sigma^2\text{I})$ and compute the gradient of the first term in Eq. 6 w.r.t. θ and σ using the reparameterization trick (Kingma and Welling 2013; Rezende, Mohamed, and Wierstra 2014). However, this approach (Blundell et al. 2015) is computationally inefficient and has high variance since it requires sampling of multiple weight components for every data point. To handle this problem, Kingma et. al. (Kingma, Salimans, and Welling 2015) proposed the *local reparameterization trick* and *variational dropout* (VD). Details about these techniques are given in the appendix.

Our Approach

We describe how to integrate weight perturbation (WP) and maximum uncertainty regularization (MUR) into CR based methods. Since WP is realized via variational Bayesian inference (VBI), we will use VBI in place of WP henceforth. We consider a general class of CR based methods whose original objectives are of the form $\mathcal{L}_{\text{xent},l}(\theta) + \lambda(t)\mathcal{L}_{\text{cons}}(\theta,\cdot)$, denoted as $\mathcal{M}$. Any method $M \in \mathcal{M}$ can be combined with VBI and MUR (denoted as M+VBI+MUR) by minimizing

the following loss:

$$\mathcal{L}_{M\text{+VBI+MUR}}(\phi) = \mathbb{E}_{w\sim q_\phi(w)}\left[\mathcal{L}_{\text{xent},l}(w)\right] + \lambda_1(t)\mathbb{E}_{w\sim q_\phi(w)}\left[\mathcal{L}_{M,\text{cons}}(w,\cdot)\right] + \lambda_2(t)D_{KL}\left(q_\phi(w)\|p(w)\right) + \lambda_3(t)\mathbb{E}_{w\sim q_\phi(w)}\left[\mathcal{L}_{\text{MUR}}(w)\right] \quad (7)$$

where $q_\phi(w) = \mathcal{N}\left(w;\theta,\sigma^2\mathrm{I}\right)$ is the variational distribution of the classifier's weights w; $\mathcal{L}_{\text{MUR}}$ is the MUR loss defined below; $\lambda_1(t)$, $\lambda_2(t)$, $\lambda_3(t)$ are different "ramp" functions.

Since Π-model and Mean Teacher are specific instances of $\mathcal{M}$, we can easily derive the losses of Π+VBI+MUR and MT+VBI+MUR from Eq. 7 by replacing $\mathcal{L}_{M,\text{cons}}(w,\cdot)$ with $\mathcal{L}_{\Pi,\text{cons}}(w,\theta_{\text{sg}})$ (Eq. 2) and $\mathcal{L}_{\text{MT,cons}}(w,\bar{\theta})$ (Eq. 4), respectively. For other CR based methods M' having additional loss terms $\mathcal{L}'(\theta)$ apart from $\mathcal{L}_{\text{xent},l}(\theta)$ and $\mathcal{L}_{\text{cons}}(\theta,\cdot)$, we can still construct the loss of M'+VBI+MUR by simply adding $\mathbb{E}_{q_\phi(w)}\left[\mathcal{L}'(w)\right]$ to the RHS of Eq. 7.

M+VBI+MUR has two special cases which are M+VBI and M+MUR. The loss of M+VBI ($\mathcal{L}_{M\text{+VBI}}$) is similar to $\mathcal{L}_{M\text{+VBI+MUR}}$ but with the last term on the RHS discarded (e.g., by setting $\lambda_3(t) = 0\ \forall t$). On the other hand, by removing the third term as well as the expectation w.r.t. $q_\phi(w)$ in all the remaining terms on the RHS of Eq. 7, we obtain the loss of M+MUR ($\mathcal{L}_{M\text{+MUR}}$).

It is important to note that the second and the last terms on the RHS of Eq. 7 are novel and have never been used for SSL. While the second term shares some similarity with ensemble learning in which different views of a classifier are combined to obtain a robust prediction for a particular training example, the last term is more related to multi-view learning (Qiao et al. 2018) as different classifiers are applied to different views of data. However, compared to ensemble learning and multi-view learning, our approach is much more efficient since we can have almost infinite numbers of views without training multiple classifiers.

Weight Perturbation via Variational Bayesian Inference

In Eq. 7, we perturb the classifier's weights w by drawing random samples from $q_\phi(w)$. Minimizing $\mathcal{L}_{\text{xent},l}(w)$, $\mathcal{L}_{M,\text{cons}}(w,\cdot)$, and $\mathcal{L}_{\text{MUR}}(w)$ makes the classifier robust against different weight perturbations while minimizing $D_{KL}\left(q_\phi(w)\|p(w)\right)$ prevents the classifier from being too complex. Both improve the classifier's generalizability.

