EnAET: Self-Trained Ensemble AutoEncoding Transformations for Semi-Supervised Learning

Xiao Wang*[1,2], Daisuke Kihara[1,5], Jiebo Luo[2,4], and Guo-Jun Qi†[2,3]

[1]Department of Computer Science, Purdue University
[2]Futurewei Technologies
[3]Laboratory for MAchine Perception and LEarning (MAPLE)
[4]University of Rochester
[5]Department of Biological Science, Purdue University

Abstract

Deep neural networks have been successfully applied to many real-world applications. However, these successes rely heavily on large amounts of labeled data, which is expensive to obtain. Recently, Auto-Encoding Transformation (AET) and MixMatch have been proposed and achieved state-of-the-art results for unsupervised and semi-supervised learning, respectively. In this study, we train an Ensemble of Auto-Encoding Transformations (EnAET) to learn from both labeled and unlabeled data based on the embedded representations by decoding both spatial and non-spatial transformations. This distinguishes EnAET from conventional semi-supervised methods that focus on improving prediction consistency and confidence by different models on both unlabeled and labeled examples. In contrast, we propose to explore the role of **self-supervised** *representations in* **semi-supervised** *learning under a rich family of transformations. Experiment results on CIFAR-10, CIFAR-100, SVHN and STL10 demonstrate that the proposed EnAET outperforms the state-of-the-art semi-supervised methods by significant margins. In particular, we apply the proposed method to extremely challenging scenarios with only 10 images per class, and show that EnAET can achieve an error rate of* 9.35% *on CIFAR-10 and* 16.92% *on SVHN. In addition, EnAET achieves the best result when compared with fully supervised learning using all labeled data with the same network architecture. The performance on CIFAR-10, CIFAR-100 and SVHN with a smaller network is even more competitive than the state-of-the-art of supervised learning methods based on a larger network. We also set a new performance record with an error rate of* 1.99% *on CIFAR-10 and* 4.52% *on STL10. The code and experiment records are released at https://github.com/maple-research-lab/EnAET.*

1. Introduction

Deep neural network has shown its sweeping successes in learning from large-scale labeled datasets like ImageNet [10]. However, such successes hinge on the availability of a large amount of labeled examples that are expensive to collect. Moreover, deep neural networks usually have a large number of parameters that are prone to over-fitting. Thus, we hope that semi-supervised learning can not only deal with the limited labels but also alleviate the over-fitting problem by exploring unlabeled data. In this paper, we successfully prove that both goals can be achieved by training a semi-supervised model built upon self-supervised representations.

Semi-Supervised Learning (SSL) [6, 18] has been extensively studied due to its great potential for addressing the challenge with limited labels. Most state-of-the-art approaches can be divided into two categories. One is confident predictions [18, 30, 26], which improves a model's confidence by encouraging low entropy prediction on unlabeled data. The other category imposes consistency regularization [7, 25, 40, 45] by minimizing discrepancy among the predictions by different models. The two approaches employ reasonable objectives since good models should make confident predictions that are consistent with each other.

On the other hand, a good model should also recognize an object even if it is transformed in different ways. With deep networks, this is usually achieved by training the model with augmented labeled data. However, unsuper-

*The work was done while X. Wang was interning at Futurewei Technologies.

†Corresponding author: G.-J. Qi. Email: guojunq@gmail.com

vised data augmentation is preferred to explore the effect of various transformations on unlabeled data. For this reason, we will show that self-supervised representations learned from auto-ending an ensemble of spatial and non-spatial transformations can play a key role in significantly enhancing semi-supervised models. To this end, we will present an Ensemble of Auto-Encoding Transformations (AETs) [50] to self-train semi-supervised classifiers with various transformations by combining the advantages of both existing semi-supervised approaches and the newly developed self-supervised representations.

Our contributions are summarized as follows:

- We propose the first method that employs an ensemble of both spatial and non-spatial transformations from both labeled and unlabeled data in a self-supervised fashion to train a semi-supervised network.
- We apply an ensemble of AETs to learn robust features under various transformations, and improve the consistency of the predictions on transformed images by minimizing their KL divergence.
- We demonstrate EnAET outperforms the state-of-the-art models on all benchmark datasets in both semi-supervised and fully-supervised tasks.
- We show in the ablation study that exploring an ensemble of transformations plays a key role in achieving new record performances rather than simply applying AET as a regularizer.

The remainder of the paper is organized as follows. We briefly review the related work in semi-supervised learning and self-supervised learning in Section 2. We present our algorithm EnAET in Section 3. Further details of the EnAET framework are provided in Section 4. To prove our method's effectiveness, extensive experiments related to supervised learning and semi-supervised learning are described in Section 5. Finally, we conclude in Section 6.

2. Related Work

In this section, we review both semi-supervised and self-supervised learning approaches in the literature.

2.1. Semi-Supervised Learning

A wide variety of Semi-Supervised Learning (SSL) methods have been developed in literature. For example, Teach-Student Models [38, 25, 45, 30] constitute a large category of SSL, which is closely related to the proposed model. Inspired by unsupervised learning [36], supervision information is incorporated into the variational auto-encoders to learn various semi-supervised classifiers [23, 31, 29, 43]. GAN-based models [37, 41] also show promising results on many semi-supervised tasks. Below we review two ideas about semi-supervised learning that are closely related to the proposed model.

