TOWARDS RESISTING LARGE DATA VARIATIONS VIA INTROSPECTIVE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning deep networks which can resist large variations between training and testing data is essential to build accurate and robust image classifiers. Towards this end, a typical strategy is to apply data augmentation to enlarge the training set. However, standard data augmentation is essentially a brute-force strategy which is inefficient, as it performs all the pre-defined transformations to every training sample. In this paper, we propose a principled approach to train networks with significantly improved resistance to large variations between training and testing data. This is achieved by embedding a learnable transformation module into the introspective networks (Jin et al., 2017; Lazarow et al., 2017; Lee et al., 2018), which is a convolutional neural network (CNN) classifier empowered with generative capabilities. Our approach alternatively synthesizes pseudo-negative samples with learned transformations and enhances the classifier by retraining it with synthesized samples. Experimental results verify that our approach significantly improves the ability of deep networks to resist large variations between training and testing data and achieves classification accuracy improvements on several benchmark datasets, including MNIST, affNIST, SVHN and CIFAR-10.

1 INTRODUCTION

Classification problems have rapidly progressed with advancements in convolutional neural networks (CNNs) (LeCun et al., 1989; Krizhevsky et al., 2012; Simonyan & Zisserman, 2014; Szegedy et al., 2015; He et al., 2016; Huang et al., 2017). CNNs are able to produce promising performance, given sufficient training data. However, when the training data is limited and unable to cover all the data variations in the testing data (e.g., the training set is MNIST, while the testing set is affNIST), the trained networks generalize poorly on the testing data. Consequently, how to learn deep networks which can resist large variations between training and testing data is a significant challenge for building accurate and robust image classifiers.

To address this issue, a typical strategy is to apply data augmentation to enlarging the training set, i.e., applying various transformations, including random translations, rotations and flips as well as Gaussian noise injection, to the existing training data. This strategy is very effective in improving the performance, but it is essentially a brute-force strategy which is inefficient, as it exhaustively performs all these transformations to every training samples. Neither is it theoretically formulated.

Alternatively, we realize that we can synthesize extra training samples with generative models. But, the problem is how to generate synthetic samples which are able to improve the robustness of CNNs to large variations between training and testing data. In this paper, we achieve this by embedding a learnable transformation module into *introspective networks* (Jin et al., 2017; Lazarow et al., 2017), a CNN classifier empowered with generative capabilities. We name our approach *introspective transformation network* (ITN), which performs training by a reclassification-by-synthesis algorithm. It alternatively synthesizes samples with learned transformations and enhances the classifier by retraining it with synthesized samples. We use a min-max formulation to learn our ITN, where the transformation module transforms the synthesized pseudo-negative samples to maximize their variations to the original training samples and the CNN classifier is updated by minimizing the classification loss of the transformed synthesized pseudo-negative samples. The transformation modules are learned jointly with the CNN classifier, which augments training data in an intelligent manner by narrowing down the search space for the variations.

Our approach can work with any models that have generative and discriminative abilities, such as generative adversarial networks (GANs) and introspective networks. In this paper, we choose the introspective networks to generate extra training samples rather than GANs, because introspective networks have several advantages over GANs. Introspective learning framework maintains one single CNN discriminator that itself is also a generator while GANs have separate discriminators and generators. The generative and discriminative models are simultaneously refined over iterations. Additionally, Introspective networks are easier to train than GANs with gradient descent algorithms by avoiding adversarial learning.

The main contribution of the paper is that we propose a principled approach that endows classifiers with the ability to resist larger variations between training and testing data in an intelligent and efficient manner. Experimental results show that our approach achieves better performance than standard data augmentation on both classification and cross-dataset generalization. Furthermore, we also show that our approach has great abilities in resisting different types of variations between training and testing data.

2 Related Work

In recent years, a significant number of works have emerged focus on resisting large variations between training and testing data. The most widely adopted approach is data augmentation that applies pre-defined transformations to the training data. Nevertheless, this method lacks efficiency and stability since the users have to predict the types of transformations and manually applies them to the training set. Better methods have been proposed by building connections between generative models and discriminative classifiers (Friedman et al., 2001; Liang & Jordan, 2008; Tu et al., 2008; Jebara, 2012; Welling et al., 2003). This type of methods capture the underlying generation process of the entire dataset. The discrepancy between training and test data is reduced by generating more samples from the data distribution.

