Li *et al. EURASIP Journal on Advances in Signal Processing* (2015) 2015:37
DOI 10.1186/s13634-015-0222-1

EURASIP Journal on
Advances in Signal Processing
a SpringerOpen Journal

RESEARCH **Open Access**

The effect of whitening transformation on pooling operations in convolutional autoencoders

Zuhe Li[1,2*], Yangyu Fan[1] and Weihua Liu[1]

Abstract

Convolutional autoencoders (CAEs) are unsupervised feature extractors for high-resolution images. In the pre-processing step, whitening transformation has widely been adopted to remove redundancy by making adjacent pixels less correlated. Pooling is a biologically inspired operation to reduce the resolution of feature maps and achieve spatial invariance in convolutional neural networks. Conventionally, pooling methods are mainly determined empirically in most previous work. Therefore, our main purpose is to study the relationship between whitening processing and pooling operations in convolutional autoencoders for image classification. We propose an adaptive pooling approach based on the concepts of information entropy to test the effect of whitening on pooling in different conditions. Experimental results on benchmark datasets indicate that the performance of pooling strategies is associated with the distribution of feature activations, which can be affected by whitening processing. This provides guidance for the selection of pooling methods in convolutional autoencoders and other convolutional neural networks.

Keywords: Convolutional neural network; Sparse autoencoder; Image classification; Computer vision; Unsupervised learning; Deep learning

1 Introduction

Unsupervised learning has been successfully used for feature extraction in many scientific and industrial applications such as pattern recognition and computer vision [1]. It is adopted to extract generally useful features from unlabelled data. Thus redundant inputs can be removed, and only essential aspects of the data are preserved [2]. As an unsupervised learning algorithm, an autoencoder is a neural network which can discover useful structures of the input data. It is trained in a way that sets the target values to be equal to the inputs [3]. However, it is impossible to learn features on entire images when the size of images becomes large because of computational expense. Then, we can take advantage of convolutional neural networks (CNNs) [4] to exploit local connectivity without training the network on full images [5]. CNNs are a special kind of multi-layer neural networks that have been successfully applied to computer vision.

In this context, convolutional autoencoders are proposed as unsupervised feature extractors to learn features and discover good CNN initializations from high-resolution images [2]. They have been adopted in semi-supervised scenarios where the label information is limited or weak, such as video recognition or pedestrian detection [6,7]. In convolutional networks, pooling layers are indispensable parts which combine the outputs of neuron clusters. This operation can reduce the resolution of feature maps to achieve spatial invariance [8-10]. In previous work, several pooling methods have been already proposed. However, the selection of pooling approaches in convolutional networks is mainly dependent upon experience. In our experiments, we found that the performance of a convolutional autoencoder with different pooling strategies varies with pre-processing techniques. Inspired by this, we evaluated the performance of pooling operations in different conditions and explored the underlying factors affecting the performance of pooling operations.

Since adjacent pixels are highly correlated, the raw input is redundant if we are training on images [11]. Whitening transformation then intends to improve the performance by making the input less redundant [12]. In convolutional

---

* Correspondence: zuheli@126.com
[1]School of Electronics and Information, Northwestern Polytechnical University, 127 West Youyi Road, Xi'an 710072, China
[2]School of Computer and Communication Engineering, Zhengzhou University of Light Industry, 5 Dongfeng Road, Zhengzhou 450002, China

© 2015 Li et al.; licensee Springer. This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly credited.

autoencoders, whitening transformation is applied to image patches sampled randomly from the dataset. And the same pre-processing step should also be performed on convolved patches to get correct activations. So the distribution of feature activations is changed by this transformation, and the performance of pooling operations is affected indirectly.

The aim of our work is therefore to explore the relationship between pooling operations and whitening processing. Taking image classification for example, we applied sparse autoencoders to benchmark datasets like STL [12] and CIFAR [13] using only single-layer networks. And we tested the classification accuracy of convolutional autoencoders using different pooling approaches both with and without whitening transformation. To further confirm this correlation, we presented a pooling approach which can automatically adjust the algorithm according to the entropy of image features. Our main contribution is that we find a correlation existing between the two operations. For instance, average pooling outperforms max pooling when whitening transformation is applied in certain circumstances. This overturns the traditional view that max pooling is always superior compared to average pooling.