Consistency Predictions One of the most popular and successful ideas in SSL is to encourage consistent predictions when inputs and/or models are perturbed. Inspired by Denoising Source Separation (DSS) [42], Γ-model [38] applies a denoising layer to improve its performance by minimizing the impact of potential perturbations. Π-model [25] is further improved by adding stochastic augmentations on images and dropout [44] on network neurons to maximize prediction consistency. Furthermore, Virtual Adversarial Training (VAT) [30] uses adversarial perturbations to replace the random noises in Π-model, making it more resilient against noises.

Compared with the previous methods on maximizing the prediction consistency under perturbations, the Mean Teacher model [45] updates the weights of a teacher model with an exponential moving average (EMA) of the weights from a sequence of student models as follows.

$$\Theta'_\tau = \alpha\Theta'_{\tau-1} + (1-\alpha)\Theta_\tau \tag{1}$$

where Θ (Θ') is the weights of the Student (Teacher) Model, τ denotes the update step and α is a smoothing coefficient, which is always set to 0.999. This can result in stable and accurate predictions, and will be integrated into the proposed EnAET.

Confident Predictions The other successful idea in SSL is to encourage a model to make confident predictions on both labeled and unlabeled data. For the feature space of a model, it is ideal that each class has a clear boundary with other classes. In other words, the boundary should be far away from the high density regions of data. "Pseudo-Label" [18] implements this idea by minimizing the entropy loss of the predictions on unlabeled data. VAT [30] also combines this entropy minimization term to make confident predictions. Similarly, several other works [26, 4] encourage confident predictions by constructing hard labels for high-confident unlabeled data to "sharpen" the predictions.

Recently, MixUp [49] was proposed to further improve the boundary between classes together with entropy minimization. Instead of only focusing on predicted results on given data, MixUp trains a model with the linear combination of the inputs and corresponding outputs. This has shown extraordinary performances in both supervised [49] and semi-supervised tasks [4, 46].

While we borrow the idea of these semi-supervised approaches in developing the proposed semi-supervised model, we find that self-supervised representations play a more vital role in exploring the data variations under a variety of transformations. Unlike data augmentation applied to labeled data, we can self-train a semi-supervised model without relying on the labeled data. This can significantly boost the performances in semi-supervised as well as fully-supervised learning tasks. For this reason, we also review

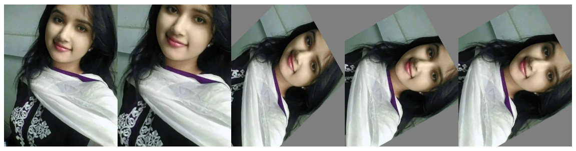
Figure 1. Spatial transformations. The images are original, projective transformation, affine transformation, similarity transformation, euclidean transformation.

the related work on self-supervised methods below.

2.2. Self-Supervised Learning

There are a wide variety of unsupervised models that apply different types of self-supervised signals to train deep networks. The self-supervising signals can be directly derived from data without manual labeling. Mehdi and Favaro [33] propose to solve Jigsaw puzzles to train a convolutional neural network. Doersch et al. [13] train the network by predicting the relative positions between sampled patches from an image as self-supervising information. Noroozi et al. [34] count the features that satisfy equivalence relations between downsampled and tiled images, while Gidaris et al. [16] classify a discrete set of image rotations to train deep networks. Dosovitskiy et al. [14] create a set of surrogate classes from individual images. Unsupervised features have also been learned from videos by estimating the self-motion of moving objects between consecutive frames [1].

More recently, the AutoEncoding Transformations (AET) model [50] has demonstrated the state-of-the-art performances in many unsupervised tasks. It aims to learn a good representation of visual structures that can decode the transformations from the learned representations of original and transformed images. We will adopt this self-supervised model to develop a self-trained model for semi-supervised tasks by exploring unlabeled data under a transformation ensemble.

3. Ensemble AutoEncoding Transformations

In this section, we will introduce Ensemble AET (EnAET), a novel idea of leveraging an ensemble of spatial and non-spatial transformations to self-train semi-supervised models.

Indeed, the difference between the features extracted from original and transformed images is caused by the applied transformations. Therefore, the transformation decoder can recover the transformations so long as the encoded features capture the necessary details of visual structures. AutoEncoding Transformation (AET) [50] can self-train a good feature representation upon which a competitive semi-supervised classifier can be developed to explore an ensemble of spatial and non-spatial transformations.

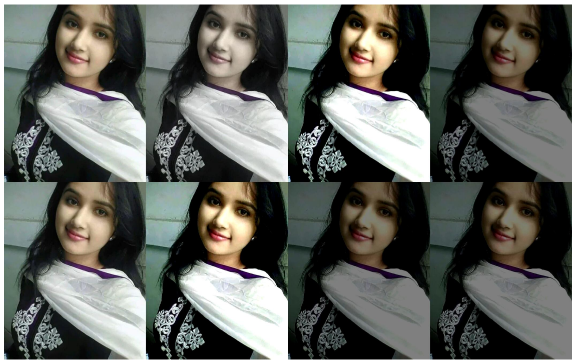
Figure 2. Non-spatial transformations. The images are original, color transformation, contrast transformation, brightness transformation, sharpen transformation, color+contrast, color+contrast+brightness, color+contrast+brightness+sharpen.