GANs (Goodfellow et al., 2014) have led a huge wave in exploring the generative adversarial structures. Combining this structure with deep CNNs can produce models that have stronger generative abilities. In GANs, generators and discriminators are trained simultaneously. Generators try to generate fake images that fool the discriminators, while discriminators try to distinguish the real and fake images. Many variations of GANs have emerged in the past three years, like DCGAN (Radford et al., 2015), WGAN (Arjovsky et al., 2017) and WGAN-GP (Gulrajani et al., 2017). These GANs variations show stronger learning ability that enables generating complex images. Techniques have been proposed to improve adversarial learning for image generation (Salimans et al., 2016; Gulrajani et al., 2017; Denton et al., 2015) as well as for training better image generative models (Radford et al., 2015; Isola et al., 2017).

Introspective networks (Tu, 2007; Lazarow et al., 2017; Jin et al., 2017; Lee et al., 2018) provide an alternative approach to generate samples. Introspective networks are closely related to GANs since they both have generative and discriminative abilities but different in various ways. Introspective networks maintain one single model that is both discriminative and generative at the same time while GANs have distinct generators and discriminators. Introspective networks focus on introspective learning that synthesizes samples from its own classifier. On the other hand, GANs emphasize adversarial learning that guides generators with separate discriminators. The generators in GANs are mappings from the features to the images. However, Introspective networks directly models the underlying statistics of an image with an efficient sampling/inference process.

3 Method

We now describe the details of our approach in this section. We first briefly review the introspective learning framework proposed by (Tu, 2007). This is followed by our the detailed mathematical explanation of our approach. In particular, we focus on explaining how our model generates unseen examples that complement the training dataset.

3.1 Introspective Learning

We only briefly review introspective learning for binary-class problems, since the same idea can be easily extended to multi-class problems. Let us denote $x \in \mathbb{R}^d$ as a data sample and $y \in +1, -1$

as the corresponding label of x. The goal of introspective learning is to model positive samples by learning the generative model $p(x|y=+1)$. Under Bayes rule, we have

$$p(x|y=+1) = \frac{p(y=+1|x)p(y=-1)}{p(y=-1|x)p(y=+1)} p(x|y=-1), \tag{1}$$

where $p(y|x)$ is a discriminative model. For pedagogical simplicity, we assume $p(y=1) = p(y=-1)$ and this equation can be further simplified as:

$$p(x|y=+1) = \frac{p(y=+1|x)}{p(y=-1|x)} p(x|y=-1). \tag{2}$$

The above equation suggests that a generative model for the positives $p(x|y=+1)$ can be obtained from the discriminative model $p(y|x)$ and a generative model $p(x|y=-1)$ for the negatives. However, to faithfully learn $p(x|y=+1)$, we need to have a representative $p(x|y=-1)$, which is very difficult to obtain. A solution was provided in (Tu, 2007) which learns $p(x|y=-1)$ by using an iterative process starting from an initial reference distribution of the negatives $p_0(x|y=-1)$, e.g., $p_0(x|y=-1) = U(x)$, a Gaussian distribution on the entire space $\mathbb{R}^d$. This is updated by

$$p_{t+1}(x|y=-1) = \frac{1}{Z_t} \frac{q_t(y=+1|x)}{q_t(y=-1|x)} p_t(x|y=-1), \tag{3}$$

where $q_t(y|x)$ is a discriminative model learned on a given set of positives and a limited number of pseudo-negatives sampled from $p_t(x|y=-1)$ and $Z_t = \int \frac{q_t(y=+1|x)}{q_t(y=-1|x)} p_t(x|y=-1)dx$ is the normalizing factor. It has been proven that $KL(p(x|y=+1)||p_{t+1}(x|y=-1)) \leq KL(p(x|y=+1)||p_t(x|y=-1)))$ (as long as each $q_t(y|x)$ makes a better-than-random prediction, the inequality holds) in (Tu, 2007), where $KL(\cdot||\cdot)$ denotes the Kullback-Leibler divergences, which implies $p_t(x|y=-1) \overset{t=\infty}{\rightarrow} p(x|y=+1)$. Therefore, gradually learning $p_t(x|y=-1)$ by following this iterative process of Eqn.(3), the samples drawn from $x \sim p_t(x|y=-1)$ become indistinguishable from the given training samples.