We can see that the first and the third terms on the RHS of Eq. 7 form a VBI objective similar to the one in Eq. 6 but with the negative log-likelihood computed on labeled data only. Due to the scarcity of labels in SSL, it seems reasonable to take into account of the unlabeled data to model $q_\phi(w)$ better by adding $\mathbb{E}_{w\sim q_\phi(w)}\left[\mathbb{E}_{x\sim\mathcal{D}}\left[-\log p(x|w)\right]\right]$ to Eq. 7. However, there are some difficulties: i) we need to create an additional model for $p(x)$ that *shares weights* with the default classifier, and ii) the impact of modeling both $p(x|w)$ and $p(y|x,w)$ using the same w on the classifier's generalizability is unclear. We observe empirically that the loss in Eq. 7 produces good results, thus, we leave modeling $p(x|w)$ for future work with a note that the work by Grathwohl et. al. (Grathwohl et al. 2019) may provide a good starting point.

To implement VBI, we adopt the variational dropout (VD) technique from (Molchanov, Ashukha, and Vetrov 2017). Our justification for this is presented in the appendix.

Maximum Uncertainty Regularization

Some CR based methods enforce smoothness on the vicinity of training data points by using standard data perturbation (DP) techniques (e.g., Gaussian noise, dropout). However, there are points in the input-output manifold unreachable by standard DPs. These *"virtual"* points usually lie beyond the local area of real data points and prevent a smooth transition of the class prediction from a data point to another. We argue that if we can find such "virtual" points and force their class predictions to be similar to those of nearby data points, we will learn a smoother classifier that generalizes better. We do this by introducing a novel consistency loss called *maximum uncertainty regularization* (MUR) $\mathcal{L}_{\text{MUR}}$. In case the classifier's weights θ are *deterministic*, $\mathcal{L}_{\text{MUR}}$ is given by:

$$\mathcal{L}_{\text{MUR}}(\theta) = \mathbb{E}_{x_0\sim\mathcal{D}}\left[\frac{1}{K}\sum_{k=1}^{K}\left(p(k|x^*,\theta) - p(k|x_0,\theta_{\text{sg}})\right)^2\right] \quad (8)$$

where x^* is mathematically defined as follows:

$$x^* = \underset{x}{\text{argmax}}\ H(p(y|x)) \quad \text{s.t.} \quad \|x - x_0\|_2 \le r \quad (9)$$

where $H(\cdot)$ is the Shannon entropy, $r \in \mathbb{R}^+$ is the largest distance between x^* and x_0. In general, it is hard to compute x^* exactly because the objective in Eq. 9 usually has many local optima. However, we can approximate x^* by optimizing a linear approximation of $H(p(y|x))$ instead. In this case, the original optimization problem becomes *convex minimization* and it has a unique solution which is:

$$x^* \approx \tilde{x}^* = x_0 + r\frac{g_0}{\|g_0\|_2} \quad (10)$$

where $g_0 = \frac{\partial H(p(y|x))}{\partial x}\big|_{x=x_0}$ is the gradient of $H(p(y|x))$ at $x = x_0$. Its derivation is presented in the appendix.

Iterative Approximations of x^* Linearly approximating $H(p(y|x))$ may cause some information loss. We can avoid that by optimizing Eq. 9 directly via *projected gradient ascent* (details in the appendix). Alternatively, *vanilla gradient ascent* update based on the Lagrangian relaxation of Eq. 9 can be done via maximizing:

$$\mathcal{F}(x) = H(p(y|x)) - \lambda^*(x)\left(\|x - x_0\|_2 - r\right) \quad (11)$$

where $\lambda^*(x) = \frac{\|x-x_0\|_2\|g_0\|_2}{r}$. Insight on $\lambda^*(x)$ is given in the appendix.

Connection to Adversarial Learning Adversarial learning (AL) (Szegedy et al. 2013) aims to build a system robust against various types of adversarial attacks. Madry et. al. (Madry et al. 2017) have shown that these methods attempt to solve the following saddle point problem:

$$\min_\theta \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\sup_{\epsilon\in\mathcal{S}} L\left(f_\theta\left(x+\epsilon\right),y\right)\right] \quad (12)$$

where L is a loss function (e.g., the cross-entropy), $\mathcal{S}$ is the support set of the adversarial noise ϵ. For example, in case of Fast Gradient Sign Method (Goodfellow, Shlens, and Szegedy 2014), $\mathcal{S}$ is defined as $\mathcal{S} = \{\epsilon : \|\epsilon\|_\infty \leq r\}$. Adversarial learning has also been shown by Sinha et. al. (Sinha, Namkoong, and Duchi 2017) to be closely related to *(distributionally) robust optimization* (Farnia and Tse 2016; Globerson and Roweis 2006) whose objective is given by:

$$\min_{\theta} \sup_{p \in \mathcal{P}(\mathcal{D})} \mathbb{E}_{(x,y)\sim p} \left[L\left(f_\theta(x), y\right) \right] \tag{13}$$

where $\mathcal{P}(\mathcal{D})$ is a class of distributions derived from the empirical data distribution $\mathcal{D}$.

At the high level, MUR (Eqs. 8, 9) is similar to AL (Eq. 12) as both consist of two optimization sub-problems: an *inner maximization w.r.t. the data* and an *outer minimization w.r.t. the parameters*. However, when looking closer, there are some differences between MUR and AL: In MUR, the two sub-problems optimize two distinct objectives (the consistency loss and the conditional entropy) while in AL, the two sub-problems share the same objective. Moreover, since MUR's objectives do not use label information, MUR is applicable to SSL while AL is not.