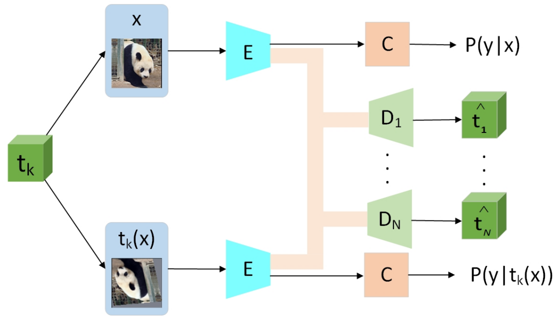

Figure 3. The Ensemble Auto Encoding Transformation Pipeline.

Fig. 3 illustrates the framework of the EnAET model. We will discuss the details in Section 3.2.

3.1. Ensemble Transformations

AutoEncoding Transformation (AET) [50] extracts the most representative features so that a transformation decoder can successfully recover parameterized transformations. In the SSL setting, instead of pretraining the model with the AET loss, we formulate AET as a regularizer along with the SSL loss to train classifiers.

Specifically, we can minimize a linear combination of the SSL loss, the AET loss to train a classifier over the network weights Θ

$$\min_{\Theta} \mathcal{L}_{SSL} + \sum_{k=1}^{N} \lambda_k \mathcal{L}_{AET_k}$$

where λ_k is the weight on the AET loss $\mathcal{L}_{AET_k}$ for the transformation t_k of the kth type.

Here, the SSL loss $\mathcal{L}_{SSL}$ can be any loss used to train a semi-supervised classifier in the literature. Particularly, we will use the MixMatch loss [4] that gives the state-of-the-art SSL result, and the consistency loss that we proposed in Section 4.2.

Table 1. Details of Geometry based Transformations.

Name	DOF	Matrix	Effect	Property
Projective	8	$\begin{bmatrix} a_1 & a_2 & b_1 \\ ,a_3 & a_4 & b_2 \\ c_1 & c_2 & 1 \end{bmatrix}$	Translation+Rotation +Scale+Aspect Ratio + Shear+Projective	Lines map to lines Parallelism may not be maintained Defined on the complement of line
Affine	6	$\begin{bmatrix} a_1 & a_2 & b_1 \\ a_3 & a_4 & b_2 \\ 0 & 0 & 1 \end{bmatrix}$	Translation+Rotation +Scale+Aspect Ratio + Shear	Preserves collinearity&parallelism Preserves the ratio of distances Does not preserve angles or lengths
Similarity	4	$\begin{bmatrix} a*cos(\theta) & -sin(\theta) & b_1 \\ sin(\theta) & a*cos(\theta) & b_2 \\ 0 & 0 & 1 \end{bmatrix}$	Translation+Rotation +Scale	Preserves collinearity&parallelism Preserves general shape of objects Preserves angles of objects
Euclidean	3	$\begin{bmatrix} cos(\theta) & -sin(\theta) & b_1 \\ sin(\theta) & cos(\theta) & b_2 \\ 0 & 0 & 1 \end{bmatrix}$	Translation+Rotation	Preserves collinearity&parallelism Preserves exact shape of objects Preserves angles&lengths of objects

Table 2. Non-spatial Transformations.

Transformation	Description
Color	Adjusts color balance of image
Contrast	Adjusts difference of light pixels and dark pixels in image
Brightness	Adds or subtracts to image matrix to change brightness
Sharpness	Adjusts pixels to make image appear sharper

The AET loss can be written as

$$\mathcal{L}_{AET_k} = \mathbb{E}_{x,t_k} \left\| D\left[E(x), E(t_k(x))\right] - t_k \right\|^2 \qquad (2)$$

where D denotes the transformation decoder, E represents the encoder, and t_k is the sampled transformation of type k. The AET loss computes the Mean-Squared Error (MSE) between the predicted transformation and the sampled transformation.

We will show that the self-trained AET regularization can help EnAET set a new record in all SSL tasks under an ensemble of spatial and non-spatial transformations.

Spatial Transformations As in [20], for any 2D spatial transformation, we can represent it with a matrix below in Eq. (3),

$$\begin{bmatrix} a_1 & a_2 & b_1 \\ a_3 & a_4 & b_2 \\ c_1 & c_2 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \qquad (3)$$

where $\left(\begin{smallmatrix} a_1 & a_2 \\ a_3 & a_4 \end{smallmatrix}\right)$ is a submatrix that controls the rotation, aspect ratio, shearing and scaling factors, $\left(\begin{smallmatrix} b_1 \\ b_2 \end{smallmatrix}\right)$ is the translation, and $\left(\begin{smallmatrix} c_1 & c_2 \end{smallmatrix}\right)$ is the projection; (x, y) is the coordinate of original image, while (x', y') denotes the coordinate after the transformation.