3.2 LARGE VARIATIONS RESISTANCE VIA INTROSPECTIVE LEARNING

Introspective Convolutional Networks (ICN) (Jin et al., 2017) and Wasserstein Introspective Neural Networks (WINN) (Lee et al., 2018) adopt the introspective learning framework and strengthen the classifiers by a reclassification-by-synthesis algorithm. However, both of them fail to capture large data variations between the training and testing data, since most of the generated pseudo-negatives are very similar to the original samples. But in practice, it is very common that the test data contain unseen variations that are not in training data, such as the same objects viewed from different angles and suffered from shape deformation.

To address this issue, we present our approach building upon the introspective learning framework to resist large data variations between training and test data. Arguably, even large training sets cannot fully contains all the possible variations. Our goal is to quickly generate extra training samples with beneficial unseen variations that is not covered by the training data to help classifiers become robust. We assume that we can generates such training samples by applying a transformation function $\mathcal{T}(\cdot\,;\sigma)$ parametrized by learnable parameters σ to the original training samples. Let us denote $g(\cdot\,;\psi)$ as the function that maps the samples x to the transformation parameters σ, where ψ is the model parameter of the function g. The generated samples still belong to the same category of the original samples, since the transformation function $\mathcal{T}$ only changes the high-level geometric properties of the samples. The outline of training procedures of ITN is presented in Algorithm 1. We denote $S^+ = \{(x_i^+, +1), i = 1...|S^+|\}$ as the positive sample set, $\mathcal{T}_t(S^+) = \{(x_i^T, +1), i = 1...|S^+|, x_i^T = \mathcal{T}(x_i^+; \sigma_t)\}$ as the transformed positive sample set at t^{th} iteration with transformation parameter σ_t and $S_t^- = \{(x_i^-, -1), i = 1...|S^-|\}$ as the set of pseudo-negatives drawn from $p_t(x|y=-1)$. We then will describe the detail of the training procedure.

Discriminative model We first demonstrate the approach of building robust classifiers with given σ_t. For a binary classification problem, at t^{th} iteration, the discriminative model is represented as

$$q_t(y|x;\theta_t) = \frac{1}{1+\exp(-y\ f_t(x;\theta_t))} \tag{4}$$

Algorithm 1: Outline of ITN Training Algorithm

Input: Positive sample set S^+, initial reference distribution $p_0(x|y=-1)$ and transformation function $\mathcal{T}$
Output: Parameters θ, ω and ψ
Build S_0^- by sampling $|S^+|$ pseudo-negatives samples from $p_0(x|y=-1)$
initialize parameters θ, ω and ψ, set t = 1
while not converge **do**
  for each $x_i^+ \in S^+$ and $x_i^- \in S_t^-$ **do**
    Compute transformation parameters $\sigma_i = g(x_i^+; \psi)$
    Choose $\epsilon_i \sim U(0,1)$ and compute $\hat{x}_i = \epsilon_i \mathcal{T}(x_i^+; \sigma_i) + (1-\epsilon_i)x_i^-$
  end for
  Compute θ, ω by Eqn.(6)
  Compute ψ by Eqn.(8)
  Sample pseudo-negatives samples $Z_t = \{z_i^t, i = 1, ..., |S^+|\}$ from $p_0(x|y=-1)$
  Update all samples in Z_t by Eqn.(12)
  Augment pseudo-negatives sample set $S_t^- = S_{t-1}^- \cup \{(z_i^t, -1), i = 1, ..., |S^+|\}$ and t = t + 1
end while

where θ_t represents the model parameters at iteration t, and $f_t(x;\theta_t)$ represents the model output at t^{th} iteration. Note that, $q_t(y|x;\theta_t)$ is trained on S^+, $\mathcal{T}(S^+;\sigma_t)$ and pseudo-negatives drawn from $p_t(x|y=-1)$. In order to achieve stronger ability in resisting unseen variations, we want the distribution of $\mathcal{T}(S^+;\sigma_t)$ to be approximated by the distribution of pseudo negatives $p_t(x|y=-1)$, which can be achieved by minimizing the following Wasserstein distance (Gulrajani et al., 2017):

$$D(\theta_t, \omega_t) = \mathbb{E}_{x^T \sim \mathcal{T}(S^+;\sigma_t)}[f_t(x^T;\theta_t,\omega_t)] - \mathbb{E}_{x^- \sim S_t^-}[f_t(x^-;\theta_t,\omega_t)] + \lambda \mathbb{E}_{\hat{x}\sim\hat{X}_t}[\|\nabla_{\hat{x}} f_t(\hat{x};\theta_t,\omega_t) - 1\|_2^2], \tag{5}$$