In the following section, we will start by reviewing related work and then move on to describe the architecture of convolutional autoencoders in ‘Overall architecture’ section. We will interpret whitening transformation and pooling operations respectively in ‘Whitening transformation and convolutional autoencoder’ and ‘Pooling operation’ section. We then present experimental results and analysis on various datasets in ‘Experiments’ section. In the last section, we will draw conclusions and discuss future work.

2 Related work

A lot of schemes for feature extraction have been proposed since the introduction of unsupervised pre-training [14]. Feature learning algorithms such as sparse autoencoders [15,16] have been frequently considered in the existing literature. In the pre-processing step, several techniques have been adopted to achieve robust invariant features from unlabeled input data. For example, denoising autoencoders [17] are trained with corrupted or noisy versions of training samples to learn anti-noise features. And local transformations such as translation, rotation, and scaling in images have been applied to the feature learning algorithms in order to obtain transformation-invariant representations [18].

Whitening transformation is also an important pre-processing step adopted in deep learning algorithms to learn good features by decorrelating the input. To our knowledge, there have not been any attempts to study the relationship between whitening transformation and pooling operation in convolutional neural networks, especially in convolutional autoencoders. However, some researchers have focused on the impact of whitening and pooling separately in feature learning systems.

For example, Coates et al. [12] have studied the effect of whitening on single-layer networks in unsupervised feature learning. They applied several feature learning algorithms to benchmark datasets using only single-layer networks and presented the performance for all algorithms both with and without whitening. The results suggest that whitening operation might improve the performance of the networks.

Jarrett et al. [19] have studied the impact of changes to the pooling approaches frequently adopted between layers of features. Scherer et al. [10] have evaluated the pooling operations in convolutional architectures by directly comparing them on a fixed architecture for various object recognition tasks. Zeiler et al. [20] have proposed a stochastic pooling algorithm which randomly picks the activation within the pooling region based on a multinomial distribution. The results of these studies show that a max pooling operation is superior for capturing invariant features in most cases. However, Boureau et al. [21,22] have considered the impact of different types of pooling both in theory and in practice. They gave extensive comparisons for object recognition tasks based on the theoretical analysis of max pooling and average pooling. Their analysis leads to a prediction that max pooling is most suitable for the separation of sparse features. In other words, whether max pooling may perform better depends on the data and features.

Coincidentally, whitening transformation can affect the distribution of image features by making image patches less correlated with each other. Inspired by work in [21,22], we therefore conducted trials to determine whether whitening operations affect the performance of pooling.

3 Overall architecture

Instead of training networks on full images, we can use convolutional networks to reduce the computational cost of learning features from large-size images with autoencoders. First, small-size image patches are sampled randomly from the training set and trained with autoencoders. Then, the learned features should be convolved with larger images and different feature activations at each location can be obtained [11].

The concrete architecture of a single-layer convolutional autoencoder for gray-scale image classification is depicted in Figure 1. For color images with three color channels (RGB), the intensities from all the color channels can be combined into one long vector in the training process. And each image can be convolved in every image channel separately to improve efficiency in the convolutional layer.

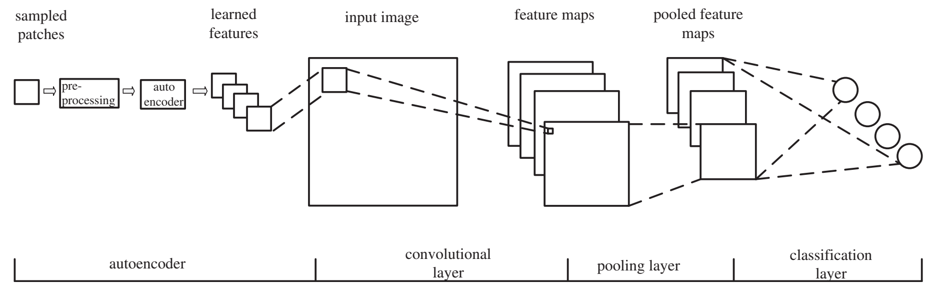


Figure 1 Architecture of a convolutional autoencoder for image classification.

Figure 2 gives the flowchart of the proposed method to evaluate the effect of whitening transformation on pooling operations. We apply whitening transformation with different parameters to image patches in the pre-processing step and explore its impact on the performance of the image classification system with different pooling approaches.