Based on Eq. (3), we incorporate four most representative transformations into the AET loss: 1) Projective transformation, 2) Affine transformation, 3) Similarity transformation, and 4) Euclidean transformation. We illustrate and compare these transformations in Table 1 and Fig. 1. Although it seems that projective transformation includes the other transformations as its special cases, it is natural to raise the question of whether we still need to include the other transformations explicitly. However, noting the fact that the Euclidean, Similarity and Affine transformations rarely happen under Projective transformation when we use the random parameters to conduct projective transformation, we will explicitly sample these transformations in EnAET. Our ablation study also shows that can improve the model's performance.

Non-spatial transformations A good classifier can also recognize objects in different color, contrast, brightness, and sharpness conditions. Therefore, we also add these non-spatial transformations to the AET network. We consider four different non-spatial transformations as shown in Table 2. For simplicity, these four transformations are applied as an entire non-spatial transformation with four strength parameters in EnAET. The effect of such a combined Color Contrast Brightness Sharpness (CCBS) transformation is illustrated in Fig. 2.

3.2. Architecture

We illustrate the architecture of the proposed EnAET in Fig. 3. For each image x, we apply five different transformations: t_1(Projective), t_2(Affine), t_3(Similarity), t_4(Euclidean), t_5(CCBS).

After that, the network is split into three parts: an representation encoder E, a classifier C, and a set of decoders D_k each for a type of transformation t_k. The original input x and all its transformed counterparts $t_k(x)$ are fed through the network. The original and transformed images have a Siamese encoder E and classifier C with shared weights.

For example, Wide ResNet-28-2 consists of four blocks,

Algorithm 1 Training Ensemble AutoEncoding Transformations.

```
Input: a batch of labeled data pair $\mathcal{X}$, unlabeled data $\mathcal{U}$, the number of transformations in EnAET $N$, the balancing coefficients $\lambda_{\mathcal{U}'}$, $\lambda_k$, and $\gamma$.
1: $\mathcal{X}'$,$\mathcal{U}' = MixMatch(\mathcal{X}$,$\mathcal{U})$    ▷ Use Algorithm 1 in MixMatch [4] to get MixUp dataset
2: $\mathcal{L}_{mix} = \mathcal{L}_{\mathcal{X}'} + \lambda_{\mathcal{U}'} * \mathcal{L}_{\mathcal{U}'}$    ▷ Calculate mix loss based on labeled and unlabeled data (see Eq.(4))
3: for $k$=1 to $N$ do
4:     $\mathcal{L}_{AET_k} = \mathbb{E}_{x \in \mathcal{U}, t_k} \|D[E(x), E(t_k(x))] - t_k\|^2$    ▷ Calculate AET loss, which is illustrated in Section 3.1.
5:     $\mathcal{L}_{KL_k} = \mathbb{E}_{x \in \mathcal{U}, t_k} \sum_y P(y|x) \log \frac{P(y|x)}{P_{t_k}(y|x)}$    ▷ Encourage consistent predictions, details in Section 4.2
6: end for
7: $\mathcal{L} = \mathcal{L}_{mix} + \sum_{k=1}^{N} \lambda_k \mathcal{L}_{AET_k} + \gamma \sum_{k=1}^{N} \mathcal{L}_{KL_k}$    ▷ Calculate the overall loss
8: Apply $\mathcal{L}$ to update model.
9: Update teacher model: $\Theta'_\tau = \alpha\Theta'_{\tau-1} + (1-\alpha)\Theta_\tau$    ▷ Use EMA [45] to update the final model's loss (see Eq.(1))
Output: Student model with weight $\Theta$ and teacher model with weight $\Theta'$.
```

and we use the last block as the classifier C, while the other three blocks constitute the encoder E. Also, all the decoders D_k's share the same network architecture as the classifier C but with different weights.

The representations of the original and transformed images will be concatenated to predict the parameters of each transformation t_k by the corresponding decoder D_k.

The classifier C is built upon the encoded representation to output the label predictions $P(y|x)$ and $P(y|t(x))$ for both the original and transformed images, respectively. The label prediction of original image needs to be "sharpened" [17] to reach a high degree of prediction confidence by minimizing the prediction entropy.

4. More Algorithm Details

In this section, we provide more details to implement EnAET for the SSL tasks. The full EnAET framework is provided in Algorithm 1.

4.1. MixMatch

MixMatch [4] is the current state-of-the-art for the semi-supervised classification task. It uses many ideas in previous approaches to build a new framework. Among them are Mixup [49] by training a network with a convex combination of examples and their labels, as well as entropy minimization by sharpening the label predictions. In addition, a guessed label with data augmentation also contributes to more consistent label predictions.

Formally, we first apply MixMatch of Algorithm 1 in [4] to mix up a batch of labeled $\mathcal{X}$ and unlabeled $\mathcal{U}$, resulting in the mixed-up $\mathcal{X}'$,$\mathcal{U}'$. Then, we minimize the following SSL loss $\mathcal{L}_{mix}$ to train the model along with the AET loss and consistency loss,

$$\begin{cases} \mathcal{L}_{\mathcal{X}'} = \mathbb{E}_{(x,y)\in\mathcal{X}'} H(y, f(x,\Theta)) \\ \mathcal{L}_{\mathcal{U}'} = \mathbb{E}_{(u,q)\in\mathcal{U}'} ||f(u,\Theta) - q||^2 \\ \mathcal{L}_{mix} = \mathcal{L}_{\mathcal{X}'} + \lambda_{\mathcal{U}'}\mathcal{L}_{\mathcal{U}'} \end{cases} \tag{4}$$

where $\mathcal{X}'$ and $\mathcal{U}'$ are the labeled and unlabeled data resulting from MixMatch, H is the cross-entropy between two distributions, and q is the sharpened predictions on a unlabeled sample u for each $(u,q) \in \mathcal{U}'$.

