Tri-net for Semi-Supervised Deep Learning*

Dong-Dong Chen, Wei Wang, Wei Gao, Zhi-Hua Zhou
National Key Laboratory for Novel Software Technology
Nanjing University, Nanjing 210023, China
{chendd, wangw, gaow, zhouzh}@lamda.nju.edu.cn

Abstract

Deep neural networks have witnessed great successes in various real applications, but it requires a large number of labeled data for training. In this paper, we propose tri-net, a deep neural network which is able to use massive unlabeled data to help learning with limited labeled data. We consider model initialization, diversity augmentation and pseudo-label editing simultaneously. In our work, we utilize output smearing to initialize modules, use fine-tuning on labeled data to augment diversity and eliminate unstable pseudo-labels to alleviate the influence of suspicious pseudo-labeled data. Experiments show that our method achieves the best performance in comparison with state-of-the-art semi-supervised deep learning methods. In particular, it achieves 8.30% error rate on CIFAR-10 by using only 4000 labeled examples.

1 Introduction

Deep neural networks (DNNs) have become a hot wave during the past few years, and great successes have been achieved in various real applications, such as image classification [Krizhevsky *et al.*, 2012], object detection [Girshick *et al.*, 2014], scene labeling [Shelhamer *et al.*, 2017], etc. DNNs always learn a large number of parameters requiring a large amount of labeled data to alleviate overfitting. It is well-known that collecting tremendous high-quality labeled data is expensive, yet we could easily collect abundant unlabeled data in many real applications. Hence, it is desirable to use unlabeled data to improve the performance of DNNs when training with limited labeled data.

A natural idea is to combine semi-supervised learning [Chapelle *et al.*, 2006; Zhu, 2007; Zhou and Li, 2010] with deep learning. The disagreement-based learning [Zhou and Li, 2010] plays an important role in semi-supervised learning, in which co-training [Blum and Mitchell, 1998] and tri-training [Zhou and Li, 2005b] are two representatives. The basic idea of disagreement-based semi-supervised learning is to train multiple learners for the task and exploit the disagreements during the learning process. The disagreement in co-training is based on different views, while tri-training uses bootstrap sampling to get diverse training sets. Co-training has been combined with deep model for the tasks which have two views [Cheng *et al.*, 2016; Ardehaly and Culotta, 2017]. Nevertheless, in real applications, we always confront the task with one-view data, and tri-training can be utilized no matter whether there are one or more views.

In this paper, we propose tri-net which combines tri-training with deep model. We first learn three initial modules, and each module is then used to predict a pool of unlabeled data, where two modules label some unlabeled instances for another module. Later, three modules are refined by using the newly labeled examples. We consider three key techniques in tri-net, i.e., model initialization, diversity augmentation and pseudo-label editing, which can be summarized as follows: we use *output smearing* [Breiman, 2000] to help generate diverse and accurate initial modules; we fine-tune the modules in some specific rounds on labeled data to augment the diversity among them; we propose a data editing method named *DES* based on the intuition that stable pseudo-labels are more reliable. Experiments are conducted on three benchmark datasets, i.e., MNIST, SVHN and CIFAR-10, and the results demonstrate that our tri-net has good performance on all datasets. In particular, it achieves 8.45% error rate on CIFAR-10 by using only 4,000 labeled examples. With more sophisticated initialization methods, tri-net can get even better performance. For example, when we use the semi-supervised deep learning method Π model [Laine and Aila, 2016] to initialize our tri-net, we can achieve 8.30% error rate on CIFAR-10 by using only 4,000 labeled examples.

The rest of this paper is organized as follows: we introduce related work in Section 2 and present our tri-net in Section 3. Experimental results are given in Section 4. Finally, we make a conclusion in Section 5.

2 Related Work

Many methods have been proposed to tackle semi-supervised learning, we only introduce the most related ones. For more information of semi-supervised learning, see [Chapelle *et al.*, 2006; Zhu, 2007; Zhou and Li, 2010].

Disagreement-based semi-supervised learning started from the seminal paper of Blum and Mitchell [1998] on co-

*This work was supported by the NSFC (61751306, 61673202, 61503179), the Jiangsu Science Foundation (BK20150586) and the Fundamental Research Funds for the Central Universities.

