Representation Learning with Smooth Autoencoder

Kongming Liang, Hong Chang(✉), Zhen Cui, Shiguang Shan, and Xilin Chen

Key Lab of Intelligent Information Processing of Chinese
Academy of Sciences (CAS), Institute of Computing Technology,
CAS, Beijing 100190, China
{kongming.liang,hong.chang,zhen.cui,shiguang.shan,
xilin.chen}@vipl.ict.ac.cn

Abstract. In this paper, we propose a novel autoencoder variant, smooth autoencoder (SmAE), to learn robust and discriminative feature representations. Different from conventional autoencoders which reconstruct each sample from its encoding, we use the encoding of each sample to reconstruct its local neighbors. In this way, the learned representations are consistent among local neighbors and robust to small variations of the inputs. When trained with supervisory information, our approach forces samples from the same class to become more compact in the vicinity of data manifolds in the new representation space, where the samples are easier to be discriminated. Experimental results verify the effectiveness of the representations learned by our approach in image classification and face recognition tasks.

1 Introduction

How to represent images and videos has been an important and fundamental problem in computer vision, as the performance of various computer vision approaches relies on the choice of feature representations. Traditional hand-crafted low level image features, such as Scale Invariant Feature Transform (SIFT), Histogram of Oriented Gradients (HOG), Local Binary Patterns (LBP), have shown their effectiveness for many specific vision problems. However, these features are sub-optimal shallow representations, which require domain knowledge and have limited generalization ability. Instead of designing features manually, a more promising approach is to learn effective feature representations automatically from vision data, through *representation learning*. For various computer vision tasks, an ideal feature representation should be *robust* for small variations, *smooth* for preserving data structures and *discriminative* for classification related tasks.

Recently, representation learning in deep learning context has aroused a great deal of interests in machine learning and computer vision community. Deep models learn multi-layer nonlinear transformations from the input data to the output representations, which is more powerful in feature extraction than hand-crafted shallow models. Moreover, deep models can progressively capture more abstract

D. Cremers et al. (Eds.): ACCV 2014, Part II, LNCS 9004, pp. 72–86, 2015.
DOI: 10.1007/978-3-319-16808-1_6

features at higher layers, corresponding to the hierarchical human vision system. Representative deep learning methods such as convolutional neural networks (CNN), deep belief networks (DBN) and stacked autoencoders (SAE) have achieved great successes in image classification [1], action recognition [2], object tracking [3], etc. Among the building blocks of these models, autoencoders directly learn a parametric feature mapping function by minimizing the reconstruction error between input and its encoding (i.e., representation). In addition, various regularization terms are proposed to improve the basic autoencoders. Sparse autoencoders penalize the hidden unit to be sparse by L1 penalty [4,5] or Kullback–Leibler divergence [6] with respect to the binomial distribution. Denoising autoencoders (DAE) [7,8] learn a representation which is robust to small random perturbations. Contractive autoencoders (CAE) [9] reduce the number of effective freedom degrees of the representation by adding an analytic contractive penalty. Both DAE and CAE are robust to small changes of the inputs among the training examples. Moreover, CAE is capable of representing nonlinear manifolds, as its output encodings contract in the directions orthogonal to the underlying data manifold. This paper focuses on parametric representation learning along the direction of autoencoder variants.

It is always preferred to preserve the manifold structure at the same time of representation learning. As proved in [10], preserving the consistence in representation of similarity between local neighbors is essential for nonlinear feature learning. Classical manifold learning methods [11–13] learn embedding coordinates instead of explicit feature mappings, thus limit their usage as feature extractors. Local sparse coding methods [14,15] are capable of revealing the manifold structure, but their iterative coding process is far from efficient. Recent works incorporate manifold regularization into deep learning models and obtain parametric feedforward encoders on nonlinear manifolds, e.g., deep learning via semi-supervised embedding [16] and CAE [9]. However, it is not natural to incorporate supervisory information into these models during layerwise pretraining, so they can seldom learn discriminative representations.

In this paper, we propose a novel parametric representation learning method, smooth autoencoder (SmAE), which possesses more advantages than previous related methods mentioned above. Specifically, our approach explicitly characterizes the similarity between input samples by minimizing the weighted reconstruction error of each sample from its local neighbors, instead of itself as previous autoencoders do. Therefore, the resultant feature mappings vary smoothly on manifolds. In addition, since SmAE constrains adjacent samples to have similar feature representations, the learned feature representations are robust to small variations as the input changes on the manifold. Moreover, SmAE can learn discriminative representations by making use of supervisory information. When trained in (semi-)supervised learning setting, SmAE can increase the within-class compactness in the learned representation space. Figure 1 illustrates the classwise contractive process of our method. Experiments on image classification and face recognition show the effectiveness of the proposed method. The good performance of our method relies on its advantages briefly summarized as follows: (1) smooth representations on data manifold, (2) robust to small variations and (3) discriminative ability due to classwise contraction.