4.2. Consistent Predictions

We also consider another idea for the SSL, transformation invariance by making consistent predictions on image labels under transformations [39, 30].

To achieve this, we minimize the KL divergence between the "guessed label" $P(y|x)$ [4] on an original image x and $P_t(y|x) \triangleq P(y|t(x))$ on a transformed image $t(x)$

$$\mathcal{L}_{KL} = \mathbb{E}_{x,t} \sum_y P(y|x) \log \frac{P(y|x)}{P_t(y|x)} \tag{5}$$

to make consistent predictions under different transformations, where the expectation is taken over the sampled data and transformations.

Furthermore, we also sharpen the output predictions [17] to make them more confident for an unlabeled image x. This fulfills the entropy minimization mechanism that works well in many SSL tasks and has been elaborately discussed in VAT [30].

4.3. Data Augmentation

In SSL, a training set contains both labeled and unlabeled examples. Thus, we consider two different types of transformations for data augmentations in EnAET:

(1) For MixMatch [49] part (Algorithm 1, line 1), we apply three augmentations: Random flip and crop (Basic operation for benchmarks), and mosaic mask inspired by Cutout [12] to compute the corresponding SSL loss. For mosaic mask, we use the average pixel value of the masked area to fill the mask.

(2) As mentioned in the Section 3, for the AET loss, we use the spatial and non-spatial transformations to augment the data.

It is worth noting that we will not use these AET transformations in MixMatch since the transformations can introduce undesired distortions that could change the image "guessed labels", and that the augmented data in MixMatch are used to train the label classifier.

5. Experiments

To evaluate the performance of the proposed method, we conduct experiments on multiple SSL benchmark datasets in Section 5.3. We also demonstrate the performance for fully supervised learning in Section 5.4. An ablation study is included in Section 5.5 to analyze the contribution of EnAET.

5.1. Training Details

For a fair comparison with other SSL methods, we use "Wide ResNet-28-2" as our backbone network and a "Wide ResNet-28-2" architecture with 135 filters per layer as our larger model (same in [4]), which we will refer to as "Wide ResNet-28-2-Large" in the following. This follows the evaluation setup for the baseline methods in [35].

We simply use the Adam solver [23] with a learning rate of 0.002 to train the backbone network E and C. The SGD optimizer is used to train the AET regularization network D_k with an initial learning rate of 0.1 as in [50]. Then we use a cosine [28] scheduler for a learning rate decay from 0.1 to 0.0001. We also fix the weight decay rate to 5e-4.

For all experiments, we use a batch size of 128 images. The model is trained for 1024 epochs. For the sake of fair comparison, the mean accuracy of the last 20 models is reported as in [4]. Also, we report the error variance based on 4 runs with different random seeds.

We also adopt the same hyper-parameter setup to minimize the MixMatch loss as in [4]. The weight λ_k of the AET loss is initialized to $1.0, 0.75, 0.5, 0.2, 0.05$ for the four spatial transformations and the CCBS transformation, respectively, and we fix that for different datasets. The weight γ of the KL divergence loss is also set to 0.2 as an initial value, and we found it would not influence the overall performance too much, which only needs to be slightly adjusted if you want to transfer to more datasets. Like MixMatch [4], we use a warm-up strategy for the weights of these losses.

Finally, we also use the exponential moving average (EMA) [45] in the experiments with a rate of 0.999.

5.2. Transformation Details

For spatial transformations, considering they can all be expressed by a matrix shown in Eq. (3), we use the same operation settings for all of them. For random rotation, we set the rotation degree from [$-180°$,$180°$]. For the translation factor, we randomly sample the translation from [-0.2,0.2] for both horizontal and vertical directions. Also, we sample the scaling factor from [0.8,1.2] to make the scaled image fall in a proper range. With the shearing factor, we limit the shearing in [$-30°$,$30°$] to make sure the image can still be recognized. For the projective factor, we set the translation factor for the 4 corners of an image in [-0.125,0.125] for both horizontal and vertical directions.

For non-spatial transformations, we randomly sample the magnitude for color, contrast, brightness and sharpness from [0.2,1.8] to keep the transformed images recognizable by human beings.

5.3. Semi-Supervised Learning

To evaluate the proposed method, we perform semi-supervised tasks on four datasets: CIFAR-10 and CIFAR-100 [24], SVHN [32], and STL-10 [8].

5.3.1 CIFAR-10 Results

For CIFAR-10, we evaluate the compared methods with different sizes of labeled data. The results are reported in Table 3. Experiments are conducted 4 times with different random seeds to test the stability of our method and we find out that our method's variance is very small when we only use 250 labels. Therefore, we report the variance based on 4 runs for 250 labels and do not further perform more experiments for other conditions.