where ω_t is the extra parameter together with $f_t(\cdot;\theta_t)$ to compute the Wasserstein distance. Each $\hat{x}$ in the set $\hat{X}_t$ is computed with the formula $\hat{x} = \epsilon x^T + (1-\epsilon)x^-$, where ϵ samples from uniform distribution $U(0,1)$, $x^T \in \mathcal{T}(S^+;\sigma_t)$ and $x^- \in S_t^-$. The term $\lambda(||\nabla_{\hat{x}} f_t(\hat{x};\theta_t)||_2 - 1)^2$ is the gradient penalty that stabilizes the training procedure of the Wasserstein loss function.

The goal of the discriminative model is to correctly classify any given x^+, x^T and x^-. Thus, the objective function of learning the discriminative model at iteration t is

$$\min_{\theta_t,\omega_t} J(\theta_t) + D(\theta_t,\omega_t), \quad \text{where} \quad J(\theta_t) = \mathbb{E}_{(x,y)\sim S^+ \cup S_t^- \cup \mathcal{T}(S^+;\sigma_t)}[-\log q(y|x;\theta_t)] \tag{6}$$

The classifiers obtain the strong ability in resisting unseen variations by training on the extra samples while preserving the ability to correctly classify the original samples. We discussed the binary classification case above. When dealing with multi-class classification problems, it is needed to adapt the above reclassification-by-synthesis scheme to the multi-class case. We can directly follow the strategies proposed in (Jin et al., 2017) to extend ITN to deal with multi-class problems by learning a series of one-vs-all classifiers or a single CNN classifier.

Exploring variations. The previous section describes how to learn the robust classifiers when the σ_t is given. However, σ_t is unknown and there are huge number of possibilities to selecting σ_t. Now, the problem becomes how do we learn the σ_t in a principled manner and apply it towards building robust classifiers? We solve this issue by forming a min-max problem upon the Eqn.(6):

$$\min_{\theta,\omega} \max_{\sigma} J(\theta,\sigma) + D(\theta,\omega,\sigma), \tag{7}$$

Here, we rewrite $J(\theta)$ and $D(\theta,\omega)$ in Eqn.(5) and Eqn.(6) as $J(\theta,\sigma)$ and $D(\theta,\omega,\sigma)$, since σ is now an unknown variable. We also subsequently drop the subscript t for notational simplicity. This formulation gives us a unified perspective that encompasses some prior work on building robust classifiers. The inner maximization part aims to find the transformation parameter σ that achieves the high loss values. On the other hand, the goal of the outer minimization is expected to find the the model parameters θ that enables discriminators to correctly classify x^T and ω allows the negative distribution to well approximate the distribution of $\mathcal{T}(S^+;\sigma_t)$. However, direclty solving Eqn. 7 is difficult. Thus, we break this learning process and first find a σ^* that satisfies

$$\max_{\sigma} \mathbb{E}_{(x^T,y)\sim\mathcal{T}(S^+;\sigma)}[-\log(q(y|x^T))] + \mathbb{E}_{x^T\sim\mathcal{T}(S^+;\sigma)}[f(x^T;\theta,\omega)] + \lambda \mathbb{E}_{\hat{x}\sim\hat{X}}[\|\nabla_{\hat{x}} f(\hat{x};\theta,\omega) - 1\|_2^2] \tag{8}$$

where θ and ω are fixed. Then, θ and ω are learned with Eqn.(6) by keep $\sigma = \sigma^*$. Empirically, the first term in the Eqn. 8 dominates over other terms, therefore we can drop the second and third terms to focus on learning more robust classifiers. The purpose of empirical approximation is to find the σ^* that make x^T hard to classify correctly. Instead of enumerating all possible examples in the data augmentation, Eqn.(8) efficiently and precisely finds a proper σ that increase the robustness of the current classifiers.

We use $g(\cdot\,;\psi)$ to learn σ, thus $\sigma = g(x;\psi)+\zeta$, where ζ is random noise follows the standard normal distribution. The function parameter ψ is learned by Eqn.(8). Notably, following the standard backpropagation procedure, we need to compute the derivative of the transformation function $\mathcal{T}$ in each step. In other words, the transformation function $\mathcal{T}(\cdot;\sigma)$ need to be differentiable with respect to the parameter ψ to allow the gradients to flow through the transformation function $\mathcal{T}$ when learning by backpropagation.