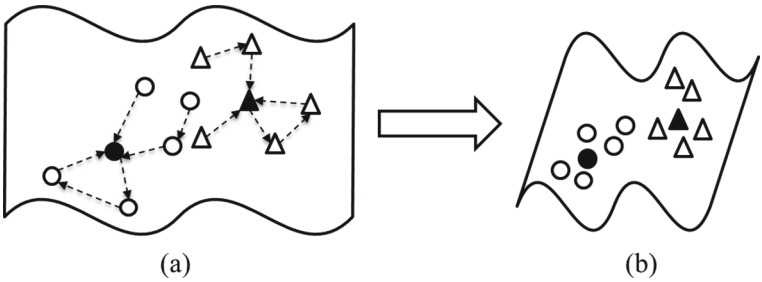

Fig. 1. Illustration of within-class contraction. The learned function maps the samples from original space (a) to feature space (b). Circles and triangles represent samples with different labels. The solid circles and triangles denote the sample correspondences between the original space and feature space. The dashed arrows between samples indicate the neighborhood relations.

The rest of this paper is organized as follows. In Sect. 2, we first review some popular autoencoder variants. Our method is then presented in Sect. 3 in detail. The relationship between SmAE and other relevant methods is discussed in Sect. 4. Experimental results on image classification and face recognition are reported in Sect. 5. Finally, we conclude this paper in Sect. 6.

2 Autoencoders and Its Variants

Autoencoders [17], as the name suggests, consist of two stages: encoding and decoding. It was first used to reduce dimensionality by setting the number of encoder output units less than the input. The model is usually trained using back-propagation in an unsupervised manner, by minimizing the reconstruction error of the decoding results from the original inputs. With the activation function chosen to be nonlinear, an autoencoder can exact more useful features than some common linear transformation methods such as PCA. If the dimension of encoding output is set higher than the input dimension, the encoding result will be enriched and more expressive. By stacking multiple pretrained autoencoders followed by a supervised classifier, the deep autoencoders will generate more abstract and high-level semantic features which are beneficial for image classification.

Basic Autoencoder (AE). The encoder is a function that maps the input data $\boldsymbol{x} \in \mathbb{R}^{d_x}$ to d_h hidden units to get the feature representation or code as:

$$\boldsymbol{h} = f(\boldsymbol{x}) = s_f(W\boldsymbol{x} + \boldsymbol{b_h}), \tag{1}$$

where s_f is a nonlinear activation function, typically a sigmoid function $s_f(z) = \frac{1}{1+e^{-z}}$, or a hyperbolic tangent function $s_f(z) = tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$. The parameters of the encoder are a $d_h \times d_x$ weight matrix W and a bias vector $\boldsymbol{b_h} \in \mathbb{R}^{d_h}$.

The decoder function g maps the outputs of hidden units back to the original input space as:

$$\boldsymbol{y} = g(\boldsymbol{h}) = s_g(W^{'}\boldsymbol{h} + \boldsymbol{b_o}), \tag{2}$$

where s_g is the activation function which usually has the same form as that in the encoder. The parameters of the decoder are a $d_x \times d_h$ weight matrix $W^{'}$ and a bias vector $\boldsymbol{b_o} \in \mathbb{R}^{d_x}$. In this paper, we choose both the encoding and decoding activation function to be sigmoid function and only consider the tied weights case, in which $W^{'} = W^T$.

To find the model parameters $\theta = \{W, \boldsymbol{b_h}, W', \boldsymbol{b_o}\}$, autoencoders are trained by minimizing the reconstruction error on a set of training data $\boldsymbol{x_i} \in \mathbb{R}^{d_x}, i = 1, ..., n$. The objective function optimized by AE is:

$$J_{AE}(\theta) = \sum_{i=1}^{n} L(\boldsymbol{x_i}, g(f(\boldsymbol{x_i}))), \tag{3}$$

where L is a loss function which is usually decided according to the input range. If the input is in [0,1], cross-entropy loss $L(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{d_x} x_i log(y_i) + (1 - x_i) log(1 - y_i)$ is usually used. In the other cases, square error $L(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|^2$ is typically chosen.

Sparse Autoencoders (SpAE). Sparse autoencoder [6] is an basic autoencoder regularized by a weight decay term and a sparsity constraints on the hidden units. The objective function of SpAE is:

$$J_{SpAE}(\theta) = \sum_{i=1}^{n} L(\boldsymbol{x_i}, g(f(\boldsymbol{x_i}))) + \lambda \sum_{ij} W_{ij}^2 + \beta \sum_{j=1}^{d_h} KL(\rho || \widetilde{\rho}_j), \tag{4}$$

where $\widetilde{\rho}_j = \frac{1}{n} \sum_{i=1}^{n} h_j(\boldsymbol{x_i})$ is the average activation of hidden unit j and the $KL(\rho||\widetilde{\rho}_j) = \rho log \frac{\rho}{\widetilde{\rho}_j} + (1 - \rho) log \frac{1-\rho}{1-\widetilde{\rho}_j}$ is the KL divergence between Bernoulli random variables with mean ρ and $\widetilde{\rho}_j$. The second term tends to decrease the magnitude of the weights and prevent over-fitting. β and λ control the tradeoff among the loss and two penalty terms.