The results show that the proposed method outperforms all the state-of-the-art methods. For example, the proposed model achieves a 7.6% error rate with 250 labels compared with the previous best rate of 11.08%. Notably, we are the first to conduct experiments with only 50 labels and 100 labels, and achieve 16.45% and 9.35% error rates, respectively. It is worth noting that the proposed model with 50 labels even outperforms most methods with $1,000$ labels.

Meanwhile, with Wide ResNet-28-2-Large, we achieve a 4.18% error rate with $4,000$ labeled images, which is very close to the error rate of 4.0% by the fully supervised "Wide ResNet-28-10" baseline [48] that has a much larger and more complicated architecture.

5.3.2 CIFAR-100 Results

Different models are compared in Table 4 on CIFAR-100. The proposed EnAET achieves an error rate of 58.73% and 31.83% with only $1,000$ and $5,000$ labels. The performance of EnAET with $5,000$ labels is even better than other models with $10,000$ labels.

In addition, with Wide ResNet-28-2-Large, we achieve a 22.92% error rate compared with the previous best error rate of 25.88% (see Table 6 for more details) with $10,000$ labels.

Table 3. Error rates of different models on CIFAR-10.

Methods/Labels	50	100	250	500	1000	2000	4000
Π-Model [25, 40]	–	–	53.02±2.05	41.82±1.52	31.53±0.98	23.07±0.66	17.41±0.37
PseudoLabel [26]	–	–	49.98±1.17	40.55±1.70	30.91±1.73	21.96±0.42	16.21±0.11
MixUp [49]	–	–	47.43±0.92	36.17±2.82	25.72±0.66	18.14±1.06	13.15±0.20
VAT [30]	–	–	36.03±2.82	26.11±1.52	18.68±0.40	14.40±0.15	11.05±0.31
MeanTeacher [45]	–	–	47.32±4.71	42.01±5.86	17.32±4.00	12.17±0.22	10.36±0.25
MixMatch [4]	–	–	11.08±0.87	9.65±0.94	7.75±0.32	7.03±0.15	6.24±0.06
EnAET	**16.45**	**9.35**	**7.6±0.34**	**7.27**	**6.95**	**6.00**	**5.35**

Table 4. Error rates of different models on CIFAR-100.

Methods/Labels	1000	5000	10000
Supervised-only	–	–	51.21±0.33
Π-Model [40]	–	–	39.19±0.36
Temporal ensembling [25]	–	–	38.65±0.51
EnAET	**58.73**	**31.83**	**26.93±0.21**

Table 5. Error rates of different models on STL10.

Methods/Labels	1000	5000
CutOut [12]	-	12.74
IIC [22]	-	11.20
SWWAE [51]	25.70	-
CC-GAN^2 [11]	22.20	-
MixMatch [4]	10.18±1.46	5.59
EnAET	**8.04**	**4.52**

Table 6. Comparison of error rates with Wide ResNet-28-2-Large.

Methods/Labels	CIFAR-10 4000 label	CIFAR-100 10000 label	SVHN 1000 label
Mean Teacher [45]	6.28	–	–
SWA [2]	5.00	28.80	–
Fast SWA [3]	5.0	28.0	–
MixMatch [4]	4.95±0.08	25.88±0.30	–
EnAET	**4.18**	**22.92**	**2.42**

5.3.3 SVHN Results

Compared with the previous methods, the proposed model achieves a new performance record on the SVHN dataset as shown in Table 7. Notably, we are the first to test SVHN under 100 images and achieved 16.92% error rate, which is even better compared to some methods with $1,000$ labels.

In addition, we test with $1,000$ labels by using a Wide ResNet-28-2-Large backbone and achieve a further reduction in the error rate from 2.92% to 2.42%.

5.3.4 STL10 Results

STL-10 contains $5,000$ labeled images and $100,000$ unlabeled images. Here we use the "Wide ResNet-28-2" architecture with one more block (same as [4]) as the backbone to build the model. We achieve the best performance with $1,000$ and $5,000$ labeled images in Table 5. We reduce the error rate from 10.18% to 8.04% with $1,000$ labeled images, and set a record error rate of 4.52% when using all labeled data.

5.4. Supervised Learning

We also compare different models in the fully supervised setting on CIFAR-10, CIFAR-100, SVHN, and STL-10 by using all labeled data in the training set. We show that the proposed model can still achieve the best results with the same network architecture.

5.4.1 Wide ResNet-28-2 Backbones

We compare different methods by using Wide ResNet-28-2 as the backbone in Table 8.

CIFAR-10 Result Using all the labels in CIFAR-10, we achieve a 3.55% error rate with the 1.47M-parameter architecture. Even compared with AutoAugment [9], which is based on reinforcement learning and spends $5,000$ hours to find optimal augmentation policies, we still perform better with a 0.55% decline in the error rate.

CIFAR-100 Result EnAET can achieve a 20.55% error rate on CIFAR-100. This improves the baseline Wide ResNet-28-2 with a reduction of 4.05% in the error rate.

SVHN Result For SVHN, the proposed method achieves a 2.48% error rate compared with 3.3% by the baseline.

5.4.2 Wide ResNet-28-2-Large Backbones

Here, we consider an even larger Wide ResNet-28-2-Large backbone by increasing the number of filters per layer to 135, resulting in a network with 26M parameters. While Wide ResNet-28-10 has 36M parameters and a more complicated architecture, we regard its performance as the baseline to compare with the proposed model. We also compare the proposed model with the current state-of-the-art methods to demonstrate its remarkable performance in Table 9. All the settings are exactly the same as in the semi-supervised experiment.