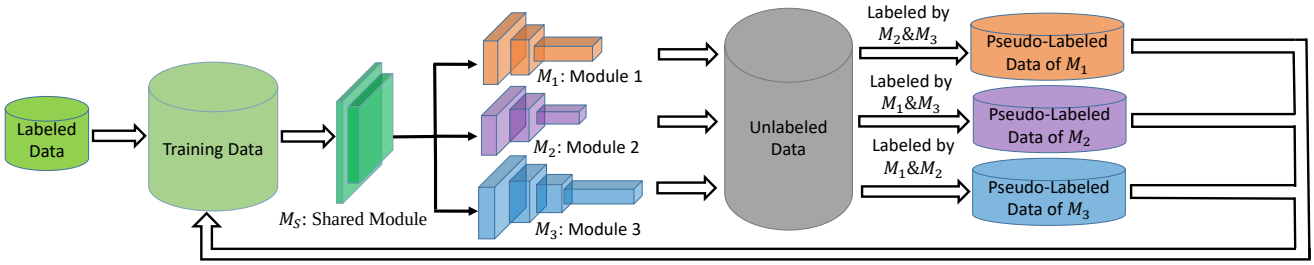

Figure 1: Training process of tri-net.

training. Co-training first learns two classifiers from two views and then lets them label unlabeled data for each other to improve performance. However, in most real applications the data sets have only one view rather than two. Some methods employed different learning algorithms or different parameter configurations to learn two different classifiers [Goldman and Zhou, 2000; Zhou and Li, 2005a]. Although these methods do not rely on the existence of two views, they require special learning algorithms to construct classifiers. Zhou and Li [2005b] proposed tri-training, which utilizes bootstrap sampling to get three different training sets and generates three classifiers from these three training sets respectively. Tri-training requires neither the existence of multiple views nor special learning algorithms, thus it can be applied to more real applications. For these algorithms, there have been some theoretical studies to explain why unlabeled data can improve the learning performance [Blum and Mitchell, 1998; Balcan *et al.*, 2004; Wang and Zhou, 2010; Balcan and Blum, 2010].

With the fast development of deep learning, disagreement-based semi-supervised learning has been combined with deep model for some applications. Cheng *et al.* [2016] developed a semi-supervised multimodal deep learning framework based on co-training to deal with the RGB-D object-recognition task. They utilized each view (i.e., RGB and depth) to learn a DNN and the two DNNs labeled unlabeled data to augment the training set. Ardehaly and Culotta [2017] combined co-training with deep model to address the demographic classification task. They generated two DNNs from two views (i.e., image and text) respectively and let them provide pseudo-labels for each other. Nevertheless, many tasks have only one view in real applications. It is more desirable to develop the disagreement-based deep models for one-view data.

There are many other methods in semi-supervised deep learning. Some of them were based on generative models. These methods paid efforts to learn the input distribution $p(x)$. Variational auto-encoder (VAE) combined variational methods with DNNs to help estimate $p(x)$ [Kingma *et al.*, 2014; Maaløe *et al.*, 2016] while generative adversarial networks (GANs) aimed to leverage a generator to detect the low-density boundaries [Salimans *et al.*, 2016; Dai *et al.*, 2017]. In contrast to the generative nature, our tri-net is a discriminative model and does not need to estimate $p(x)$. Some combined graph-based methods with deep neural networks [Weston *et al.*, 2012; Luo *et al.*, 2017]. They enforced smoothness of the predictions with respect to the graph structure while we do not need to construct the graph. Some were perturbation-based discriminative methods. They utilized local variations of the input to regularize the output to be smooth [Bachman *et al.*, 2014; Rasmus *et al.*, 2015; Laine and Aila, 2016; Sajjadi *et al.*, 2016]. VAT [Miyato *et al.*, 2017] and VAdD [Park *et al.*, 2018] introduced adversarial training [Goodfellow *et al.*, 2014] into these methods while temporal ensembling [Laine and Aila, 2016] and mean teacher [Tarvainen and Valpola, 2017] introduced ensemble learning [Zhou, 2012] into them. Compared with these state-of-the-art methods, our method can achieve better performance.

3 Our Approach

3.1 Overview

In semi-supervised learning, we have a small labeled data set $\mathcal{L} = \{(x_l, y_l)|l = 1, 2, \ldots, L\}$ with L labeled examples and a large-scale unlabeled data set $\mathcal{U} = \{(x_u)|u = 1, 2, \ldots, U\}$ with U unlabeled instances. Suppose the data have C classes and $y_l = (y_{l1}, y_{l2}, \ldots, y_{lC})$, where $y_{lc} = 1$ if the example belongs to the c-th class otherwise $y_{lc} = 0$, for $c = 1, 2, \ldots, C$. Our goal is to learn a model from the training set $\mathcal{L} \cup \mathcal{U}$ to classify unseen instances. In this paper, we propose tri-net by combining tri-training with deep neural network. Our tri-net has three phases which are described as follows.

Initialization. The first step in tri-net is to generate three accurate and diverse modules. Instead of training three networks separately, tri-net is one DNN which is composed of a shared module M_S and three different modules M_1, M_2 and M_3. Here, M_1, M_2 and M_3 classify the shared features generated by M_S. This network structure is inspired by Saito *et al.* [2017] and is efficient for implementation. In order to get three accurate and diverse modules, we use output smearing (Section 3.2) to generate three different labeled data sets, i.e., $\mathcal{L}_{os}^1$, $\mathcal{L}_{os}^2$ and $\mathcal{L}_{os}^3$. We train M_S, M_1, M_2 and M_3 simultaneously on the three data sets. Specifically, M_S and M_v are trained on $\mathcal{L}_{os}^v$ $(v = 1, 2, 3)$.