Denoising Autoencoders (DAE). To make the representations robust to partial corruption of the input patterns, Vincent et al. [7,8] present an alternative form to train autoencoders. Instead of directly reconstructing the original input samples, denoising autoencoders learn to reconstruct the clean input $\boldsymbol{x_i}$ from the artificially corrupted counterpart $\widetilde{\boldsymbol{x}}_i$. DAE is trained by optimizing the following objective function:

$$J_{DAE}(\theta) = \sum_{i=1}^{n} \mathbb{E}_{\widetilde{\boldsymbol{x}}_i \sim q(\widetilde{\boldsymbol{x}}_i | \boldsymbol{x_i})} [L(\boldsymbol{x_i}, g(f(\widetilde{\boldsymbol{x}}_i)))]. \tag{5}$$

Typically, two common corruptions are additive isotropic Gaussian noise, $\widetilde{\boldsymbol{x}} = \boldsymbol{x} + \boldsymbol{\epsilon}, \boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 I)$, and binary masking noise, where a random fraction of inputs are set to zero.

Contractive Autoencoders (CAE). To robustness to small perturbations around the training points, [9] proposes a regularization that measures the sensitivity of the encodings with respect to the input. The contractive autoencoder(CAE) has the following objective function:

$$J_{CAE}(\theta) = \sum_{i=1}^{n} \left(L(\boldsymbol{x_i}, g(f(\boldsymbol{x_i}))) + \lambda \left\| \mathcal{J}_f(\boldsymbol{x_i}) \right\|_F^2 \right), \tag{6}$$

where $\mathcal{J}_f(\boldsymbol{x_i}) = \sum_{jk}(\frac{\partial h_j(\boldsymbol{x_i})}{\partial x_{ik}})^2$ is the Jacobian matrix which encourages the mapping to the feature space to be contractive in the neighborhood of the training data.

3 Smooth Autoencoders

In this paper, we propose a novel autoencoder variant, smooth autoencoders (SmAE), to learn nonlinear feature representations. For each input, SmAE aims to reconstruct its *target neighbors*, instead of reconstructing itself as traditional autoencoder variants do. Formally, the objective function of SmAE is defined as:

$$J_{SmAE}(\theta) = \sum_{i=1}^{n} \sum_{j=1}^{k} w(\boldsymbol{x_j}, \boldsymbol{x_i}) L(\boldsymbol{x_j}, g(f(\boldsymbol{x_i}))) + \beta \sum_{j=1}^{d_h} KL(\rho || \widetilde{\rho}_j), \tag{7}$$

where $w(\cdot, \cdot)$ is a weight function defined through a smoothing kernel $w(\boldsymbol{x_j}, \boldsymbol{x_i}) = \frac{1}{Z}\mathcal{K}(d(\boldsymbol{x_j}, \boldsymbol{x_i}))$, and the item Z is used to guarantee $\sum_{j=1}^{k} w(\boldsymbol{x_j}, \boldsymbol{x_i}) = 1$ for all i. k is the number of target neighbors of $\boldsymbol{x_i}$(see Sect. 3.1 for detail discussions). $d(\cdot, \cdot)$ is a distance function which measures the feature distance/similarity in the original space. The first term in Eq. (7) forces neighboring input samples to have similar representations. In this way, the resultant feature representations are not only robust to local variations but also smooth as the input samples vary on the manifold. The second term in the objective function regularizes on model complexity by using KL sparsity.

Besides the advantages of robustness and smoothness, SmAE can learn discriminative representations under (semi-)supervised learning settings, with proper selection of target neighbors. In the following subsections, we discuss on the choice of target neighbors and the reconstruction loss term in more details, and then describe the model training process.

There are different ways to define the weight function by using arbitrary kernel functions $\mathcal{K}(\cdot)$ and distance functions $d(\cdot, \cdot)$. Some common choices of kernel functions include Gaussian kernel, as well as triangular, uniform and tricube kernels. The distance function can be chosen as any standard distance functions based on L_p norm or learned through metric learning methods designed by domain experts. In this paper we choose Gaussian kernel which is widely used for manifold learning:

$$w(\boldsymbol{x_j}, \boldsymbol{x_i}) = \begin{cases} \frac{1}{Z} exp(-\frac{\|\boldsymbol{x_i} - \boldsymbol{x_j}\|^2}{\sigma}) & \boldsymbol{x_j} \in \mathcal{N}_i \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

$\mathcal{N}_i$ denotes the target neighborhood of sample x_i. The bandwidth of kernel σ is selected by cross-validation in our experiments. In this setting, the normalization item is $Z = \sum_{j=1}^{k} exp(-\frac{\|\boldsymbol{x_i}-\boldsymbol{x_j}\|^2}{\sigma})$.

3.1 Target Neighbors

Target neighbors can have different definitions, depending on the learning tasks and domain knowledge. Concretely speaking, we may decide the target neighbors according to unsupervised, supervised and semi-supervised learning settings, as well as the characteristics of learning tasks and training data.

Unsupervised Target Neighbors. For each x_i from the training data, we may choose k nearest neighbors based on an appropriate metric, such as Euclidian, Mahalanobis and cosine. In this paper, we use Euclidian distance to define the neighbourhood. The k nearest neighbors are considered as the k target neighbors and the corresponding distances are used to compute the weight function.