CIFAR-10 Result Compared with the Wide ResNet-28-10 baseline, we achieve the best 1.99% error rate. Even compared with the state-of-the-art fully supervised Proxy-

Table 7. Error rates of different models on SVHN.

Methods/Labels	100	250	500	1000	2000	4000
Π-Model [25, 40]	–	17.65±0.27	11.44±0.39	8.6±0.18	6.94±0.27	5.57±0.14
PseudoLabel [26]	–	21.16±0.88	14.35±0.37	10.19±0.41	7.54±0.27	5.71±0.07
MixUp [49]	–	39.97±1.89	29.62±1.54	16.79±0.63	10.47±0.48	7.96±0.14
VAT [30]	–	8.41±1.01	7.44±0.79	5.98±0.21	4.85±0.23	4.20±0.15
MeanTeacher [45]	–	6.45±2.43	3.82±0.17	3.75±0.10	3.51±0.09	3.39±0.11
MixMatch [4]	–	3.78±0.26	3.64±0.46	3.27±0.31	3.04±0.13	2.89±0.06
EnAET	**16.92**	**3.21±0.21**	**3.05**	**2.92**	**2.84**	**2.69**

Table 8. Error Rates of fully supervised models with a Wide ResNet-28-2 backbone.

Methods/Labels	CIFAR-10	CIFAR-100	SVHN
Baseline [48]	5.1	24.6	3.3
AutoAugment [9]	4.1	21.5	1.7
MixMatch [4]	4.13	–	2.59
EnAET	**3.55**	**20.55**	**2.48**

Table 9. Error Rates of fully supervised models with Wide ResNet-28-2-Large.

Methods/Labels	CIFAR-10	CIFAR-100	SVHN
Baseline [48]	3.9	18.8	3.1
AutoAugment [9]	2.6	17.1	**1.9**
PBA [21]	2.6	16.7	-
Fast AA [27]	2.7	17.3	-
EnAET	**1.99**	16.87	2.22
ProxylessNAS [5]	2.08	-	-
CutMix [47]	2.88	**13.81**	-

Table 10. Ablation study of EnAET on CIFAR-10.

Ablation	250labels
EnAET	**7.64**
Only Projective Transformation	9.96
Only Affine Transformation	8.12
Only Similarity Transformation	10.25
Only Euclidean Transformation	10.56
Only CCBS Transformation	15.34
Remove EMA	19.79
Remove KL divergence	9.88
Remove AET regularization	13.54
Remove mosaic augmentation	8.23
Remove KL and mosaic	8.18
Remove AET and mosaic	13.79

lessNAS based on the time-consuming network architecture search, we achieve better results with mean accuracy.

CIFAR-100 Result Compared with the baseline Wide ResNet-28-10, we further decrease the error rate from 18.8% to 16.87%. Although it is slightly worse than PBA [21], the latter relied on a time-consuming search for the best transformation policies on an extra validation set. The state-of-the-art method CutMix [47] has a 13.81% error rate, but it is based on a much more complicated PyramidNet-200 architecture [19].

SVHN Result Compared with the baseline 3.1% error rate, we further reduce it to 2.22%.

The results show that EnAET produces comparable performances with those methods based on network architecture search and augmentation policy search. It also greatly improves the baseline network that has more model parameters. We expect the proposed method should achieve even better results when more complicated network structures such as PyramidNet [19] and more powerful regularization such as ShakeShake [15] are used.

5.5. Ablation Study

We perform an ablation study of the proposed EnAET on CIFAR-10 and results are shown in Table 10. Specifically, we focus on the roles of different transformations and the different components in EnAET:

1) With four different spatial transformations (see Table 10), we can see that Affine transformation contributes most to the performance. The results are reasonable since Affine transformation contains both Similarity and Euclidean transformations. Also, Projective transformation performs slightly worse than Affine transformation, although the former contains the latter as a special case. This suggests a need to search for a proper range of transformations. This is left to our future work, where we will use policy search to find the best range for different transformations. For non-spatial transformation here, it can't contribute to improvement when it is used alone.

2) We also find that the AET loss plays an indispensable role in achieving the superior performances. As shown in Table 10, without the AET regularization, the error rate jumps to 13.54%.

3) Mosaic data augmentation (in Section 4.3) in MixUp [49] and KL divergence (in Section 4.2) also contributes to the final performance, but it cannot work alone without the AET regularization.

6. Conclusion

In this paper, we present EnAET, a novel framework that integrates the idea of self-trained representations into semi-supervised models. Throughout our experiments, the proposed method outperforms the state-of-the-art methods on all the datasets by a significant margin in the performance. Meanwhile, with the same architecture, the proposed method can also greatly improve the fully-supervised baseline. Furthermore, for many datasets, EnAET even performs better than NAS, the auto-augmentation policy search. The results even show that comparable performances with the state-of-the-art methods can be achieved with a less complicated backbone with fewer model parameters. In the future, we will employ policy search to find the best combination of transformations and their ranges, which we believe can lead to even more competitive results.

