Virtual Adversarial Training:A Regularization Method for Supervised andSemi-Supervised Learning
(2017)

Takeru Miyato et al.
Resume

May 7, 2019

1 Introduction

The authors propose a new regularization method based on a virtual adversarial loss, virtual given that in the calculation of the loss the true labels are not needed, VAT measure the local smoothness of the outputs (label distribution) conditionned on the inputs, and finds the error to add to the inputs in the direction that is most sensitive, VAT can be used in supervised and semi-supervised settings.

VAT trains the output distribution to be isotropically smooth around each data point (identical in all directions) by selectively smoothing the most anisotropic (most sensitive to perturbations) diretion. Instead of finding the adversarial direction of the perturbation that can most reduce the models probability of correct classification, or the direction that can most greatly deviate the prediction of the model from the correct label, VAT direction can be defined on unlabeled data point, because it is the direction that can most greatly deviate the current inferred output distribution from the status quo.

Advantages of VAT:

- applicability to semi-supervised learning tasks
- applicability to any parametric models for which we can evaluate the gradient with respect to input and parameter
- small number of hyperparameters
- parametrization invariant regularization

2 Virtual Adversarial Training

Let x_* represent either a labeled x_l or an unlabeled example x_{ul}, the objective function is now given by

$$D\left[q\left(y|x_*\right), p\left(y|x_* + r_{\text{qadv}}, \theta\right)\right]$$
$$\text{where } r_{\text{qadv}} := \underset{r;\|r\|\leq\epsilon}{\arg\max} D\left[q\left(y|x_*\right), p\left(y|x_* + r, \theta\right)\right.$$

where $D[q, p]$ is a non-negative function that measures the divergence between two distributions, true distribution q and predicted distribution p, and r_{qadv} is the virtual adversarial perturbation, in a unsupervised setting we have no information about q so we replace it by the current approximation

$p(y|x,\theta)$, which is close to the q given a large number of labeled exampled, so the VAT loss will be rewritten as:

$$\mathrm{LDS}\left(x_*,\theta\right) := D\left[p\left(y|x_*,\hat{\theta}\right), p\left(y|x_*+r_{\mathrm{vadv}},\theta\right)\right]$$
$$r_{\mathrm{vadv}} := \arg\max_{r:\|r\|_2\leq\epsilon} D\left[p\left(y|x_*,\hat{\theta}\right), p\left(y|x_*+r\right)\right]$$

And the total objective will be:

$$\ell\left(\mathcal{D}_l,\theta\right) + \alpha\mathcal{R}_{\mathrm{vadv}}\left(\mathcal{D}_l,\mathcal{D}_{ul},\theta\right)$$

Where $\mathcal{R}_{\mathrm{vadv}}$ is the normalized VAT loss:

$$\mathcal{R}_{\mathrm{vadv}}\left(\mathcal{D}_l,\mathcal{D}_{ul},\theta\right) := \frac{1}{N_l+N_{ul}} \sum_{x_*\in\mathcal{D}_l,\mathcal{D}_{ul}} \mathrm{LDS}\left(x_*,\theta\right)$$

And the virtual adersarial perturbation is calculated as follows:

$$r_{\mathrm{vadv}} \approx \epsilon\frac{g}{\|g\|_2}$$
$$\text{where } g = \nabla_r D\left[p(y|x,\hat{\theta}), p(y|x+r,\hat{\theta})\right]\Big|_{r=\xi d}$$

So we first sample a random unit vector / tensor, normalize it, get the current appriximation $p(x|y)$ by passing the clean example through the network, we then add the random vecotors d to the inputs: $x+\xi\times d$, we pass it throuhgh the network to get the perturbated output: $p(y|x+r_{\mathrm{vadv}})$, we calculate the distance beween the two distributions and then we get the grandient of the random vector d with respec to this divergence, we normalize the gradient of d, and finnaly we get: $r_{\mathrm{vadv}} = \mathrm{grad}_d \times \epsilon$, we can also calculate the grads of d with a number of iterations for a better approximation, here is the algorithm:

Algorithm 1 Mini-batch SGD for $\nabla_\theta\mathcal{R}_{\mathrm{vadv}}(\theta)|_{\theta=\hat{\theta}}$, with a one-time power iteration method.