Supervised Target Neighbors. Under this setting, target neighbours are defined as k nearest neighbors with the same label of x_i. Besides the traditional global metrics, some local metrics can also be used to compute the weight function.

Beyond label information and original distance, smooth autoencoders can also utilize other forms of information, such as pairwise constraints, artificial deformation and temporal/spatial coherence to define or generate target neighbours. When we make use of supervisory information to decide the supervised target neighbors, SmAE can increase the within-class compactness in the learned representation space, as illustrated in Fig. 1. In this way, SmAE can learn discriminative representations in supervised and semi-supervised learning settings.

3.2 The Loss Function

Two typical loss functions commonly adopted in training autoencoders are squared error $L_{se}(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|^2$ and cross-entropy loss $L_{ce}(\boldsymbol{x}, \boldsymbol{y}) = -\boldsymbol{x} \cdot log(\boldsymbol{y}) - (1 - \boldsymbol{x}) \cdot log(1 - \boldsymbol{y})$, where $\cdot$ denotes the element-wise product operator. When the input data and feature representation are normalized in [0,1], the cross-entropy loss function is usually applied.

If the cross-entropy loss function is chosen, the weighted reconstruction error of SmAE for sample $\boldsymbol{x_i}$ can be simplified into the form $-\sum_{j=1}^{k} w(\boldsymbol{x_j}, \boldsymbol{x_i})\boldsymbol{x_j} \cdot log(g(f(\boldsymbol{x_i}))) - (1 - \sum_{j=1}^{k} w(\boldsymbol{x_j}, \boldsymbol{x_i})\boldsymbol{x_j}) \cdot log(1 - g(f(\boldsymbol{x_i})))$. Let us consider

$$\widetilde{\boldsymbol{x}}_i = \sum_{j=1}^{k} w(\boldsymbol{x_j}, \boldsymbol{x_i})\boldsymbol{x_j} \tag{9}$$

as a transformed version of $\boldsymbol{x_i}$. Then, the objective function of SmAE can be rewritten as:

$$J_{SmAE}(\theta) = \sum_{i=1}^{n} L_{ce}(\widetilde{\boldsymbol{x}}_i, g(f(\boldsymbol{x_i}))) + \beta \sum_{j=1}^{d_h} KL(\rho||\widetilde{\rho}_j). \tag{10}$$

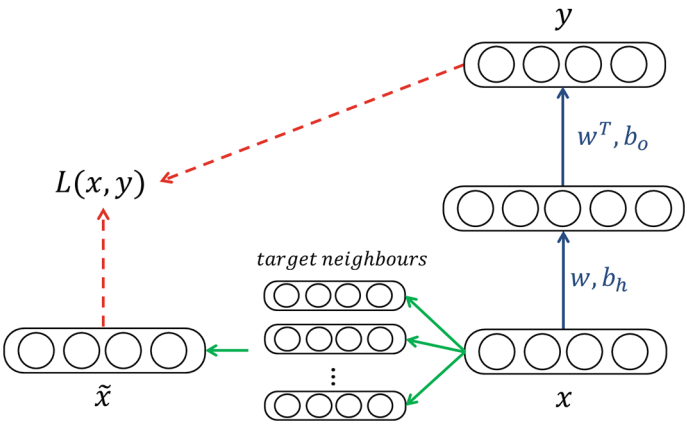

Fig. 2. Smooth autoencoder with cross-entropy loss

Therefore, SmAE can be considered as an ordinary sparse autoencoder, with new deformation samples constructed according to Eq. (9) as the reconstruction targets. Figure 2 shows the actual model architecture of SmAE with cross-entropy loss. In all experiments in this paper, cross-entropy loss is adopted as the loss function.

3.3 Model Pretraining and Stacking

Similar with previous autoencoder variants, smooth autoencoder can be used to build a deep network. For the first layer in the deep model, we choose the target neighbors defined in 3.1 and calculate the transformed version according to (9) for each training sample. The objective function (10) is minimized using standard back-propagation algorithm. The optimization procedure in layer-wise pretraining for smooth autoencoder based on cross-entropy loss function is shown in Algorithm 1. The representations learned by the first layer are then used as the input of the second layer, and so on so forth. After layer-wise pretraining, the network parameters are further fine-tuned using label information provided on top of the network in the supervised case.

4 Discussions

The smooth autoencoders possess close relationship with popular methods including sparse coding and other autoencoder variants.