References

[1] Pulkit Agrawal, Joao Carreira, and Jitendra Malik. Learning to see by moving. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 37–45, 2015.

[2] Ben Athiwaratkun, Marc Finzi, Pavel Izmailov, and Andrew Gordon Wilson. Improving consistency-based semi-supervised learning with weight averaging. *arXiv preprint arXiv:1806.05594*, 2, 2018.

[3] Ben Athiwaratkun, Marc Finzi, Pavel Izmailov, and Andrew Gordon Wilson. There are many consistent explanations of unlabeled data: Why you should average. *arXiv preprint arXiv:1806.05594*, 2018.

[4] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. Mixmatch: A holistic approach to semi-supervised learning. *arXiv preprint arXiv:1905.02249*, 2019.

[5] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. *arXiv preprint arXiv:1812.00332*, 2018.

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

[7] Dan Claudiu Cirecsan, Ueli Meier, Luca Maria Gambardella, and Jürgen Schmidhuber. Deep, big, simple neural nets for handwritten digit recognition. *Neural computation*, 22(12):3207–3220, 2010.

[8] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 215–223, 2011.

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

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

[11] Emily Denton, Sam Gross, and Rob Fergus. Semi-supervised learning with context-conditional generative adversarial networks. *arXiv preprint arXiv:1611.06430*, 2016.

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

[13] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1422–1430, 2015.

[14] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *Advances in neural information processing systems*, pages 766–774, 2014.

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

[16] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728*, 2018.

[17] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

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

[19] Dongyoon Han, Jiwhan Kim, and Junmo Kim. Deep pyramidal residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5927–5935, 2017.

[20] Richard I Hartley. Theory and practice of projective rectification. *International Journal of Computer Vision*, 35(2):115–127, 1999.

[21] Daniel Ho, Eric Liang, Ion Stoica, Pieter Abbeel, and Xi Chen. Population based augmentation: Efficient learning of augmentation policy schedules. *arXiv preprint arXiv:1905.05393*, 2019.

[22] Xu Ji, João F Henriques, and Andrea Vedaldi. Invariant information distillation for unsupervised image segmentation and clustering. *arXiv preprint arXiv:1807.06653*, 2018.

[23] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pages 3581–3589, 2014.

[24] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

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

[26] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, volume 3, page 2, 2013.

[27] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast autoaugment. *arXiv preprint arXiv:1905.00397*, 2019.

[28] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[29] Lars Maaloe, Casper Kaae Sonderby, Soren Kaae Sonderby, and Ole Winther. Auxiliary deep generative models. *arXiv preprint arXiv:1602.05473*, 2016.

[30] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993, 2018.

[31] Siddharth Narayanaswamy, T Brooks Paige, Jan-Willem Van de Meent, Alban Desmaison, Noah Goodman, Pushmeet Kohli, Frank Wood, and Philip Torr. Learning disentangled representations with semi-supervised deep generative models. In *Advances in Neural Information Processing Systems*, pages 5925–5935, 2017.

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

[33] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016.

[34] Mehdi Noroozi, Hamed Pirsiavash, and Paolo Favaro. Representation learning by learning to count. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5898–5906, 2017.

[35] Avital Oliver, Augustus Odena, Colin A Raffel, Ekin Dogus Cubuk, and Ian Goodfellow. Realistic evaluation of deep semi-supervised learning algorithms. In *Advances in Neural Information Processing Systems*, pages 3235–3246, 2018.

[36] Guo-Jun Qi and Jiebo Luo. Small data challenges in big data era: A survey of recent progress on unsupervised and semi-supervised methods. *arXiv preprint arXiv:1903.11260*, 2019.

[37] Guo-Jun Qi, Liheng Zhang, Hao Hu, Marzieh Edraki, Jingdong Wang, and Xian-Sheng Hua. Global versus localized generative adversarial nets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1517–1525, 2018.

[38] Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. In *Advances in neural information processing systems*, pages 3546–3554, 2015.

[39] Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, pages 833–840. Omnipress, 2011.

[40] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *Advances in Neural Information Processing Systems*, pages 1163–1171, 2016.

[41] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. In *Advances in neural information processing systems*, pages 2234–2242, 2016.

[42] Jaakko Särelä and Harri Valpola. Denoising source separation. *Journal of machine learning research*, 6(Mar):233–272, 2005.

[43] Casper Kaae Sonderby, Tapani Raiko, Lars Maaloe, Soren Kaae Sonderby, and Ole Winther. Ladder variational autoencoders. In *Advances in neural information processing systems*, pages 3738–3746, 2016.

[44] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

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

[46] Vikas Verma, Alex Lamb, Juho Kannala, Yoshua Bengio, and David Lopez-Paz. Interpolation consistency training for semi-supervised learning. *arXiv preprint arXiv:1903.03825*, 2019.

[47] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. *arXiv preprint arXiv:1905.04899*, 2019.

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

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

[50] Liheng Zhang, Guo-Jun Qi, Liqiang Wang, and Jiebo Luo. Aet vs. aed: Unsupervised representation learning by auto-encoding transformations rather than data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2547–2555, 2019.

[51] Junbo Zhao, Michael Mathieu, Ross Goroshin, and Yann Lecun. Stacked what-where auto-encoders. *arXiv preprint arXiv:1506.02351*, 2015.