Training. In the training process, some unlabeled data will be labeled and added into the labeled training sets. In order not to change the distribution of labeled training sets, we assume that the unlabeled data are selected from a pool of $\mathcal{U}$. We use N to denote the size of the pool. This strategy is widely used in semi-supervised learning [Blum and Mitchell, 1998; Zhou and Li, 2005a; Saito *et al.*, 2017]. With three modules, if two modules agree on the prediction of the unlabeled instance from the pool and the prediction is confident

Algorithm 1 Tri-net

Input:
Labeled set $\mathcal{L}$ and unlabeled set $\mathcal{U}$
$Labeling$: the methods of labeling when the predictions of two classifiers are confident and agree with each other
DES: the methods of pseudo-label editing
σ_0: the initial threshold parameter for filtrating the unconfident pseudo-labels
σ_{os}: the value to decrease σ if output smearing is used in this learning round
Output:
Tri-net: the model composed of M_S, M_1, M_2 and M_3

Initialization:
Generate $\{\mathcal{L}_{os}^1, \mathcal{L}_{os}^2, \mathcal{L}_{os}^3\}$ by using output smearing on $\mathcal{L}$
Train M_S, M_1, M_2, M_3 with mini-batch from training set $\mathcal{L}_{os}^1$, $\mathcal{L}_{os}^2$, $\mathcal{L}_{os}^3$
$flag_{os} = 1$; $\sigma = \sigma_0$
Training:
for $t = 1 \rightarrow T$ **do**
  $N_t = min(1000 \times 2^t, U)$
  if $N_t = U$ **then**
    if $mod(t, 4) = 0$ **then**
      Train M_S, M_1, M_2, M_3 with mini-batch from training set $\mathcal{L}_{os}^1$, $\mathcal{L}_{os}^2$, $\mathcal{L}_{os}^3$
      $flag_{os} = 1$; $\sigma = \sigma - 0.05$
      continue
  if $flag_{os} = 1$ **then**
    $flag_{os} = 0$; $\sigma_t = \sigma - \sigma_{os}$
  else
    $\sigma_t = \sigma$
  for $v = 1 \rightarrow 3$ **do**
    $\mathcal{PL}_v \leftarrow \emptyset$
    $\mathcal{PL}_v \leftarrow Labeling(M_S, M_j, M_h, \mathcal{U}, N_t, \sigma_t)(j, h \neq v)$
    $\mathcal{PL}_v \leftarrow DES(M_S, \mathcal{PL}_v, M_j, M_h)$
    $\hat{\mathcal{L}}_v \leftarrow \mathcal{L} \cup \mathcal{PL}_v$
    if $v = 1$ **then**
      Train M_S, M_v with mini-batch from training set $\hat{\mathcal{L}}_v$
    else
      Train M_v with mini-batch from training set $\hat{\mathcal{L}}_v$
return M_S, M_1, M_2 and M_3

and stable, the two modules will teach the third module on this instance. The instance with the pseudo-label predicted by the two modules is added into the training sets of the third module. Then the third module is refined with the augmented training set. Here, confident prediction means that the average maximum posterior probability of the two modules is larger than the threshold σ. Stable prediction means that the pseudo-label should not change much when the modules predict the instance repeatedly and the details will be presented in Section 3.4. Three modules will be more and more similar since they augment the training sets of one another [Wang and Zhou, 2017]. To tackle this problem, we fine-tune the modules on labeled data to augment the diversity among them in some specific rounds. The whole training process is shown in Algorithm 1.

Inference. Given an unseen instance x, we use the average of the posterior probability of the three modules as the posterior probability of our method. The unseen instance x is classified with maximum posterior probability shown in Eq. 1, where M_S denotes the shared module and $M_v\big(M_S(x)\big)$ denotes the label predicted by M_v ($v = 1, 2, 3$) on x.

$$y = \underset{c\in\{1,2,\ldots,C\}}{\arg\max} \Big\{ p\big(M_1\big(M_S(x)\big) = c|x\big) + p\big(M_2\big(M_S(x)\big) = c|x\big) + p\big(M_3\big(M_S(x)\big) = c|x\big) \Big\} \quad (1)$$

3.2 Output Smearing

Output smearing was proposed by Breiman [2000]. It constructs diverse training sets by injecting random noise into true labels and generates modules from the diverse training sets respectively. Injecting noise into true labels can also regularize the modules by smoothing the labels [Szegedy *et al.*, 2016]. We apply this technique to initialize our modules M_1, M_2 and M_3. For an example $\{x_l, y_l\}$ ($l = 1, 2, \ldots, L$), where $y_l = (y_{l1}, y_{l2}, \ldots, y_{lC})$, $y_{lc} = 1$ if the example belongs to the c-th class otherwise $y_{lc} = 0$. In output smearing, we add noise into every component of y_l.