Compared to virtual adversarial training (VAT) (Miyato et al. 2018), MUR is different in how x^* is chosen. VAT defines x^* to be a point in the local neighborhood of x_0 whose output $p(y|x^*)$ is the most different from $p(y|x_0)$. It means that x^* can have very low $H(p(y|x^*))$ as long as its corresponding pseudo class is different from the (pseudo) class of x_0. MUR, by contrast, always looks for x^* with the highest $H(p(y|x^*))$ regardless of the (pseudo) class of x_0. Inspired by VAT and MUR, we propose a new CR based method called *maximum uncertainty training* (MUT) with the loss function defined as:

$$\mathcal{L}_{\text{MUT}}(\theta) = \mathcal{L}_{\text{xent},l}(\theta) + \lambda(t)\mathcal{L}_{\text{MUR}}(\theta)$$

MUT can be seen as a special case of Π+MUR in which the coefficient of $\mathcal{L}_{\Pi,\text{cons}}(\theta, \theta_{\text{sg}})$ equals 0. We can also view it as a variant of Π-model (Eq. 2) with $\mathcal{L}_{\Pi,\text{cons}}(\theta, \theta_{\text{sg}})$ replaced by $\mathcal{L}_{\text{MUR}}(\theta)$. Note that for other CR based methods like MT or ICT, their original consistency losses cannot be replaced by $\mathcal{L}_{\text{MUR}}$ since these losses and $\mathcal{L}_{\text{MUR}}$ are inherently different. For example, in MT, $\mathcal{L}_{\text{MT,cons}}(\theta, \bar{\theta})$ involves both the student and teacher networks while $\mathcal{L}_{\text{MUR}}(\theta)$ only involves the student network.

Experiments

We now show that using VBI (or VD in particular) and MUR leads to significant improvements in performance and generalization of CR based methods that do not use strong data augmentation. For methods that use strong data augmentation (e.g., FixMatch (Sohn et al. 2020)), results are discussed in the appendix. We evaluate our approaches on three standard benchmark datasets: SVHN, CIFAR-10 and CIFAR-100. Details about the datasets, data preprocessing scheme, the classifier's architecture and settings, and the training hyperparameters are all provided in the appendix.

Classification Results on SVHN, CIFAR-10 and CIFAR-100

In Tables 1 and 2, we compare the classification errors of state-of-the-art CR based methods with/without using VD and MUR on SVHN, CIFAR-10, and CIFAR-100. Results of the baselines are taken from existing literature. We provide results from our own implementations of some baselines when necessary. Each setting of our models is run 3 times.

SVHN When there are 500 and 1000 labeled samples, combining Π with MUR reduces the error by about 1-2% compared to the plain one. In case VD is used instead of MUR, the error reduction is about 0.5-0.9%. It suggests that MUR is more helpful for Π than VD. On the other hand, when the base model is MT, using VD leads to bigger improvements than using MUR. Interestingly, using both VD and MUR for MT boosts the performance even further. By contrast, using both VD and MUR for Π leads to higher error with larger variance compared to using individual methods. We think the main cause is the inherent instability of Π as this model does not use weight averaging for prediction like MT. Thus, too much randomness from both VD and MUR can be harmful for Π.

When the number of labels is 250, our implementations of Π and MT yield much poorer results than the original models. However, we note that the same problem can also be found in (Berthelot et al. 2019b) (Table 6) and (Sohn et al. 2020) (Table 2). Therefore, to ensure fair comparison, we only consider the results of our implementations. While using VD still improves the performances of Π and MT, using MUR hurts the performances of these models. A possible reason is that with too few labeled examples, the classifier is unable to learn correct class-intrinsic features (unless strong data augmentation is given), hence, the gradient of $H(p(y|x))$ w.r.t. x (Eq. 9) may point to wrong directions.

CIFAR-10/CIFAR-100 We observe the same pattern for MT on CIFAR-10 and CIFAR-100 as on SVHN: Using VD+MUR leads to much better results than using either VD or MUR. Specifically on CIFAR-10, VD+MUR decreases the errors of MT by about 3-4.5% while for VD and MUR, the amounts of error reduction are 2-2.7% and 1.5-2.8%, respectively. Compared to MT+FSWA (Athiwaratkun et al. 2018), our MT+VD+MUR achieves slightly better results on CIFAR-10 but perform worse on CIFAR-100. The reason could be that they use better settings for CIFAR-100 than ours, which is reflected in the lower error of their MT compared to our reimplemented MT. However, we want to note that FSWA only provides MT with advanced learning rate scheduling (Loshchilov and Hutter 2016) and postprocessing (Izmailov et al. 2018) but does not change the objective of MT like VD or MUR. It means one can easily combine MT+VD+MUR with FSWA to further improve the results.