4.1 Relationship with Sparse Coding

Standard sparse coding solves the following optimization problem:

$$\min_{\substack{D\in\mathbb{R}^{d_x\times K}\\ \alpha_i\in\mathbb{R}^K, i=1,\ldots,n}} \sum_{i=1}^{n}(\|\boldsymbol{x_i} - D\boldsymbol{\alpha_i}\|_2^2 + \gamma|\boldsymbol{\alpha_i}|_1), \tag{11}$$

Algorithm 1. Layer-wise pretraining of Smooth Autoencoder with Cross Entropy loss

Input: The training data set $\{x_i\}_{i=1}^n$
Output: learned weight W, bias b_h,b_o

1: Compute the transformed versions $\{\widetilde{x}_i\}_{i=1}^n$ for each training data, and initialize W, b_h, b_o
2: **while** not stopping criterion **do**
3: Set $\Delta W = 0, \Delta b_h = 0, \Delta b_o = 0$
4: Perform the feedforward pass, compute the activation of the hidden layer $z_i^h = Wx_i + b_h$ and output layer $z_i^o = W^T s_f(z_i^h) + b_o$
5: Compute the error term:

$$\delta_i^o = \frac{\partial L_{ce}((\widetilde{x}_i, g(f(x_i))))}{\partial z_i^o} = -\left(\frac{\widetilde{x}_i}{s_g(z_i^o)} + \frac{1-\widetilde{x}_i}{1-s_g(z_i^o)}\right) \cdot s_g'(z_i^o)$$

$$\delta_i^h = \left(W\delta_i^o + \beta\left(\frac{\rho}{\widehat{\rho}_i} + \frac{1-\rho}{1-\widehat{\rho}_i}\right)\right) \cdot s_f'(z_i^h)$$

6: Compute the partial derivatives:

$$\Delta W = \sum_{i=1}^n \frac{J_{SmAE}(W,b_h,b_o;x_i)}{\partial W} = \sum_{i=1}^n s_f(x_i)\delta_i^{oT} + \delta_i^h x_i^T;$$

$$\Delta b_h = \sum_{i=1}^n \frac{J_{SmAE}(W,b_h,b_o;x_i)}{\partial b_h} = \sum_{i=1}^n \delta_i^o;$$

$$\Delta b_o = \sum_{i=1}^n \frac{J_{SmAE}(W,b_h,b_o;x_i)}{\partial b_o} = \sum_{i=1}^n \delta_i^h;$$

7: Update W, b_h, b_o by gradient descent
8: **end while**

where $D = [\boldsymbol{d_1}, \boldsymbol{d_2}, ..., \boldsymbol{d_K}] \in \mathbb{R}^{d_x \times K}$ is the dictionary to be learned, with $\boldsymbol{d_i}$ being the ith atom. $\boldsymbol{\alpha_i}$ is the encoding of sample $\boldsymbol{x_i}$ with respect to dictionary D. The factor γ is used to balance the reconstruction error and sparsity penalty. An autoencoder can be viewed as sparse coding with explicit encoding function (i.e., an forward inference process) if the decoder is linear. Therefore, the direct encoding of autoencoders may be used to approximately replace the computation-expensive sparse coding by our intuition.

Some variants of sparse coding methods also attempt to use the local property from a manifold perspective. Local coordinate coding(LCC) [14] approximately represents each data point by a linear combination of its nearby anchor points from the view of reconstruction. More Recently, Smooth sparse coding(SSC) [15] is proposed to incorporate local feature similarities into the sparse coding framework. Different from them, our proposed method can learn an explicit encoding process with better robustness in small variations as well as discriminative ability due to within-class compactness. Besides, the proposed SmAE can further be easily stacked into a deep framework, which also makes the whole model optimized globally with back-propagation.

4.2 Relationship with Other Autoencoder Variants

Both smooth autoencoders and sparse autoencoders use the sparsity constraints to push the majority of representations close to zero. Sparse autoencoders encode the samples individually and ignore the mutual dependence. Therefore, small variances in the input space may result in distinct changes on the learned representations. Different from sparse autoencoders, smooth autoencoders can capture

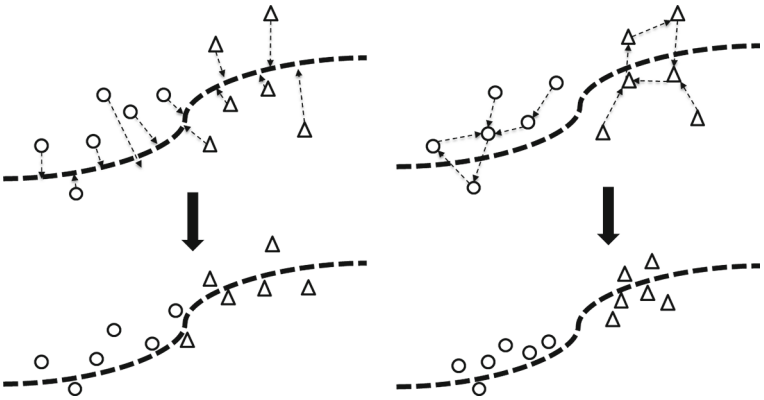

Fig. 3. Comparison between contractive autoencoder (the left) and smooth autoencoder (the right). The dashed curve refers the manifold that data rely on.

the local similarity by explicitly considering manifold structures, and thus the learned representations are less sensitive to variational inputs.

When the cross entropy loss function is adopted, the weighted reconstruction error of smooth autoencoders has similar form with denoising autoencoders, as expressed in Eq. (10). However, the robustness of denoising autoencoders is stochastic which is determined by the corruption progress. Smooth autoencoder uses transformed samples generated by target neighbors to model local variance. Therefore, the learning processing is analytic rather than stochastic, and the learned representations are more effective for classification tasks.