4 Whitening transformation and convolutional autoencoder

4.1 Whitening transformation

As an important pre-processing step to remove redundancy, whitening transformation is applied to image patches before training sparse autoencoders and the same transformation is performed on every image region to be convolved in convolutional autoencoders. The goal of whitening is to make features less correlated with each other and having identity covariance matrix [11]. In practice, whitening transformation is usually combined with principal component analysis (PCA) or zero-phase whitening filters (ZCA) [23]. In this paper, we adopt the ZCA whitening because it has been widely used in previous works.

In ZCA whitening, processing is required to ensure that the data has zero-mean before computing the covariance matrix. Then, normalized image patches x^i are stored as column vectors and the covariance matrix is computed as follows:

$$\Sigma = E\left(xx^T\right) = \frac{1}{m}\sum_{i=1}^{m}\left(x^i\right)\left(x^i\right)^T \tag{1}$$

where m is the number of the image patches sampled from the dataset. Then, we can compute the eigenvectors $u_1, u_2,\ldots, u_n$ and corresponding eigenvalues $\lambda_1, \lambda_2,\ldots, \lambda_n$ of the covariance matrix. And the ZCA whitening is defined as:

$$\begin{aligned} x_{ZCAwhite} &= W_{ZCAwhite}x \\ &= U\begin{bmatrix} \frac{1}{\sqrt{\lambda_1+\varepsilon}} & 0 & \cdots & 0 \\ 0 & \frac{1}{\sqrt{\lambda_2+\varepsilon}} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \frac{1}{\sqrt{\lambda_i+\varepsilon}} \end{bmatrix}U^T x \end{aligned} \tag{2}$$

where U is the matrix $[u_1, u_2,\ldots, u_n]$ and ε is a small constant added to the eigenvalues λ_i. With the regularization term ε, the data will not blow up or produce instability when an eigenvalue is close to 0. It also has an effect of low-pass filtering the input image [11]. The regularization term ε is rather important, and its value should not be set too low or too high. Results of whitening transformation with different values of ε will be demonstrated in 'Experiments' section.

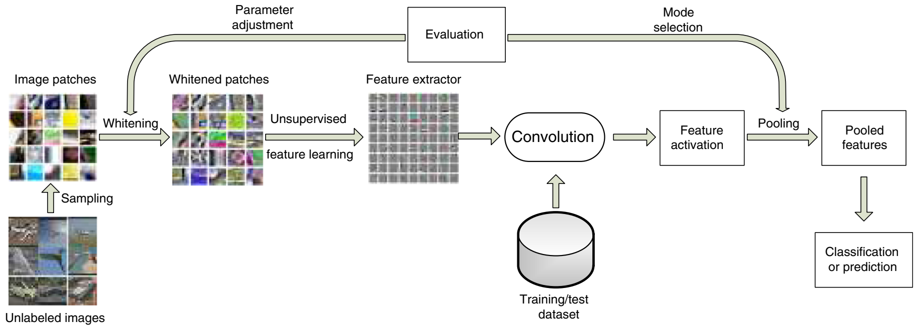


Figure 2 Flowchart of the proposed method to evaluate the effect of whitening on pooling in convolutional autoencoders.

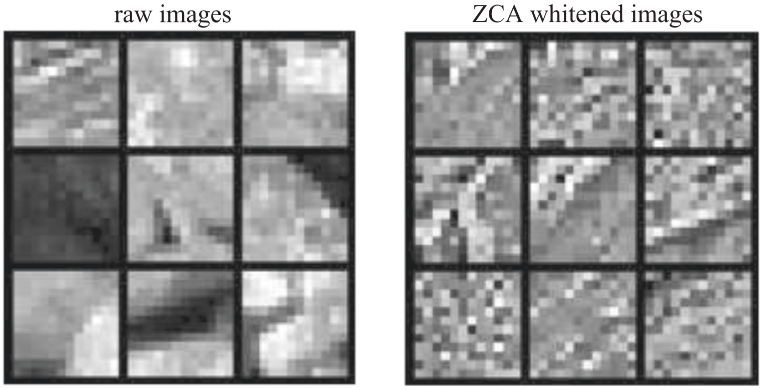


Figure 3 The effect of ZCA whitening on image patches.

This transformation therefore decorrelates the image patches and has an impact on the data distribution. Figure 3 illustrates the effect of ZCA whitening on patches sampled from natural images. Image patches have the same variance, and edges in these patches are enhanced after whitening. However, the input is no longer constrained to [0, 1] if whitening transformation is used. In a sparse autoencoder, the problem can be solved with a linear decoder which makes an adjustment to the activation function of the output layer.