Generative model In the discriminative models, the updated discriminative model $p(y|x)$ is learned by Eqn.(6). The updated discriminative model is then used to compute the generative model by the Eqn.(3) in section 3.1. The generative is learned by maximizing the likelihood function $p(x)$. However, directly learning the generative model is cumbersome since we only need samples from the latest generative model.

Let's denote initial reference distribution as $p_0^-(x)$ and $p_n(x|y=-1)$ as $p_n^-(x)$ for simplicity. Following standard introspective learning, we can approximate samples drawn from latest negative distribution by first sampling from $p_0^-(x)$ and iteratively update them to approach desired samples. With p_0^- and Eqn.(3), we have

$$p_n^-(x) = (\prod_{t=1}^{n-1} \frac{1}{Z_t} \frac{q_t(y=+1|x)}{q_t(y=-1|x)}) p_0^-(x), \tag{9}$$

where Z_t indicates the normalizing factor at t^{th} iteration. The random samples x are updated by increasing maximize the log likelihood of $p_n^-(x)$. Note that maximizing $\log p_n^-(x)$ can be simplified as maximizing $\prod_{t=1}^{n-1} \frac{q_t(y=+1|x)}{q_t(y=-1|x)}$ since Z_t and p_0^- are fixed in Eqn.(9). From this observation, we can directly learn a model $h_t(x)$ such that

$$h_t(x) = \frac{q_t(y=+1|x)}{q_t(y=-1|x)} = \exp(f_t(x;\theta_t)) \tag{10}$$

Taking natural logarithm on both side of the equation above, we can get $\ln h_t(x) = f_t(x;\theta_t)$. Therefore, $\log p_n^-(x)$ can be rewritten as

$$\log p_n^-(x) = \log(\prod_{t=1}^{n-1} \frac{1}{Z_t} \frac{q_t(y=+1|x)}{q_t(y=-1|x)}) p_0^-(x) = C \sum_{t=1}^{n-1} f_t(x;\theta_t)\; p_0^-(x), \tag{11}$$

where C is the constant computed with normalizing factors Z_t. This conversion allows us to maximize $\log p_n^-(x)$ by maximizing $\sum_{t=1}^{n-1} f_t(x;\theta_t)$. By taking the derivative of $\log p_n^-(x)$, the update step ∇x is:

$$\nabla x = \lambda \nabla (\sum_{t=1}^{n-1} f_t(x;\theta_t)) + \eta, \tag{12}$$

where $\eta \sim N(0,1)$ is the random Gaussian noise and λ is the step size that is annealed in the sampling process. In practice, we update from the samples generated from previous iterations to reduce time and memory complexity. An update threshold is introduced to guarantee the generated negative images are above certain criteria, which ensures the quality of negative samples. We modify the update threshold proposed in (Lee et al., 2018) and keep track of the $f_t(x;\theta_t)$ in every iteration. In particular, we build a set D by recording $\mathbb{E}[f_t(x;\theta_t)]$, where $x \in S^+$ in every iteration. We form a normal distribution $\mathcal{N}(a,b)$, where a and b represents mean and standard deviation computed from set D. The stop threshold is set to be a random number sampled from this normal distribution. The reason behind this threshold is to make sure the generated negative images are close to the majority of transformed positive images in the feature space.

4 EXPERIMENTS

In this section, we demonstrate the ability of our algorithm in resisting the large variations between training and testing data through a series of experiments. First, we show the outstanding classification performance of ITN on several benchmark datasets. We also analyze the properties of the

generated examples from different perspectives. We then further explore the ability of our algorithm in resisting large variations with two challenging classification tasks and show the consistently better performance. Finally, we illustrate the flexibility of our architecture in addressing different types of unseen variations.

Baselines We compare our method against CNNs, DCGAN (Radford et al., 2015), WGAN-GP (Gulrajani et al., 2017), ICN (Jin et al., 2017) and WINN (Lee et al., 2018). For generative models DCGAN and WGAN-GP, we adopt the evaluation metric proposed in (Jin et al., 2017). The training phase becomes a two-step implementation. We first generate negative samples with the original implementation. Then, the generated negative images are used to augment the original training set. We train a simple CNN that has the identical structure with our method on the augmented training set. All results reported in this section are the average of multiple repetitions.