In case of ICT, using VD leads to impressive decreases of error by 1.7-4% on CIFAR-10 and by 1.5% on CIFAR-100. Meanwhile, MUR only improves the results slightly, by about 0.6-1% on CIFAR-10 and by 0.7% on CIFAR-100. The performance of ICT+VD+MUR is also just comparable

Model	CIFAR-10 1000	CIFAR-10 2000	CIFAR-10 4000	CIFAR-100 10000
Π$^{\heartsuit}$	31.65 ±1.20	17.57±0.44	12.36±0.31	39.19±0.54
Π + FSWA$^{\diamondsuit}$	17.23±0.34	12.61±0.18	10.07±0.27	34.25±0.16
TempEns + SNTG$^{\clubsuit}$	18.14±0.52	13.64±0.32	10.93±0.14	-
VAT$^{\spadesuit}$	-	-	10.55±0.05	-
MT$^{\heartsuit}$	21.55±1.48	15.73±0.31	12.31±0.28	-
MT$^{\diamondsuit}$	18.78±0.31	14.43±0.20	11.41±0.27	35.96±0.77
MT + FSWA$^{\diamondsuit}$	15.58±0.12	11.02±0.23	9.05±0.21	33.62±0.54
MT*	19.63±0.33	15.07±0.10	11.65±0.09	37.65±0.25
MT + VD*	16.35±0.18	12.51±0.43	9.62±0.13	35.47±0.21
MT + MUR*	17.96±0.32	12.23±0.21	10.16±0.04	35.93±0.32
MT + VD + MUR*	**15.47±0.13**	**10.57±0.28**	**8.54±0.20**	35.24±0.06
ICT$^{\spadesuit}$	15.48±0.78	9.26±0.09	7.92±0.02	-
ICT*	14.15±0.16	11.56±0.07	9.18±0.03	35.67±0.07
ICT + VD*	**10.13±0.21**	8.83±0.15	**7.48±0.11**	34.12±0.16
ICT + MUR*	13.54±0.23	10.49±0.07	8.55±0.06	34.91±0.20
ICT + VD + MUR*	10.37±0.25	**8.79±0.16**	7.55±0.14	**33.21±0.24**
Co-train (8 views)†	-	-	8.35±0.06	-

Table 1: Classification errors on CIFAR-10 and CIFAR-100. $^{\heartsuit}$: (Tarvainen and Valpola 2017), $^{\diamondsuit}$: (Athiwaratkun et al. 2018), $^{\clubsuit}$: (Luo et al. 2018), $^{\spadesuit}$: (Verma et al. 2019), †: (Qiao et al. 2018), *: Our implementations.

Model	250	500	1000
Π$^{\heartsuit}$	9.69±0.92	6.65±0.53	4.82±0.17
Π + SNTG$^{\diamondsuit}$	5.07±0.25	4.52±0.30	3.82±0.25
Π*	13.37±0.97	7.25±0.36	5.18±0.13
Π + VD*	12.98±0.84	6.38±0.42	4.65±0.23
Π + MUR*	15.04±0.75	5.43±0.27	4.15±0.10
Π + VD + MUR*	16.63±1.22	6.57±0.73	4.72±0.48
VAT$^{\clubsuit}$	-	-	3.86±0.11
MT$^{\heartsuit}$	4.53±0.50	4.18±0.27	3.95±0.19
MT + SNTG$^{\diamondsuit}$	**4.29±0.43**	3.99±0.24	3.86±0.27
MT*	5.57±1.52	3.86±0.15	3.72±0.10
MT + VD*	5.26±1.73	3.39±0.10	3.28±0.08
MT + MUR*	6.45±1.29	3.66±0.07	3.48±0.04
MT + VD + MUR*	6.82±2.01	**3.21±0.13**	**3.16±0.07**
ICT$^{\clubsuit}$	4.78±0.68	4.23±0.15	3.89±0.04
Co-train (8 views)$^{\spadesuit}$	-	-	3.29±0.03

Table 2: Classification errors on SVHN. $^{\heartsuit}$: (Tarvainen and Valpola 2017), $^{\diamondsuit}$: (Luo et al. 2018), $^{\clubsuit}$: (Verma et al. 2019), $^{\spadesuit}$: (Qiao et al. 2018), *: Our implementations.

to that of ICT+VD. A possible reason is that because ICT enforces smoothness on points interpolated between pairs of random real data points which, to some extent, are similar to the "virtual" points in MUR. Thus, the regularization effect of ICT may overlap that of MUR.