4.2 Sparse autoencoder

As a typical unsupervised learning algorithm, an autoencoder is a neural network with a symmetrical structure to discover representative information from unlabeled training examples by reconstructing the input values. Given an input vector $x^{(n)} \in R^D$ where D is the size of the input, an autoencoder applies backpropagation to set the output value $y^{(n)} = x^{(n)}$. The outputs of a neural network can be formally described as:

$$y^{(n)} = \sigma\left(W^T \sigma\left(Wx^{(n)} + b\right) + c\right) \tag{3}$$

where σ is the activation function like sigmoid function, W is the encoding weight matrix connecting the input layer and the hidden layer, W^T is the decoding weight matrix, b is the encoding bias, and c is the decoding bias.

Considering the sparsity constraint together, the objective function of a sparse autoencoder is expressed as follows [24]:

$$\begin{aligned} J(W, b, \mathrm{c}) = \frac{\lambda}{M} \sum_{n=1}^{M} \left\| \sigma\left(W^T \sigma\left(Wx^{(n)} + b\right) + c\right) - x^{(n)} \right\|_2^2 \\ + S\left(\{W, b\}, x^{(1)}, \ldots, x^{(M)}\right) \end{aligned} \tag{4}$$

where λ is the weight decay parameter, M is the number of training data, and S is the sparse penalty function imposing sparsity constraints on the neural network. Thus, feature extractors W are obtained by

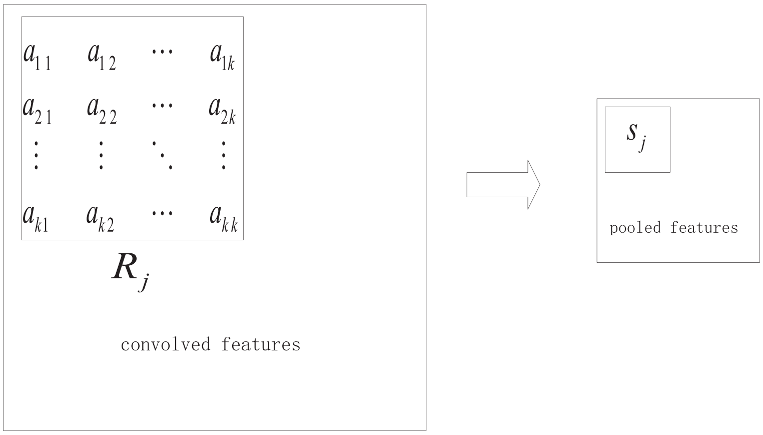

a_{11} a_{12} $\cdots$ a_{1k}
a_{21} a_{22} $\cdots$ a_{2k}
$\vdots$ $\vdots$ $\ddots$ $\vdots$
a_{k1} a_{k2} $\cdots$ a_{kk}
R_j
convolved features
s_j
pooled features

Figure 4 The process of pooling operations.

minimizing the objective function with backpropagation algorithms.

As described in 'Whitening transformation' section, the input is out of the range [0, 1] when whitening processing is applied to image patches. Therefore, conventional activation functions which map output values to the range [0, 1] are no longer suitable for the output layer of an autoencoder. Here, we use a linear decoder with the activation function $\sigma(x) = x$ in the output layer. Additionally taking into account the whitening transformation, we modify the objective function as follows:

$$\begin{aligned} J(W, b, \mathrm{c}) &= \frac{\lambda}{M}\sum_{n=1}^{M}\left\| \begin{matrix} W^T\sigma\left(WW_{ZCAwhite}x^{(n)}+b\right) \\ +c - W_{ZCAwhite}x^{(n)} \end{matrix} \right\|_2^2 \\ &\quad + S\left(\{W, b\}, W_{ZCAwhite}x^{(1)}, \ldots, W_{ZCAwhite}x^{(M)}\right) \end{aligned} \tag{5}$$