1) Choose M samples of $x^{(i)}(i=1,\ldots,M)$ from dataset $\mathcal{D}$ at random.
2) Generate a random unit vector $d^{(i)}\in R^I$ using an iid Gaussian distribution.
3) Calculate r_{vadv} via taking the gradient of D with respect to r on $r=\xi d^{(i)}$ on each input data point $x^{(i)}$:
   $g^{(i)} \leftarrow \nabla_r D\left[p(y|x^{(i)},\hat{\theta}), p(y|x^{(i)}+r,\hat{\theta})\right]\Big|_{r=\xi d^{(i)}}$,
   $r^{(i)}_{\mathrm{vadv}} \leftarrow g^{(i)}/\|g^{(i)}\|_2$
4) **Return**
   $$\nabla_\theta\left(\frac{1}{M}\sum_{i=1}^{M} D\left[p(y|x^{(i)},\hat{\theta}), p(y|x^{(i)}+r^{(i)}_{\mathrm{vadv}},\theta)\right]\right)\Big|_{\theta=\hat{\theta}}$$

3 Experiments

TABLE 1: Test performance of supervised learning methods on MNIST with 60,000 labeled examples in the permutation invariant setting. The top part cites the results provided by the original paper. The bottom part shows the performance achieved by our implementation.

Method	Test error rate(%)
SVM (Gaussian kernel)	1.40
Dropout [39]	1.05
Adversarial, L_∞ norm constraint [14]	0.78
Ladder networks [33]	**0.57** ($\pm$0.02)
Baseline (MLE)	1.11 ($\pm$0.06)
RPT	0.84 ($\pm$0.03)
Adversarial, L_∞ norm constraint	0.79 ($\pm$0.03)
Adversarial, L_2 norm constraint	0.71 ($\pm$0.03)
VAT	0.64 ($\pm$0.05)

TABLE 2: Test performance of supervised learning methods implemented with CNN on CIFAR-10 with 50,000 labeled examples. The top part cites the results provided by the original paper. The bottom part shows the performance achieved by our implementation.

Method	Test error rate(%)
Network in Network [26]	8.81
All-CNN [38]	7.25
Deeply Supervised Net [25]	7.97
Highway Network [40]	7.72
ResNet (1001 layers) [17]	4.62 ($\pm$0.20)
DenseNet (190 layers) [18]	**3.46**
Baseline (only with dropout)	6.67 ($\pm$0.07)
RPT	6.30 ($\pm$0.04)
VAT	5.81 ($\pm$0.02)

Method	Test error rate(%) SVHN $N_l = 1000$	CIFAR-10 $N_l = 4000$
SWWAE [48]	23.56	
*Skip DGM [27]	16.61 ($\pm$0.24)	
*Auxiliary DGM [27]	22.86	
Ladder networks, Γ model [33]		20.40 ($\pm$0.47)
CatGAN [37]		19.58 ($\pm$0.58)
GAN with FM [36]	8.11 ($\pm$1.3)	18.63 ($\pm$2.32)
$\prod$ model [24]	5.43 ($\pm$0.25)	16.55 ($\pm$0.29)
(on Conv-Small used in [36])		
RPT	8.41 ($\pm$0.24)	18.56 ($\pm$0.29)
VAT	**6.83** ($\pm$0.24)	**14.87** ($\pm$0.13)
(on Conv-Large used in [24])		
VAT	5.77 ($\pm$0.32)	14.18 ($\pm$0.38)
VAT+EntMin	**4.28** ($\pm$0.10)	**13.15** ($\pm$0.21)

Table 5 shows the performance of VAT and contemporary semi-supervised learning methods implemented with moderate image data augmentation (translation and horizontal flip). All methods other than VAT were implemented with the strong standing assumption on the unlabeled training samples that the label of the image does not change by the deformation. On CIFAR-10, VAT still outperformed the listed methods with this handicap. 'VAT+EntMin' with moderate data augmentation also outperformed the current state-of-the-art methods for semi-supervised learning on both SVHN and CIFAR-10. This result tells us that the effect of VAT does not overlap so much with that of the data augmentation methods, so that they can be used in combination to boost the performance.