Effects of VBI and MUR on Sensitivity

Besides accuracy, we should also examine *sensitivity* since this metric is closely related to generalization (Alain and Bengio 2014; Novak et al. 2018; Rifai et al. 2011). The *sensitivity* of a classifier f w.r.t. small changes of a data point

	Sensitivity MT	Sensitivity ICT
Default	0.21±0.26	0.22±0.44
+VD	0.19±0.30	0.20±0.46
+MUR	0.12±0.16	0.16±0.32
+VD+MUR	0.13±0.29	0.19±0.38

Table 3: Sensitivities of MT, ICT and their variants trained on CIFAR-10 with 1000 labels.

x is measured as the Frobenius norm of the Jacobian matrix of f w.r.t. x:

$$\text{Sensitivity}(x) = \|J(x)\|_F = \sqrt{\sum_{i,j} J_{i,j}^2(x)}$$

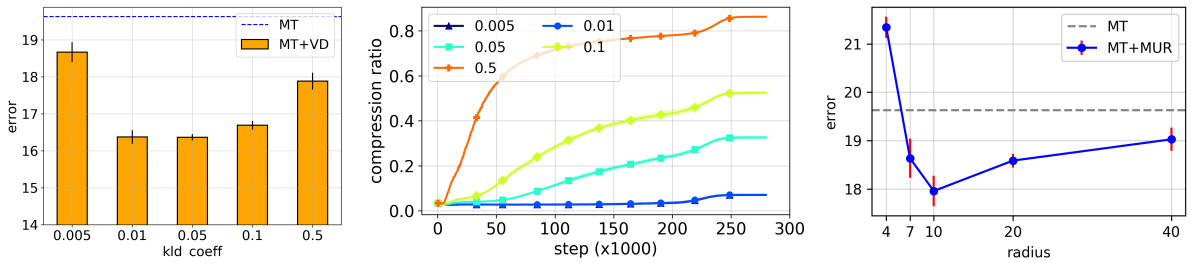
where $J_{i,j}(x) = \frac{\partial f(x)_i}{\partial x_j} = \frac{\partial p(y=i|x)}{\partial x_j}$. A low sensitivity value means that the local area surrounding x is flat[1] and f is robust to small variations of x.

Compared to MT and ICT, the corresponding variants using VD and/or MUR achieve lower sensitivity on average (Table 3) and have more test data points with sensitivities close to 0 (Figs. 2, 3 in the appendix). These results empirically verify that VD and MUR actually make the classifier smoother.

Ablation Study

The coefficient of $D_{KL}\left(q_\phi(w)\|p(w)\right)$ in VBI In Fig. 2a, we compare the errors of MT+VD w.r.t. different values

[1]It should not be confused with flat minima (Hochreiter and Schmidhuber 1997) in the weight space.



(a) classification error (b) sparsity curve (c) classification error

Figure 2: (a), (b): Classification errors and weight sparsity curves of MT+VD w.r.t. different coefficients of the KLD of weights. (c): Classification errors of MT+MUR w.r.t. different values of the radius r. The dataset is CIFAR-10 with 1000 labels.

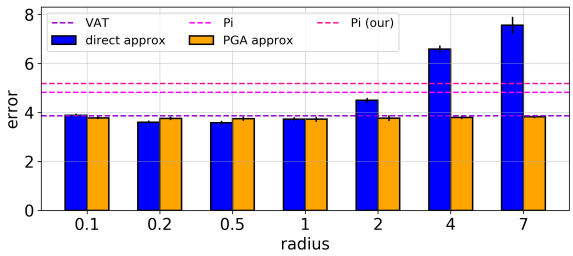


Figure 3: Classification errors of MUT w.r.t. different values of the radius r. For the PGA update of x^*, lr=0.3 and #steps=5. The dataset is SVHN with 1000 labels.

of the coefficient of $D_{KL}\left(q_\phi(w)\|p(w)\right)$[2]. Either too small or too large coefficients lead to inferior results since they correspond to too little or too much regularization on weights (Fig. 2b). However, even in the worst setting, MT+VD still outperforms MT. This again demonstrates the clear advantage of VBI in improving the robustness of models.

The radius r in MUR We now examine how the radius r (Eq. 9) affects performance. If r is too small, it is hard to find an adequate virtual point x^* that the classifier f is uncertain about. Moreover, as x^* is very close to x_0, minimizing $\mathcal{L}_{\text{MUR}}$ causes f to focus too much on ensuring the local flatness around x_0 instead of smoothing the area between x_0 and other data points, exacerbating the problem. By contrast, if r is too big, x^* is very different from x_0 and forcing consistency between these points may be inappropriate. Fig. 2c shows the error of MT+MUR on CIFAR-10 with 1000 labels as a function of r ($r \in \{4, 7, 10, 20, 40\}$), which reflects the intuition presented: MT+MUR performs poorly when r is too small (4) and worse than MT. When r is too big (20, 40), the results are also not good. The optimal value of r is 10. To make sure that this result is reasonable, we visualize the virtual samples w.r.t. different values of r in the appendix.