And the learning rule for W^T and c in the backpropagation algorithm is especially given as follows:

$$\Delta W^T = \left(y^{(n)} - x^{(n)}\right)\cdot\left(\sigma\left(WW_{ZCAwhite}x^{(n)}+b\right)\right)^T \tag{6}$$

$$\Delta c = y^{(n)} - x^{(n)} \tag{7}$$

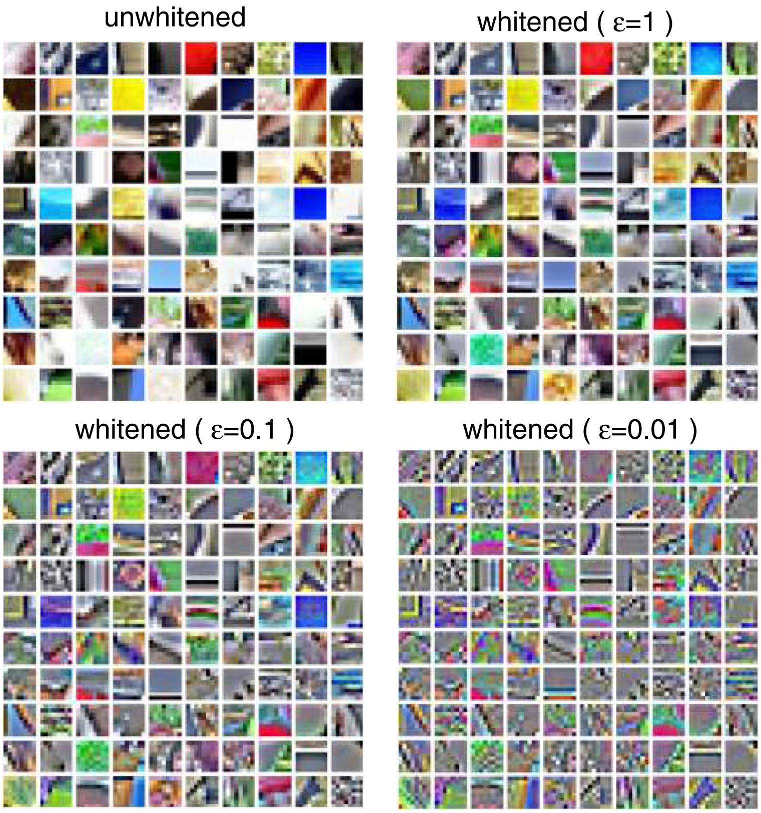
4.3 Convolution

Features learned over small image patches should be convolved with larger images in order to get feature activations at each location of whole images. To obtain correct feature activations with feature extractors learned from whitened image patches, we should normalize each image patch in each image to zero mean and multiply each normalized image patch x_i by $W_{ZCAwhite}$ for each activation:

$$a_i = \sigma\left(WW_{ZCAwhite}x_i + b\right) \tag{8}$$

where W denotes the feature learned by a sparse autoencoder.

Therefore, image features (i.e., W) learned by sparse autoencoders are affected in case whitening transformation is applied to image patches. And the distribution of feature maps after convolution is naturally different from that without whitening processing. For the reason that the



unwhitened

whitened (ε=1)

whitened (ε=0.1)

whitened (ε=0.01)

Figure 5 The first 100 image patches with and without whitening.

performance of different pooling approaches varies with the distribution of feature maps, we can say pooling operations are influenced by whitening transformation.

For color images, we simultaneously perform 2D convolution [11] in each color channel to improve efficiency. Furthermore, we implement whitening by multiplying the feature extractor W by $W_{ZCAwhite}$ before convolution to avoid repeating 'whitening operation' on each patch in the convolution process. The algorithm for image feature activation acquisition with 2D convolution in convolutional autoencoders is summarized as follows.

Algorithm 1: Given mean patch matrix mp, ZCA whitening matrix $W_{ZCAwhite}$, feature extractor W, b and a large image I with color channel R, G, and B, perform the following steps.

(1) Compute the feature matrix $\hat{W} = W \times W_{ZCAwhite}$ and the bias unit $\hat{b} = b - W \times W_{ZCAwhite} \times mp$;
(2) Divide $\hat{W}$ and each image I into three color channels to obtain feature matrices and image matrices: $\hat{W}_R$, $\hat{W}_G$, $\hat{W}_B$, I_R, I_G, and I_B in each color channel, respectively;
(3) For each feature number in each color channel, flip the feature matrix and perform 2D convolution of image matrix with flipped feature matrix. Add up the convolution results in R, G, and B channel;
(4) For the convolution results of each feature number, add the bias unit $\hat{b}$ and apply the sigmoid function to get the hidden activation.

5 Pooling operation

Pooling operation can reduce the resolution of feature maps and achieve spatial invariance by combining the outputs of neuron clusters in convolutional networks [4,25,26]. In this layer, each output feature map combines the input from a $k \times k$ patch of units in the previous layer. The pooling region can vary in size and be overlapping [10]. The process is depicted in Figure 4 in details.