$$\hat{y}_{lc} = y_{lc} + \text{ReLU}(z_{lc} \times std) \quad (2)$$

where z_{lc} is sampled independently from the standard normal distribution, std is the standard deviation, ReLU is a function

$$\text{ReLU}(a) = \begin{cases} a, & a > 0\,, \\ 0, & a \leq 0\,. \end{cases} \quad (3)$$

Here, we use ReLU function to ensure $\hat{y}_{lc}$ non-negative and normalize $\hat{y}_{lc}$ according to Eq. 4.

$$\hat{y}_l = (\hat{y}_{l1}, \hat{y}_{l2}, \ldots, \hat{y}_{lC}) / \sum_{c=1}^{C} \hat{y}_{lc}. \quad (4)$$

With output smearing, we construct three diverse training sets $\mathcal{L}_{os}^1, \mathcal{L}_{os}^2$ and $\mathcal{L}_{os}^3$ from the initial labeled data set $\mathcal{L}$, where $\mathcal{L}_{os}^v = \{(x_l, \hat{y}_l^v) | 1 \leq l \leq L\}$ ($v = 1, 2, 3$) is constructed by output smearing and $\hat{y}_l^v$ is calculated according to Eq. 4. Then we initialize tri-net with $\mathcal{L}_{os}^1, \mathcal{L}_{os}^2$ and $\mathcal{L}_{os}^3$ by minimizing $Loss$ shown in Eq. 5.

$$Loss = \frac{1}{L} \sum_{l=1}^{L} \Big\{ L_y\big(M_1\big(M_S(x_l)\big), \hat{y}_l^1\big) + L_y\big(M_2\big(M_S(x_l)\big), \hat{y}_l^2\big) + L_y\big(M_3\big(M_S(x_l)\big), \hat{y}_l^3\big) \Big\} \quad (5)$$

Here, L_y denotes the standard softmax cross-entropy loss function, M_S denotes the shared module, M_1,M_2 and M_3 denote the three modules in tri-net, $M_v\big(M_S(x_l)\big)$ denotes the output of M_v on x_l where M_v classifies the features generated by M_S on x_l ($v = 1, 2, 3$).

3.3 Diversity Augmentation

Diversity among three modules in tri-net plays an important role in the training process. When three modules label unlabeled data to augment the training sets of one another, they become more and more similar. In order to maintain the diversity, we fine-tune three modules M_1, M_2 and M_3 on the diverse training sets $\mathcal{L}_{os}^1, \mathcal{L}_{os}^2$ and $\mathcal{L}_{os}^3$ in some specific rounds. In the experiments, the fine-tuning is executed every 3 rounds, which will be described in Section 4.

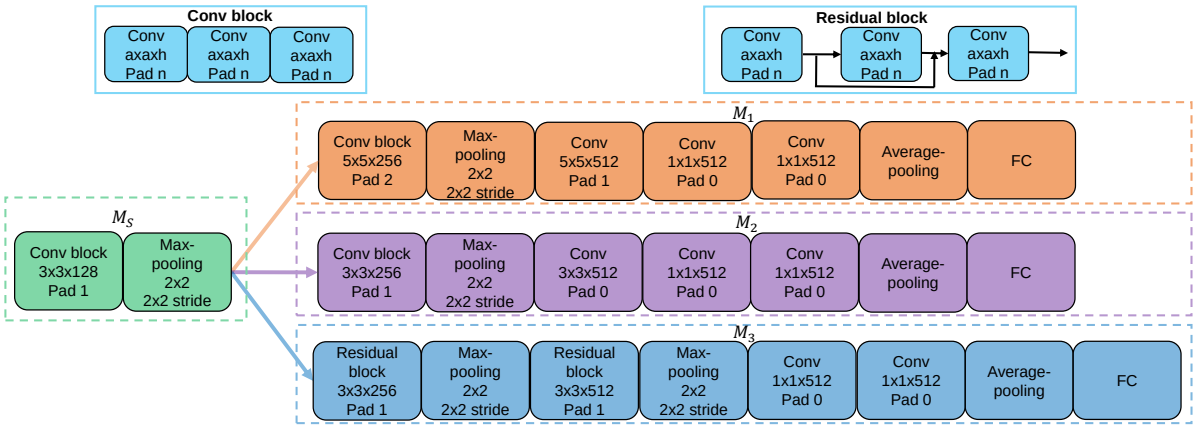

Figure 2: The architecture of tri-net. It is composed of a shared module M_S and three different modules M_1, M_2 and M_3.