Iterative approximations of x^* in MUR We investigate the performance of MT+MUR when iterative approximations of x^* are used instead of the direct (linear) approximation (Eq. 10). We try both projected gradient ascent (PGA) and vanilla gradient ascent (GA) updates with the learning rate α varying in $\{0.1, 1.0, 10.0\}$ and the number of steps s varying in $\{2, 5, 8\}$. We report results of the GA update in Fig. 4. . Clearly, larger α and s both lead to smaller gradient norms of (real) data points ($\|g_0\|_2$ in Eq. 10) (Fig. 4b) and causes the model to learn faster early in training. However, if α is too large (10.0), the model performance tends to degrade over time. If α is too small (0.1), the results are usually suboptimal when s is small (2) and many update steps are required to achieve good results (8). The best setting of the GA update is (α, s)=(0.1,8) at which the error is 17.21, smaller than the error in case the direct approximation is used (17.96). (Results of the PGA update are presented in the appendix)

How important is finding the most uncertain virtual points? We define a "random regularization" loss $\mathcal{L}_{\text{RR}}$ which has the same formula as $\mathcal{L}_{\text{MUR}}$ in Eq. 8 except that it acts on a *random* virtural points $\hat{x}^*$ instead of the most uncertain virtual point x^*. $\hat{x}^*$ is computed as follows:

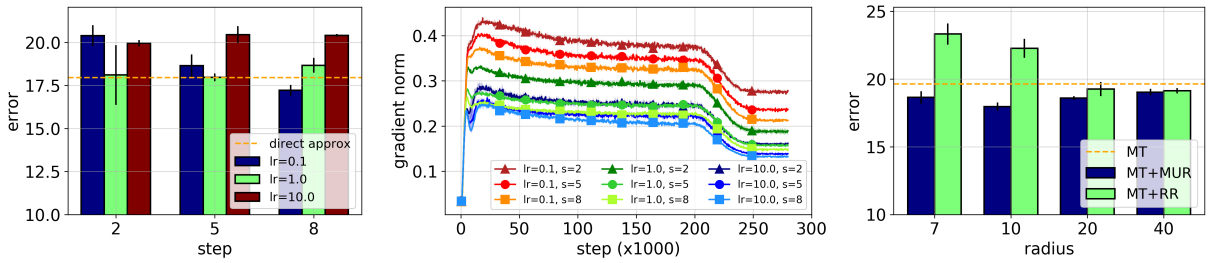
$$\hat{x}^* = x_0 + r \times \frac{u}{\|u\|_2}$$

where u is a random vector/tensor whose elements are drawn independently from a standard Gaussian distribution $\mathcal{N}(0, 1)$. Choosing u like this ensures that $\hat{x}^*$ is sampled *uniformly* on the sphere of radius r centered at x_0 (Muller 1959).

We compare MT+MUR (with the direct approximation of x^*) against MT combined with $\mathcal{L}_{\text{RR}}$ (denoted as MT+RR) w.r.t. different values of r and show the results in Fig. 4c. The advantage of finding the most uncertain virtual points is clear when r is not too big (e.g., 7 or 10). However, when r becomes bigger and bigger (e.g., 20 or 40), this advantage disappears and MT+MUR performs similarly to MT+RR. We think the main reason is that when r is big, the direct approximation of x^* (Eq. 10) is no longer correct, making x^* look more like a random point.

Maximum Uncertainty Training (MUT) It is important to know how well MUT performs compared to VAT and Π-model. To this end, we train MUT using the same settings

[2]The "coefficient" in this context is referred to as the *maximum* value of $\lambda_2(t)$ in Eq. 7



(a) classification error (b) $\|g_0\|_2$ (c) classification error

Figure 4: (a), (b): Classification errors and gradient norm curves of MT+MUR with the GA update of x^* w.r.t. different learning rates and numbers of steps. (c): Classification errors of MT+RR and MT+MUR w.r.t. different values of the radius r. The dataset is CIFAR-10 with 1000 labels.

for training Π-model on the SVHN dataset. The classification errors of MUT w.r.t. different values of the radius r are shown in Fig. 3. At $r = 0.5$, MUT with the direct approximation of x^* (blue bars in Fig. 3) achieves the best error of 3.58±0.08 which is smaller the reported errors of VAT (Miyato et al. 2018) (3.86±0.11) and Π-model (Laine and Aila 2016) (4.82±0.17), and the error of our own implemented Π-model (5.18±0.13 in Table 2). However, the results become worse as r increases. We believe the reason is that without the local smoothness term $\mathcal{L}_{\Pi,\text{cons}}(\theta, \theta_{\text{sg}})$ in MUT, the true most uncertain point x^* usually lies *closely* to the real data point x_0 while the direct approximation $\tilde{x}^* = x_0 + r\frac{g_0}{\|g_0\|_2}$ is always r *away from* x_0. Therefore, if r is not small enough, $\tilde{x}^*$ is no longer a correct approximation of x^*, which eliminates the smoothing effect of $\mathcal{L}_{\text{MUR}}$. Luckily, we can relax the distance constraint of $\tilde{x}^*$ by using the projected gradient ascent (PGA) update instead. To this end, the distance between $\tilde{x}^*$ and x_0 can be smaller than r, which gives us more freedom in choosing large r. This can be seen from Fig. 3 as the performance of MUT with the PGA approximation of x^* is almost unaffected by r.