Experiment Setup All experiments are conducted with a simple CNN architecture (Lee et al., 2018) that contains 4 convolutional layers, each having a 5×5 filter size with 64 channels and stride 2 in all layers. Each convolutional layer is followed by a batch normalization layer (Ioffe & Szegedy, 2015) and a swish activation function (Ramachandran et al., 2018). The last convolutional layer is followed by two consecutive fully connected layers to compute logits and Wasserstein distances. The training epochs are 200 for both our method and all other baselines. The optimizer used is the Adam optimizer (Kingma & Ba, 2014) with parameters $\beta_1 = 0$ and $\beta_2 = 0.9$. Our method relies on the transformation function $\mathcal{T}(\cdot)$ to convert the original samples to the unseen variations. In the following experiments, we demonstrate the ability of ITN in resisting large variations with spatial transformers (STs) (Jaderberg et al., 2015) as our transformation function unless specified. Theoretically, STs can represent all affine transformations, which endows more flexible ability in resisting unseen variations. More importantly, STs are fully differentiable, which allows the learning procedure through standard backpropagation.

4.1 CLASSIFICATION

To demonstrate the effectiveness of ITN, we first evaluate our algorithm on 4 benchmark datasets, MNIST (LeCun et al., 1998), affNIST (Tieleman, 2013), SVHN (Netzer et al., 2011) and CIFAR-10 (Krizhevsky & Hinton, 2009). The MNIST dataset includes 55000, 5000 and 10000 handwritten digits in the training, validation and testing set, respectively. The affNIST dataset is a variant from the MNIST dataset and it is built by applying various affine transformations to the samples in MNIST dataset. To accord with the MNIST dataset and for the purpose of the following experiments, we reduce the size of training, validation and testing set to 55000, 5000 and 10000, respectively. SVHN is a real-world dataset that contains house numbers images from Google Street View and CIFAR-10 contains 60000 images of ten different objects from the natural scenes. The purpose of introducing these two datasets is to further verify the performance of ITN on real-world datasets. The data augmentation we applied in the following experiments is the standard data augmentation that includes affine transformations, such as rotation, translation, scaling and shear.

	w/o DA				w/ DA			
Method	MNIST	affNIST	SVHN	CIFAR-10	MNIST	affNIST	SVHN	CIFAR-10
CNN	0.89%	2.82%	9.86%	31.31%	0.57%	1.65%	7.01%	24.35%
DCGAN	0.79%	2.78%	9.78%	31.22%	0.57%	1.63%	6.98%	24.18%
WGAN-GP	0.74%	2.76%	9.73%	31.08%	0.56%	1.56%	6.73%	24.07%
ICN	0.72%	2.97%	9.72%	32.34%	0.56%	1.54%	6.81%	24.98%
WINN	0.67%	2.56%	9.84%	30.72%	0.52%	1.48%	6.44%	23.74%
ITN	**0.49%**	**1.52%**	**6.73%**	**21.93%**	**0.47%**	**1.09%**	**5.92%**	**20.65%**

Table 1: testing errors of the classification experiments discussed in Section 4.1, where w/DA and w/o DA indicates whether data augmentation is applied.

As shown in Table 1, our method achieves the best performance on all four datasets. The overall improvements can be explained by the fact that our method generates novel and reliable negative images (shown in Figure 1) that effectively strengthen the classifiers. The images we generate are different from the previous ones, but can still be recognized as the same class. The boosted performance in value on MNIST dataset is marginal perhaps because the performance on the MNIST dataset is close to saturation. The difference between training and testing split in MNIST dataset is

	CNN	WGAN-GP	WINN	ITN
w/o DA	72.06%	70.60%	70.36%	**34.29%**
w/DA	40.74%	36.29%	33.53%	**21.31%**

Table 2: testing errors of the classification experiments described in Section 4.2.1, where w/DA and w/o DA indicates whether data augmentation is applied.

also very small compared to other datasets. Moreover, the amount of improvements increases as the dataset becomes complicated. Based on the observation of the results, we conclude that our method has stronger ability in resisting unseen variations especially when the dataset is complicated. On the other hand, we can clearly observe that our method outperforms the standard data augmentation on all datasets. This result confirms the effectiveness and the advantages of our approach. Additionally, ITN does not contradict with data augmentation since ITN shows even greater performance when integrating with data augmentation techniques. The possible reason for this observation is that the explored space between ITN and data augmentation is not overlapped. Therefore, the algorithm achieves greater performance when combining two methods together since more unseen variations are discovered in this case.