Smooth autoencoders are closely related to contractive autoencoders, as both of them learn robust representations along data manifolds. CAE penalizes on the Frobenius of the Jacobian matrix of the encoding function. As illustrated in Fig. 3, this penalty makes the representations contractive in the direction of noise which is orthogonal to the manifold. SmAE penalizes weighted reconstruction errors on target neighbors to guarantee robust and smooth representations. However, contractive autoencoders do have limitations: it is robust only to infinitesimal input variations and it cannot learn discriminative representations. Smooth autoencoders alleviate these limitations as they consider relaxed variations within local neighborhoods and increase within-class compactness using supervisory information.

5 Experiments

We evaluate smooth autoencoders as a representation learner in handwritten digit recognition on MNIST dataset [18] and its variations [19], and in face recognition on the Extended YaleB [20] and AR face [21] datasets. In our method, the main model parameters are chosen by cross-validating on the training set or using the available validation set.

5.1 Hand Digital Recognition

We first verify the effectiveness of SmAE on the widely used MNIST dataset, which consists of handwritten digit images with 28×28 pixels. The original MNIST dataset and MNIST variations are used in the following experiments.

Classification on MNIST. Following the standard protocol of MNIST, 60,000 images from MNIST dataset are used for training and 10,000 for testing. To intrinsically validate the effectiveness of the proposed method, here we compare several baselines, including AE, SpAE, DAE and CAE. In the proposed SmAE, the network is constrained with tied weights and the sigmoid activation function. The number of hidden nodes is set to 1000. To verify the effectiveness of preservation of neighbor structures, here we only consider unsupervised SmAE by choosing $k = 5$ target neighbors in the unsupervised way (see Sect. 3.1). The comparison results are reported in Table 1. As shown in this table, unsupervised smooth autoencoder has lower classification error than AE, SpAE and DAE, and is even comparable to the state-of-the-art method CAE. Furthermore, if we stack two SmAEs into a deep framework, the proposed method can sharply reduce the error rate to **1.06** %.

Table 1. Classification Results of different autoencoders on MNIST dataset

Method	AE	SpAE	DAE [7]	CAE [9]	SmAE
Test error(%)	1.68	1.19	1.18	1.14	1.15

Classification on MNIST Variation. As a benchmark, MNIST variation [19] is usually used to evaluate deep learning algorithms recently. It contains various classification tasks. Here, we choose three tasks: "mnist-basic", "mnist-rot" and "rect" to test our method. "mnist-basic" is a subset dataset of MNIST. Images in "mnist-rot" are rotated by an angle generated uniformly between 0 and 2π radians. The above two datasets both contain 10000 samples for training, 2000 samples for validating and 50000 samples for testing. "rect" is a binary classification task to discriminate between wide and long rectangles. It has 1000, 200 and 50000 samples as training, validating and testing sets respectively.

As the deep learning algorithms have reached the state-of-the-art performance, here we compare those classic deep learning methods with properly stacked layers, including stacked AE (SAE), DAE, CAE and Restricted Boltzman Machine (RBM). In our method, we use the unsupervised target neighbor to define the weight function. After layerwise pretraining, the network is further finetuned with a softmax classifier. The hyper-parameters such as layer sizes, sparsity penalty and kernel bandwidth are obtained by using the validation set. In Fig. 4, we report the classification accuracy of all comparison methods, in the above three classification tasks, where the digit after the method name marks the number of layers in its deep network, e.g., "SmAE-2" means the proposed model network is constructed by stacking two SmAEs. The other comparison results

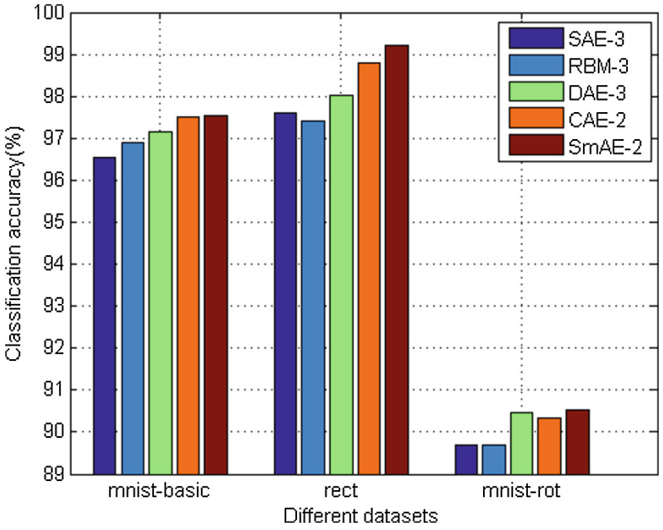

Fig. 4. Classification accuracy on MNIST variations (Color figure online).

are quoted from the related published literatures [7] and [9], whose reported best results are shown in this table. In most cases, smooth autoencoders achieve comparable and even better results than the other current state-of-the-art methods.