3.4 Pseudo-Label Editing

The pseudo-labels of the newly labeled examples may be incorrect, and these incorrect pseudo-labels will degenerate the performance. Data editing which can deal with the suspicious pseudo-labels is important and there have been some data-editing methods in semi-supervised learning [Zhang and Zhou, 2011]. However, these existing methods are usually based on graph and are difficult to be used in DNNs due to the high dimension. Here, we propose a new data-editing method for DNNs with dropout [Srivastava *et al.*, 2014]. Generally, dropout works in two modes: at training mode, the connections of the network are different in every forward pass; at test mode, the connections are fixed. This means that the prediction for dropout working in training mode may change. For each $(x_i, \overline{y}_i)$, $\overline{y}_i$ is the pseudo-label predicted by the modules working in test mode. We use dropout working in train mode to measure the stability of the pseudo-labeled data, i.e., we use the modules to predict the label of x_i for K times in training mode and record the frequency k that the prediction is different from $\overline{y}_i$. If $k > \frac{K}{3}$, we regard the pseudo-label $\overline{y}_i$ of x_i as an unstable pseudo-label. For these unstable pseudo-labels, we will eliminate them. We set $K = 9$ in all experiments.

4 Experiments

4.1 Setup

Datasets. We run experiments on three widely used benchmark datasets, i.e., MNIST, SVHN, and CIFAR-10. We randomly sample 100, 1,000, and 4,000 labeled examples from MNIST, SVHN and CIFAR-10 as the initial labeled data set $\mathcal{L}$ respectively and use the standard data split for testing as that in previous work.

Network Architectures. The network architecture of tri-net for CIFAR-10 is shown in Figure 2, which is derived from the popular architecture [Laine and Aila, 2016] used in semi-supervised deep learning. In order to get more diversity among three modules, we use different convolution kernel sizes, different network structures (with/without residual block) and different depths for M_1, M_2 and M_3. The network architectures for MNIST and SVHN are similar to that in Figure 2 but in a smaller size.

Parameters. In order to prevent the network from overfitting, we gradually increase the pool size $N = 1000 \times 2^t$ up to the size of unlabeled data U [Saito *et al.*, 2017], where t denotes the learning round. The maximal learning round T is set to be 30 in all experiments. We gradually decrease the confidence threshold σ after $N = U$ to make more unlabeled data to be labeled (line 11, Algorithm 1). In the training process, we respectively fine-tune three modules M_1, M_2 and M_3 on the diverse training sets $\mathcal{L}^1_{os}$, $\mathcal{L}^2_{os}$ and $\mathcal{L}^3_{os}$ every 3 rounds after $N = U$ to maintain the diversity (line 10, Algorithm 1). Since $\mathcal{L}^1_{os}$, $\mathcal{L}^2_{os}$ and $\mathcal{L}^3_{os}$ are injected into random noise, the confidence threshold σ is decreased by σ_{os} (line 14, Algorithm 1). We set $\sigma_0 = 0.999$ and $\sigma_{os} = 0.01$ in MNIST; $\sigma_0 = 0.95$ and $\sigma_{os} = 0.25$ in SVHN and CIFAR-10. We use dropout ($p = 0.5$) after each max-pooling layer, use Leaky-ReLU ($\alpha = 0.1$) as activate function except the FC layer, and use soft-max for FC layer. We also use Batch-Normalization [Ioffe and Szegedy, 2015] for all layers except the FC layer. We use SGD with a mini-batch size of 16. The learning rate starts from 0.1 in initialization (from 0.02 in training) and is divided by 10 when the error plateaus. In initialization, three modules M_1, M_2 and M_3 are trained for up to 300 epochs in SVHN and CIFAR-10 (100 in MNIST). In training, three modules M_1, M_2 and M_3 are trained for up to 90 epochs in SVHN and CIFAR-10 (60 in MNIST). We set $std = 0.05$ in SVHN and CIFAR-10 (0.001 in MNSIT). We use a weight decay of 0.0001 and a momentum of 0.9. Following the setting in Laine and Aila [2016], we use ZCA, random crop and horizon flipping for CIFAR-10, zero-mean normalization and random crop for SVHN.

4.2 Results

We compare our tri-net with state-of-the-art methods shown in Table 1. Recently, Abbasnejad *et al.* [2017] exploited a pre-trained model in their infinite Variational Autoencoder (infinite VAE) method, however, the state-of-the-art methods did not use the pre-trained model. To make a fair comparison, we do not exploit the pre-trained model as that in state-of-the-art methods. The results in Table 1 indicate that tri-net has good performance. It achieves the error rate of 0.53% on

Table 1: Error rates (%) of methods on MNIST, SVHN and CIFAR-10. * indicates that the method does not use data augmentation.