Related Work

Semi-supervised learning (SSL) is a long-established research area with diverse approaches (Chapelle, Scholkopf, and Zien 2006). Within the scope of this paper, we mainly focus on "consistency regularization" (CR) based methods. Details about other methods can be found in (van Engelen and Hoos 2019) and (Ouali, Hudelot, and Tami 2020). CR based methods aim at learning a smooth classifier by forcing it to give similar predictions for different perturbed inputs. There are many ways to define data perturbation but usually, the more distinct the two perturbations are, the smoother the classifier is. Standard methods such as Ladder Network (Rasmus et al. 2015), Π-model (Laine and Aila 2016) and Mean Teacher (Tarvainen and Valpola 2017) perturb data by applying small additive Gaussian noise and binary dropout to the input and hidden activations. VAT (Miyato et al. 2018) and VAdD (Park et al. 2018) use adversarial noise and adversarial dropout for perturbation, respectively. Both can be viewed as performing *selective smoothing* because they only flatten the input-output manifold along the direction that gives the largest variance in class prediction.

Data augmentation is another perturbation technique which is more effective than general noise injection in specific domains because it exploits the intrinsic domain structures (Xie et al. 2019). For example, augmenting image data with different color filters and affine transformations encourages the classifier to be insensitive to changes in color and shape (Cubuk et al. 2020). In case of text, using thesaurus (Zhang, Zhao, and LeCun 2015; Mueller and Thyagarajan 2016) and back translation (Sennrich, Haddow, and Birch 2016; Edunov et al. 2018) for augmentation makes the classifier robust to different paraphrases. These generalization capabilities cannot be obtained with general noise injections. However, data augmentation also comes with several limitations such as domain dependence and requirement for external knowledge not available in the training data.

Apart from data augmentation, standard CR based methods can also be improved by using additional smoothness-inducing objectives. For example, SNTG (Luo et al. 2018) introduces a new loss that forces every two data points with similar pseudo labels (up to a certain confidence level) to be close in the low-dimensional feature space. ICT (Verma et al. 2019) is a variant of MT which leverages MixUp (Zhang et al. 2017) to encourage linearity of the classifier. Holistic methods like MixMatch (Berthelot et al. 2019b), ReMixMatch (Berthelot et al. 2019a) and FixMatch (Sohn et al. 2020) combine different advanced techniques in SSL such as strong data augmentation (Cubuk et al. 2020), MixUp (Zhang et al. 2017), entropy minimization (Grandvalet and Bengio 2005) and pseudo labeling (Lee 2013) into a unified framework that performs well yet uses very few labeled data.

Conclusion

We have presented VBI and MUR - two general methods for improving SSL. We have demonstrated that combining existing CR based methods with VBI and MUR significantly reduces errors of these methods on various benchmark datasets. In the future, we would like to incorporate the likelihood of unlabeled data into VBI to learn a better posterior distribution of classifier's weights. We also want to apply MUR to other machine learning problems that demand robustness and generalization.

References

Alain, G.; and Bengio, Y. 2014. What regularized auto-encoders learn from the data-generating distribution. *The Journal of Machine Learning Research* 15(1): 3563–3593.

Athiwaratkun, B.; Finzi, M.; Izmailov, P.; and Wilson, A. G. 2018. There are many consistent explanations of unlabeled data: Why you should average. *arXiv preprint arXiv:1806.05594* .

Bachman, P.; Alsharif, O.; and Precup, D. 2014. Learning with pseudo-ensembles. In *Advances in Neural Information Processing Systems*, 3365–3373.

Berthelot, D.; Carlini, N.; Cubuk, E. D.; Kurakin, A.; Sohn, K.; Zhang, H.; and Raffel, C. 2019a. ReMixMatch: Semi-Supervised Learning with Distribution Alignment and Augmentation Anchoring. *arXiv preprint arXiv:1911.09785* .

Berthelot, D.; Carlini, N.; Goodfellow, I.; Papernot, N.; Oliver, A.; and Raffel, C. A. 2019b. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, 5050–5060.

Blundell, C.; Cornebise, J.; Kavukcuoglu, K.; and Wierstra, D. 2015. Weight uncertainty in neural networks. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning-Volume 37*, 1613–1622.

Chapelle, O.; Scholkopf, B.; and Zien, A. 2006. *Semi-Supervised Learning*. MIT Press.

Cubuk, E. D.; Zoph, B.; Shlens, J.; and Le, Q. V. 2020. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 702–703.

Edunov, S.; Ott, M.; Auli, M.; and Grangier, D. 2018. Understanding Back-Translation at Scale. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 489–500.

Farnia, F.; and Tse, D. 2016. A minimax approach to supervised learning. In *Advances in Neural Information Processing Systems*, 4240–4248.

Globerson, A.; and Roweis, S. 2006. Nightmare at test time: robust learning by feature deletion. In *Proceedings of the 23rd international conference on Machine learning*, 353–360.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572* .

Grandvalet, Y.; and Bengio, Y. 2005. Semi-supervised Learning by Entropy Minimization. In *Advances in neural information processing systems*, 529–536.