Figure 1: Images generated by our method on MNIST, affNIST, SVHN and CIFAR-10 dataset. In each sector, the top row is the original images and the bottom row is our generated images.

4.2 Quantitative Analysis

4.2.1 Cross Dataset Generalization

We have shown the substantial performance improvements of ITN against other baselines on several benchmark datasets. In this section, we want to further explore the ability of our method in resisting large variations. We design a challenging cross dataset classification task between two significantly different datasets (cross dataset generalization). The training set in this experiment is the MNIST dataset while the testing set is the affNIST dataset. The difficulty of this classification tasks is clearly how to overcome such huge data discrepancy between training and testing set since the testing set includes much more variations. Another reason why we pick these two datasets as training and testing set is that they share the same categories, which ensures the challenge is only about resisting large data variations.

As shown in Table 2, ITN has clear improvements over CNN, WGAN-GP and WINN. The amount of improvement is much larger than on the regular training and testing splits shown in Section 4.1. More importantly, our performance in this challenging task is still better than CNN with data augmentation. This encouraging result further verifies the efficiency and effectiveness of ITN compared with data augmentation. It's not surprising that data augmentation improves the performance by a significant margin since the space of unseen variations is huge. Data augmentation increases the classification performance by enumerating a large number of unseen samples, however, this brute-force searching inevitably lacks efficiency and precision.

4.2.2 Resisting Ability Under Data Paucity

Another way to evaluate the ability of resisting variations is to reduce the amount of training samples. Intuitively, the discrepancy between the training and testing sets increases when the number of samples in the training set shrinks. The purpose of this experiment is to demonstrate the potential of ITN in resisting unseen variations from a different perspective. We design the experiments where the training set is the MNIST dataset with only 0.1%, 1%, 10% and 25% of the whole training set while the testing set is the whole MNIST testing set. Each sample is randomly selected from the pool while keeps the number of samples per class same. Similarly, we repeat the same experiments on the CIFAR-10 dataset to further verify the results on a more complicated dataset. As shown in Table 3, our method has better results on all tasks. This result is consistent with Section 4.2.1 and Section 4.1, which undoubtedly illustrate the strong ability of ITN in resisting unseen variations in the testing set. The constant superior performance over data augmentation also proves the efficiency of ITN.

	w/o DA				w/ DA			
Method	CNN	WGAN-GP	WINN	ITN	CNN	WGAN-GP	WINN	ITN
0.1%(M)	36.50%	29.43%	27.18%	16.47 %	18.07%	15.35%	14.46%	**12.67%**
1%(M)	7.66%	6.86%	5.10%	3.48 %	4.48%	3.98%	3.66%	**2.82%**
10%(M)	2.02%	1.63%	1.49%	0.98 %	1.24%	1.18%	1.00%	**0.92%**
25%(M)	1.29%	1.13%	1.00%	0.78 %	0.83%	0.79%	0.77%	**0.66%**
0.1%(C)	81.99%	80.92%	78.24%	72.50 %	79.04%	78.75%	76.97%	**70.43%**
1%(C)	72.31%	71.34%	69.79%	61.48 %	65.23%	64.84%	63.26%	**58.07%**
10%(C)	59.02%	57.37%	56.02%	45.06 %	47.75%	46.86%	46.04%	**42.62%**
25%(C)	51.35%	49.01%	48.43%	34.56 %	36.50%	35.46%	34.29%	**30.60%**

Table 3: testing errors of the classification tasks described in Section 4.2.2, where M and C represents the experiments conducted on the MNIST dataset and CIFAR-10 dataset, respectively.

4.3 Beyond Spatial Transformer

Even though we utilize STs to demonstrate our ability in resisting data variations, our method actually has the ability to generalize to other types of transformations. Our algorithm can take other types of differentiable transformation functions and strengthen the discriminators in a similar manner. Moreover, our algorithm can utilize multiple types of transformations at the same time and provide even stronger ability in resisting variations. To verify this, we introduce another recently proposed work, Deep Diffeomorphic Transformer (DDT) Networks (Detlefsen et al., 2018). DDTs are similar to STs in a way that both of them can be optimized through standard backpropagation.