Methods	MNIST ($L = 100$)	SVHN ($L = 1000$)	CIFAR-10 ($L = 4000$)
Ladder network [Rasmus *et al.*, 2015]	0.89 ± 0.50	-	20.40 ± 0.47*
GoodSemiBadGan [Dai *et al.*, 2017]	0.795 ± 0.098	4.25 ± 0.03*	14.41 ± 0.03*
Π model [Laine and Aila, 2016]	-	4.82 ± 0.17	12.36 ± 0.31
Temporal ensembling [Laine and Aila, 2016]	-	4.42 ± 0.16	12.16 ± 0.24
Mean teacher [Tarvainen and Valpola, 2017]	-	3.95 ± 0.19	12.31 ± 0.28
VAT + EntMin [Miyato *et al.*, 2017]	-	3.86	10.55
$\Pi + \mathrm{SNTG}$ [Luo *et al.*, 2017]	0.66 ± 0.07	3.82 ± 0.25	11.00 ± 0.13
VAdD(KL)+VAT [Park *et al.*, 2018]	-	3.55 ± 0.05	9.22 ± 0.10
Tri-net	$\mathbf{0.53 \pm 0.10}$	3.71 ± 0.14	$\mathbf{8.45 \pm 0.22}$
Tri-net + Π model	$\mathbf{0.52 \pm 0.05}$	$\mathbf{3.45 \pm 0.10}$	$\mathbf{8.30 \pm 0.15}$

Table 2: Results of tri-net with/without output smearing. err means the error rate of ensemble of three modules M_1, M_2 and M_3. arg means the ratio of the agreed data by modules M_1, M_2 and M_3.

datasets	MNIST		SVHN		CIFAR-10	
index	err	agr	err	agr	err	agr
without output smearing	8.55 ± 0.00	$\mathbf{85.69 \pm 0.50}$	12.47 ± 0.12	82.56 ± 0.88	16.51 ± 0.09	81.47 ± 0.40
with output smearing	$\mathbf{7.85 \pm 0.48}$	86.52 ± 0.55	$\mathbf{12.20 \pm 0.21}$	$\mathbf{81.25 \pm 0.22}$	$\mathbf{15.42 \pm 0.17}$	$\mathbf{79.98 \pm 0.89}$

MNIST with 100 labeled examples and 8.45% error rate on CIFAR-10 with 4000 labeled examples, which are much better than state-of-the-art methods. Since tri-net exploits three modules while the state-of-the-art methods exploit one or two modules, the time cost of tri-net is more than that of these methods.

There is an initialization in tri-net, with more sophisticated initialization methods, tri-net could have better performance. Π model [Laine and Aila, 2016] is a rising semi-supervised deep learning method. It evaluates each input twice based on the neural network and calculates the loss between the two predictions to regularize the neural network. We also use Π model to initialize three modules M_1, M_2 and M_3 in tri-net and call it tri-net + Π model. The results are also shown in Table 1. From Table 1, we can find that tri-net + Π model performs better than tri-net and achieves the error rate of 3.45% on SVHN with 1000 labeled examples.

Tri-net is a semi-supervised learning method by using unlabeled data to improve learning performance. It has been reported that semi-supervised learning with the exploitation of unlabeled data might deteriorate learning performance [Balcan and Blum, 2010; Chapelle *et al.*, 2006]. Now, we demonstrate whether the performance of tri-net will be deteriorated by keeping on using unlabeled data. As tri-net labels more and more unlabeled data, we depict the error rates of three modules M_1, M_2, M_3 and tri-net in every learning round in Figure 3, which shows that except very few learning rounds, the performance is not deteriorated by keeping on using unlabeled data.

4.3 Further Discussion

In order to generate three accurate and diverse modules M_1, M_2 and M_3, we introduce output smearing in initialization. We record the error rates of ensemble of three modules M_1, M_2, M_3 and their agreement in the initialization with/without output smearing. The results are shown in Table 2. Table 2 indicates that on all three datasets the error rates of ensemble of M_1, M_2 and M_3 in initialization with output smearing are lower than that without output smearing. Three modules M_1, M_2 and M_3 generated with output smearing also have large diversity (low agreement means large diversity). As tri-net goes on, M_1, M_2 and M_3 become similar, and then fine-tuning is introduced to augment the diversity among them. Some pseudo-labels may be incorrect, pseudo-label editing is used to alleviate the influence of suspicious pseudo-labels.

To show that whether these techniques are helpful to tri-net, we run experiments with/without them in tri-net, and the results are shown in Figure 4. Figure 4 indicates that when all three techniques are used, tri-net has the best performance. It implies that these techniques are very necessary for tri-net and each of them makes a contribution to the good performance of tri-net.