To explicitly show the good property of smooth autoencoder, we do the 2D visualization on the mnist-basic dataset. In Fig. 5, different colors represent the images from 0∼9 digits. Principal Component Analysis is used to reduce the dimensions of the representations learned by conventional autoencoder, unsupervised smooth autoencoder and supervised smooth autoencoder. All the three methods are respectively abbreviated as AE-2, SmAE-2.unsup and SmAE-2.sup, since the number of hidden layers is two. We use all the 10000 training data and randomly choose 10000 test data for comparison. As shown in the figure, the unsupervised SmAE-2 can capture locality property which makes the data more separable than the conventional autoencoder. By further using the label information, the supervised SmAE-2 can learn representations which distinctly increase the within-class compactness. In addition, we randomly choose 5 samples of each digit from test data and construct the affinity matrix as shown in Fig. 6. As we can see, samples from the same digit are highly compacted in SmAE-2.sup.

5.2 Face Recognition

In this section, we conduct extensive experiments on two standard face datasets, extended YaleB [20] and AR face [21], to evaluate the proposed method. Extended YaleB dataset consists of 2,414 frontal-face images of 38 persons under 64 illumination conditions. The original images are cropped to 50×50 pixels. For each person, we randomly sample 32 images for training and the rest for testing. AR dataset contains the frontal images of 126 persons, which are collected across two

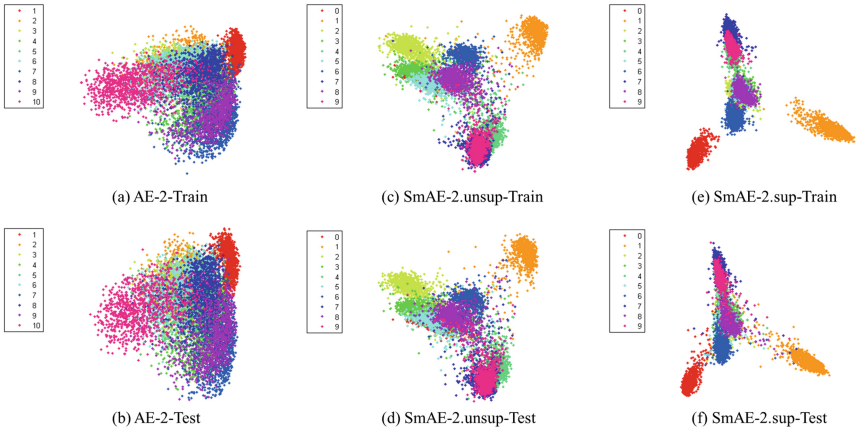

Fig. 5. 2D visualization of within-class compactness

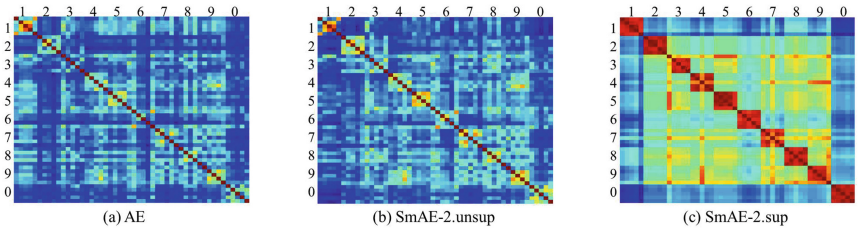

Fig. 6. Affinity Matrix of mnist-basic

separate sessions with different facial expressions, lighting and occlusion variations. Following the standard evaluation protocol, we random choose 50 males and 50 females with 2,600 images to verify the proposed methods. Among them, 20 images per person are randomly selected for training and the rest six of each person for testing. All the images are cropped into 80×64 pixels gray images.

First, we perform single layer smooth autoencoder and compare it with different autoencoder variants. Besides unsupervised SmAE, here we also consider the supervised case by using manual labels in the construction neighbor weights to further enhance the discriminative feature learning. Both methods are respectively abbreviated as SmAE.unsup and SmAE.sup. In our methods, the nearest neighbors $k = 5$ and $k = 10$ are respectively set for the unsupervised and supervised cases. The number of the hidden units is set to be 600. A softmax classifier is connected to the output layer to check the performance on face recognition. The parameter σ is set to be 0.05, β and ρ are set to 0.1. For other methods, we try to tune their referred parameters and reported the best results. As shown in Table 2, under both unsupervised and supervised setting, smooth autoencoder can still improve the recognition performance by using the preservation of manifold structures in our proposed SmAE.

Table 2. Single Layer Face Recognition Rate(%)

Method	Extended YaleB	AR face	Mean
SpAE	94.07	93.50	93.79
DAE	93.57	93.17	93.37
CAE	93.82	93.83	93.83
SmAE.unsup	94.17	94.00	94.09
SmAE.sup	95.16	95.00	95.08

Table 3. Face recognition rates (%) on Extended YaleB and AR face database

Method	Extended YaleB	AR face
K-SVD	90.5	90.0
SRC	88.6	74.5
LC-KSVD	95.0	93.7
DDL-PC2	95.3	96.6
MMDL	97.3	97.3
SmAE-2	**98.2**	**98.4**

Next, we further compare our method with several related face feature representation methods by using the solider sparse coding theory, including K-SVD [22], SRC [23], LC-KSVD [24], DDL-PC2 [25] and MMDL [26]. All these comparison methods have demonstrated their robust representation ability in the description of images. To learn more abstract and robust representation, in our method we stack two-layer smooth autoencoders by using supervised target neighbor definition. The number of units in both hidden layers are set to 1000. k is set to 5 and the parameters σ, β and ρ are simply set to be 0.1 without further tuning. We conduct 10-round random experiments, and then report average recognition rate in Table 3. As we can see, smooth autoencoders achieve the highest accuracies on both datasets. This further demonstrates that the representation learned by our method is more effective for image classification.