We replace the ST modules with the DDT modules and check whether our algorithm can resist such type of transformation. Then, we include both STs and DDTs in our model and verify the performance again. Let MNIST dataset be the training set of the experiments while the testing sets are the MNIST dataset with different types of transformation applied. We introduce two types of testing sets in this section. The first one is the normal testing set with random DDT transformation only. The second one is similar to the first one but includes both random DDT and affine transformations. The DDT transformation parameters are drawn from $N(0, 0.7 \times \mathcal{I}_d)$ as suggest in (Detlefsen et al., 2018), where $\mathcal{I}_d$ represents the d dimensional identity matrix. Then the transformed images are randomly placed in a 42×42 images. We replicate the same experiment on the CIFAR-10 dataset.

	MNIST		CIFAR-10	
	DDT	DDT + ST	DDT	DDT + ST
CNN	17.75%	55.11%	76.14 %	78.01 %
WGAN-GP	17.53%	53.24%	75.93%	77.02 %
WINN	17.20%	52.43%	75.43 %	76.92 %
ITN(DDT)	12.85 %	40.60%	53.62%	63.56 %
ITN(DDT + ST)	**9.41%**	**34.37%**	**45.26%**	**56.95 %**

Table 4: testing errors of cross dataset classification experiments, where CNN (w/ DA) represents the CNNs with data augmentation.

We can make some interesting observations from the Table 4. First, ITN can integrate with flexibly with DDT or DDT + ST to resist the corresponding variations. Second, ITN can resist partial unseen variations out of a mixture of transformations in the testing data. More importantly, the performance of ITN won't degrade when the model has transformation functions that doesn't match the type of variations in the testing data, e.g. ITN(DDT + ST) on testing data with DDT only. This observation allows us to apply multiple transformation functions in ITN without knowing the types of variations in the testing data and still maintain good performance.

5 Conclusion

We proposed a principled and smart approach that endows the classifiers with the ability to resist larger variations between training and testing data. Our method, ITN strengthens the classifiers by generating unseen variations with various learned transformations. Experimental results show consistent performance improvements not only on the classification tasks but also on the other challenging classification tasks, such as cross dataset generalization. Moreover, ITN demonstrates its advantages in both effectiveness and efficiency over data augmentation. Our future work includes applying our approach to large scale datasets and extending it to generate samples with more types of variations.

REFERENCES

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Emily L Denton, Soumith Chintala, Rob Fergus, et al. Deep generative image models using a laplacian pyramid of adversarial networks. In *Advances in neural information processing systems*, pp. 1486–1494, 2015.

Nicki Skafte Detlefsen, Oren Freifeld, and Søren Hauberg. Deep diffeomorphic transformer networks. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, NY, USA:, 2001.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pp. 5769–5779, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Gao Huang, Zhuang Liu, Kilian Q Weinberger, and Laurens van der Maaten. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, volume 1, pp. 3, 2017.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint*, 2017.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pp. 2017–2025, 2015.

Tony Jebara. *Machine learning: discriminative and generative*, volume 755. Springer Science & Business Media, 2012.

Long Jin, Justin Lazarow, and Zhuowen Tu. Introspective classification with convolutional nets. In *Advances in Neural Information Processing Systems*, pp. 823–833, 2017.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

Justin Lazarow, Long Jin, and Zhuowen Tu. Introspective neural networks for generative modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2774–2783, 2017.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Kwonjoon Lee, Weijian Xu, Fan Fan, and Zhuowen Tu. Wasserstein introspective neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Percy Liang and Michael I Jordan. An asymptotic analysis of generative, discriminative, and pseudo-likelihood estimators. In *Proceedings of the 25th international conference on Machine learning*, pp. 584–591. ACM, 2008.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, pp. 5, 2011.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. 2018.

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*, pp. 2234–2242, 2016.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al. Going deeper with convolutions. CVPR, 2015.

Tijmen Tieleman. affnist, 2013. URL `https://www.cs.toronto.edu/~tijmen/affNIST/`.

Zhuowen Tu. Learning generative models via discriminative approaches. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pp. 1–8. IEEE, 2007.

Zhuowen Tu, Katherine L Narr, Piotr Dollár, Ivo Dinov, Paul M Thompson, and Arthur W Toga. Brain anatomical structure segmentation by hybrid discriminative/generative models. *IEEE transactions on medical imaging*, 27(4):495–508, 2008.

Max Welling, Richard S Zemel, and Geoffrey E Hinton. Self supervised boosting. In *Advances in neural information processing systems*, pp. 681–688, 2003.