Table 3: Error rates (%) of tri-net with the same/different structures.

datasets	MNIST	SVHN	CIFAR-10
with the same structure	0.60	3.95	9.05
with different structures	**0.53**	**3.71**	**8.45**

Different network structures are used to get three diverse modules M_1, M_2 and M_3, we conduct the experiments with the same network structure for three modules M_1, M_2 and M_3 as a comparison. The results shown in Table 3 indicate that different structures bring better performance. The parameter σ_{os} controls the confidence threshold when output smearing is used in the training process. We conduct the experiments with different $\sigma_{os} \in [0.01, 0.25]$, and the results

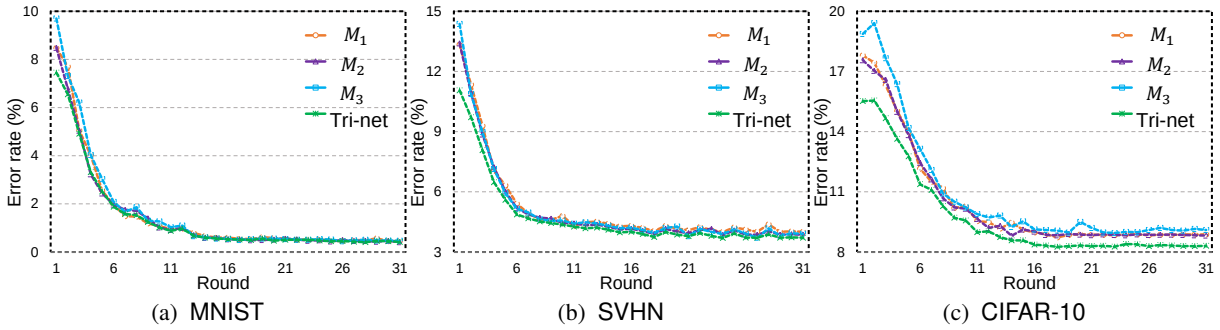

Figure 3: Error rates of tri-net and its three modules M_1, M_2 and M_3.

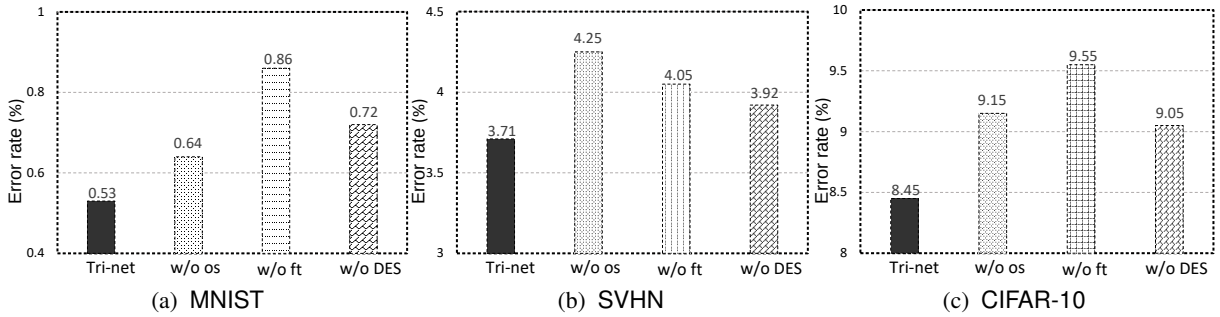

Figure 4: Error rates of tri-net with/without three techniques. Specifically, "w/o os" means tri-net without output smearing, "w/o ft" means tri-net without fine-tuning, and "w/o DES" means tri-net without pseudo-label editing.

shown in Table 4 indicate that tri-net is not very sensitive to the parameter σ_{os}.

Table 4: Error rates (%) of tri-net with different σ_{os}.

σ_{os}	0.01	0.05	0.1	0.25
MNIST	**0.53**	0.55	0.58	0.60
SVHN	4.23	4.09	3.81	**3.71**
CIFAR-10	9.38	9.10	8.65	**8.45**

5 Conclusion

In this paper, we propose tri-net for semi-supervised deep learning, in which we generate three modules to exploit unlabeled data by considering model initialization, diversity augmentation and pseudo-label editing simultaneously. Experiments on several benchmarks demonstrate that our method is superior to state-of-the-art semi-supervised deep learning methods. In particular, it can achieve the error rate of 8.30% on CIFAR-10 by using only 4000 labeled examples. Extending tri-net with more modules could exploit the power of ensemble in labeling the unlabeled data confidently. In this situation, one important issue is to maintain the diversity among these modules, which will be an interesting research direction in semi-supervised deep learning.

References

[Abbasnejad *et al.*, 2017] E. Abbasnejad, A. Dick, and A. v. d. Hengel. Infinite variational autoencoder for semi-supervised learning. In *CVPR*, pages 781–790, 2017.

[Ardehaly and Culotta, 2017] E. M. Ardehaly and A. Culotta. Co-training for demographic classification using deep learning from label proportions. In *ICDM Workshop*, pages 1017–1024, 2017.

[Bachman *et al.*, 2014] P. Bachman, O. Alsharif, and D. Precup. Learning with pseudo-ensembles. In *NIPS*, pages 3365–3373, 2014.

[Balcan and Blum, 2010] M. F. Balcan and A. Blum. A discriminative model for semi-supervised learning. *Journal of the ACM*, 57(3):19:1–19:46, 2010.

[Balcan *et al.*, 2004] M. F. Balcan, A. Blum, and K. Yang. Co-training and expansion: Towards bridging theory and practice. In *NIPS*, pages 89–96, 2004.