Grathwohl, W.; Wang, K.-C.; Jacobsen, J.-H.; Duvenaud, D.; Norouzi, M.; and Swersky, K. 2019. Your Classifier is Secretly an Energy Based Model and You Should Treat it Like One. *arXiv preprint arXiv:1912.03263* .

Graves, A. 2011. Practical variational inference for neural networks. In *Advances in neural information processing systems*, 2348–2356.

Hinton, G.; and von Cramp, D. 1993. Keeping neural networks simple by minimising the description length of weights. In *Proceedings of COLT-93*, 5–13.

Hochreiter, S.; and Schmidhuber, J. 1997. Flat minima. *Neural Computation* 9(1): 1–42.

Honkela, A.; and Valpola, H. 2004. Variational learning and bits-back coding: an information-theoretic view to Bayesian learning. *IEEE transactions on Neural Networks* 15(4): 800–810.

Izmailov, P.; Podoprikhin, D.; Garipov, T.; Vetrov, D.; and Wilson, A. G. 2018. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407* .

Kingma, D. P.; Salimans, T.; and Welling, M. 2015. Variational dropout and the local reparameterization trick. In *Advances in neural information processing systems*, 2575–2583.

Kingma, D. P.; and Welling, M. 2013. Auto-encoding Variational Bayes. *arXiv preprint arXiv:1312.6114* .

Laine, S.; and Aila, T. 2016. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242* .

Lee, D.-H. 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, 2.

Loshchilov, I.; and Hutter, F. 2016. SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983* .

Luo, Y.; Zhu, J.; Li, M.; Ren, Y.; and Zhang, B. 2018. Smooth Neighbors on Teacher Graphs for Semi-Supervised Learning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8896–8905. IEEE.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083* .

Miyato, T.; Maeda, S.-i.; Koyama, M.; and Ishii, S. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence* 41(8): 1979–1993.

Molchanov, D.; Ashukha, A.; and Vetrov, D. 2017. Variational dropout sparsifies deep neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2498–2507. JMLR. org.

Mueller, J.; and Thyagarajan, A. 2016. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2786–2792.

Muller, M. E. 1959. A note on a method for generating points uniformly on n-dimensional spheres. *Communications of the ACM* 2(4): 19–20.

Novak, R.; Bahri, Y.; Abolafia, D. A.; Pennington, J.; and Sohl-Dickstein, J. 2018. Sensitivity and generalization in neural networks: an empirical study. *arXiv preprint arXiv:1802.08760* .

Ouali, Y.; Hudelot, C.; and Tami, M. 2020. An Overview of Deep Semi-Supervised Learning. *arXiv preprint arXiv:2006.05278* .

Park, S.; Park, J.; Shin, S.-J.; and Moon, I.-C. 2018. Adversarial dropout for supervised and semi-supervised learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Qiao, S.; Shen, W.; Zhang, Z.; Wang, B.; and Yuille, A. 2018. Deep co-training for semi-supervised image recognition. In *Proceedings of the european conference on computer vision (eccv)*, 135–152.

Rasmus, A.; Berglund, M.; Honkala, M.; Valpola, H.; and Raiko, T. 2015. Semi-supervised learning with ladder networks. In *Advances in neural information processing systems*, 3546–3554.

Rezende, D. J.; Mohamed, S.; and Wierstra, D. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. *arXiv preprint arXiv:1401.4082* .

Rifai, S.; Vincent, P.; Muller, X.; Glorot, X.; and Bengio, Y. 2011. Contractive Auto-Encoders: Explicit Invariance During Feature Extraction. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, 833–840. Omnipress.

Sajjadi, M.; Javanmardi, M.; and Tasdizen, T. 2016. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *Advances in Neural Information Processing Systems*, 1163–1171.

Sennrich, R.; Haddow, B.; and Birch, A. 2016. Improving Neural Machine Translation Models with Monolingual Data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 86–96.

Sinha, A.; Namkoong, H.; and Duchi, J. 2017. Certifying some distributional robustness with principled adversarial training. *arXiv preprint arXiv:1710.10571* .

Sohn, K.; Berthelot, D.; Li, C.-L.; Zhang, Z.; Carlini, N.; Cubuk, E. D.; Kurakin, A.; Zhang, H.; and Raffel, C. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint arXiv:2001.07685* .

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15(1): 1929–1958.

Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; and Fergus, R. 2013. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199* .

Tarvainen, A.; and Valpola, H. 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*, 1195–1204.

van Engelen, J. E.; and Hoos, H. H. 2019. A survey on semi-supervised learning. *Machine Learning* 1–68.

Verma, V.; Lamb, A.; Kannala, J.; Bengio, Y.; and Lopez-Paz, D. 2019. Interpolation consistency training for semi-supervised learning. *arXiv preprint arXiv:1903.03825* .

Xie, Q.; Dai, Z.; Hovy, E.; Luong, M.-T.; and Le, Q. V. 2019. Unsupervised data augmentation for consistency training .

Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2017. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412* .

Zhang, X.; Zhao, J.; and LeCun, Y. 2015. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, 649–657.