6 Conclusions and Future Work

In this paper, we present a novel neural network method: Smooth Autoencoder. By using the encoding of a sample to reconstruct its target neighbors instead of itself, the relationship between similar local features can be captured. We further show the representations learned by our method are robust to small variations. By making use of supervisory information, smooth autoencoder can enhance within-class compactness which is beneficial for classification tasks. Experimental results show that our approach improves the conventional autoencoder and achieve comparable or better performance on handwritten digit recognition and

face recognition. For future work, we can extend the target neighbor definition to different applications, such as action recognition in spatial-temporal videos.

Acknowledgement. This work is partially supported by the National Natural Science Foundation of China under contract No. 61390515, 61272319, and 61202297 and Natural Science Foundation of Fujian Province under contract No.2013J01239.

References

1. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS (2012)
2. Ji, S., Xu, W., Yang, M., Yu, K.: 3d convolutional neural networks for human action recognition. IEEE Trans. Pattern Anal. Mach. Intell. **35**, 221–231 (2013)
3. Wang, N., Yeung, D.Y.: Learning a deep compact image representation for visual tracking. In: NIPS (2013)
4. Ngiam, J., Coates, A., Lahiri, A., Prochnow, B., Le, Q.V., Ng, A.Y.: On optimization methods for deep learning. In: ICML (2011)
5. Ranzato, M., Boureau, Y.L., LeCun, Y.: Sparse feature learning for deep belief networks. In: NIPS (2007)
6. Xie, J., Xu, L., Chen, E.: Image denoising and inpainting with deep neural networks. In: NIPS (2012)
7. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and composing robust features with denoising autoencoders. In: ICML (2008)
8. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A.: Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. J. Mach. Learn. Res. **11**, 3371–3408 (2010)
9. Rifai, S., Vincent, P., Muller, X., Glorot, X., Bengio, Y.: Contractive auto-encoders: explicit invariance during feature extraction. In: ICML (2011)
10. Bengio, Y., Courville, A., Vincent, P.: Representation learning: a review and new perspectives. IEEE Trans. Pattern Anal. Mach. Intell. **35**, 1798–1828 (2013)
11. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. Science **290**, 2323–2326 (2000)
12. Tenenbaum, J.B., De Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science **290**, 2319–2323 (2000)
13. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput. **15**, 1373–1396 (2003)
14. Yu, K., Zhang, T., Gong, Y.: Nonlinear learning using local coordinate coding. In: NIPS (2009)
15. Balasubramanian, K., Yu, K., Lebanon, G.: Smooth sparse coding via marginal regression for learning sparse representations. In: ICML (2013)
16. Weston, J., Ratle, F., Mobahi, H., Collobert, R.: Deep learning via semi-supervised embedding. In: Montavon, G., Orr, G.B., Müller, K.-R. (eds.) Neural Networks: Tricks of the Trade, 2nd edn. LNCS, vol. 7700, pp. 639–655. Springer, Heidelberg (2012)
17. Hinton, G.E., Zemel, R.S.: Autoencoders, minimum description length, and helmholtz free energy. In: NIPS (1994)
18. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**, 2278–2324 (1998)

19. Larochelle, H., Erhan, D., Courville, A., Bergstra, J., Bengio, Y.: An empirical evaluation of deep architectures on problems with many factors of variation. In: ICML (2007)
20. Georghiades, A.S., Belhumeur, P.N., Kriegman, D.: From few to many: Illumination cone models for face recognition under variable lighting and pose. IEEE Trans. Pattern Anal. Mach. Intell. **23**, 643–660 (2001)
21. Martinez, A.M.: The ar face database. Technical report 24 (1998)
22. Aharon, M., Elad, M., Bruckstein, A.: K-svd: an algorithm for designing overcomplete dictionaries for sparse representation. IEEE Trans. Signal Process. **54**, 4311–4322 (2006)
23. Wright, J., Yang, A.Y., Ganesh, A., Sastry, S.S., Ma, Y.: Robust face recognition via sparse representation. IEEE Trans. Pattern Anal. Mach. Intell. **31**, 210–227 (2009)
24. Jiang, Z., Lin, Z., Davis, L.S.: Learning a discriminative dictionary for sparse coding via label consistent k-svd. In: CVPR (2011)
25. Guo, H., Jiang, Z., Davis, L.S.: Discriminative dictionary learning with pairwise constraints. In: Lee, K.M., Matsushita, Y., Rehg, J.M., Hu, Z. (eds.) ACCV 2012, Part I. LNCS, vol. 7724, pp. 328–342. Springer, Heidelberg (2013)
26. Wang, Z., Yang, J., Nasrabadi, N., Huang, T.: A max-margin perspective on sparse representation-based classification. In: CVPR (2013)