[Blum and Mitchell, 1998] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *COLT*, pages 92–100, 1998.

[Breiman, 2000] L. Breiman. Randomizing outputs to increase prediction accuracy. *Machine Learning*, 40(3):229–242, 2000.

[Chapelle *et al.*, 2006] O. Chapelle, B. Schölkopf, and A. Zien. *Semi-supervised learning*. MIT Press, 2006.

[Cheng *et al.*, 2016] Y. Cheng, X. Zhao, R. Cai, Z. Li, K. Huang, and Y. Rui. Semi-supervised multimodal deep learning for RGB-D object recognition. In *IJCAI*, pages 3345–3351, 2016.

[Dai *et al.*, 2017] Z. Dai, Z. Yang, F. Yang, W. W. Cohen, and R. Salakhutdinov. Good semi-supervised learning that requires a bad GAN. In *NIPS*, pages 6513–6523, 2017.

[Girshick *et al.*, 2014] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014.

[Goldman and Zhou, 2000] S. A. Goldman and Y. Zhou. Enhancing supervised learning with unlabeled data. In *ICML*, pages 327–334, 2000.

[Goodfellow *et al.*, 2014] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572, 2014.

[Ioffe and Szegedy, 2015] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456, 2015.

[Kingma *et al.*, 2014] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling. Semi-supervised learning with deep generative models. In *NIPS*, pages 3581–3589, 2014.

[Krizhevsky *et al.*, 2012] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1106–1114, 2012.

[Laine and Aila, 2016] S. Laine and T. Aila. Temporal ensembling for semi-supervised learning. *CoRR*, abs/1610.02242, 2016.

[Luo *et al.*, 2017] Y. Luo, J. Zhu, M. Li, Y. Ren, and B. Zhang. Smooth neighbors on teacher graphs for semi-supervised learning. *CoRR*, abs/1711.00258, 2017.

[Maaløe *et al.*, 2016] L. Maaløe, C. K. Sønderby, S. K. Sønderby, and O. Winther. Auxiliary deep generative models. In *ICML*, pages 1445–1453, 2016.

[Miyato *et al.*, 2017] T. Miyato, S. Maeda, M. Koyama, and S. Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *CoRR*, abs/1704.03976, 2017.

[Park *et al.*, 2018] S. Park, J. K. Park, S. J. Shin, and I. C. Moon. Adversarial dropout for supervised and semi-supervised learning. In *AAAI*, 2018.

[Rasmus *et al.*, 2015] A. Rasmus, M. Berglund, M. Honkala, H. Valpola, and T. Raiko. Semi-supervised learning with ladder networks. In *NIPS*, pages 3546–3554, 2015.

[Saito *et al.*, 2017] K. Saito, Y. Ushiku, and T. Harada. Asymmetric tri-training for unsupervised domain adaptation. In *ICML*, pages 2988–2997, 2017.

[Sajjadi *et al.*, 2016] M. Sajjadi, M. Javanmardi, and T. Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *NIPS*, pages 1163–1171, 2016.

[Salimans *et al.*, 2016] T. Salimans, I. J. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen. Improved techniques for training GANs. In *NIPS*, pages 2226–2234, 2016.

[Shelhamer *et al.*, 2017] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):640–651, 2017.

[Srivastava *et al.*, 2014] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[Szegedy *et al.*, 2016] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, pages 2818–2826, 2016.

[Tarvainen and Valpola, 2017] A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NIPS*, pages 1195–1204, 2017.

[Wang and Zhou, 2010] W. Wang and Z.-H. Zhou. A new analysis of co-training. In *ICML*, pages 1135–1142, 2010.

[Wang and Zhou, 2017] W. Wang and Z.-H. Zhou. Theoretical foundation of co-training and disagreement-based algorithms. *CoRR*, abs/1708.04403, 2017.

[Weston *et al.*, 2012] J. Weston, F. Ratle, H. Mobahi, and R. Collobert. Deep learning via semi-supervised embedding. In *Neural Networks: Tricks of the Trade - Second Edition*, pages 639–655. 2012.

[Zhang and Zhou, 2011] M.-L. Zhang and Z.-H. Zhou. CoTrade: Confident co-training with data editing. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 41(6):1612–1626, 2011.

[Zhou and Li, 2005a] Z.-H. Zhou and M. Li. Semi-supervised regression with co-training. In *IJCAI*, pages 908–916, 2005.

[Zhou and Li, 2005b] Z.-H. Zhou and M. Li. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Transactions on Knowledge and Data Engineering*, 17(11):1529–1541, 2005.

[Zhou and Li, 2010] Z.-H. Zhou and M. Li. Semi-supervised learning by disagreement. *Knowledge and Information Systems*, 24(3):415–439, 2010.

[Zhou, 2012] Z.-H. Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC, 2012.

[Zhu, 2007] X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, University of Wisconsin-Madison, 2007.