The essence of pooling operations is to find a function that aggregates the information within a small local region R_j to generate a pooled feature s_j. This operation can be expressed as follows:

$$s_j = pool(a_i) \forall i \in R_j \tag{9}$$

where R_j is pooling region j of size $k \times k$ in a feature map and i is the index of each element [20]. Although several novel pooling algorithms such as stochastic pooling [20] have been proposed, average pooling and max pooling are still the most commonly used approaches. These two

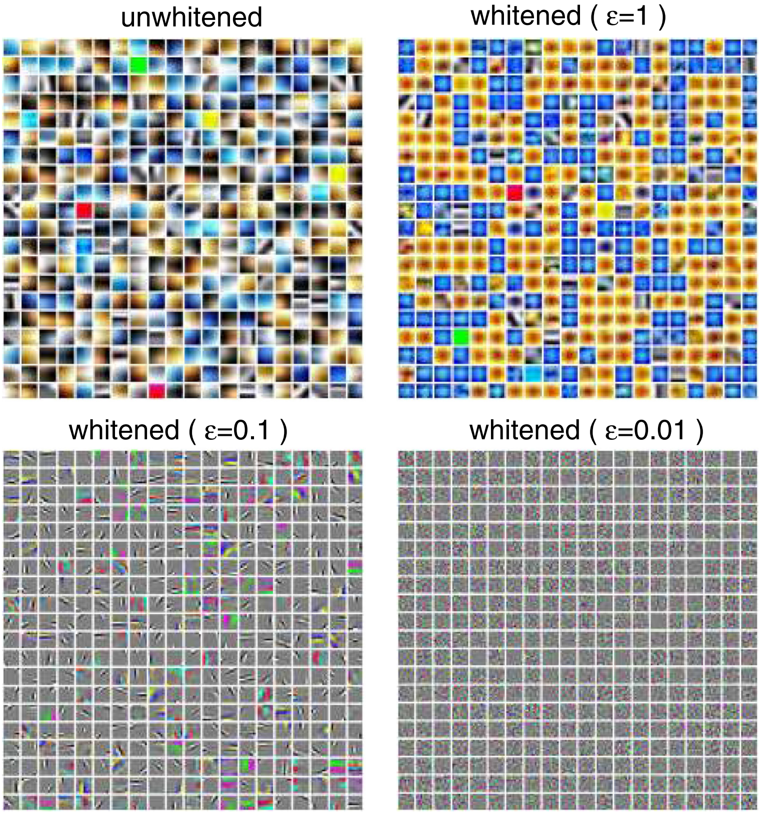


Figure 6 Features learned by autoencoders with and without whitening.

Table 1 Classification results on reduced STL-10 dataset without overlapping pooling

Pooling size	Pooling type	Unwhitened	Whitened ($\varepsilon = 1$)	Whitened ($\varepsilon = 0.1$)	Whitened ($\varepsilon = 0.01$)
19 × 19	Avg	75.44%	74.53%	81.31%	80.00%
	Max	77.75%	74.97%	79.84%	77.50%
	Ada	76.66%	75.94%	80.50%	80.78%
5 × 5	Avg	72.53%	73.53%	78.38%	75.97%
	Max	75.34%	74.25%	77.56%	74.56%
	Ada	73.06%	73.59%	77.87%	75.81%

conventional choices for *pool()* can be depicted in Equations 4 and 5, respectively:

$$s_j = \frac{1}{n}\sum_{i\in R_j} a_i = \frac{1}{k \times k}\sum_{i\in R_j} a_i \tag{10}$$

$$s_j = \max_{i\in R_j} a_i \tag{11}$$

where n denotes the number of features in a pooling region.

It has been shown that pooling operations can influence the performance of convolutional networks. For example, Scherer et al. have given the conclusion that a max pooling operation is superior for learning feature from image-like data [10]. However, this is not always true in our experiments with convolutional autoencoders. In [22], Boureau et al. have proved that max pooling performs better when the features in R_j are very sparse (i.e., have a very low probability of being active). In other words, whether max pooling is superior to average pooling depends on the distribution of convolved features.

Here, we proposed a method to evaluate the sparsity degree of the convolved features using entropy theory [27]. Concretely, we can first normalize the activations within a pooling region as follows:

$$p_i = \frac{a_i}{\sum\limits_{i\in R_j} a_i} \tag{12}$$

Thus, each normalized activation p_i is constrained to [0, 1] and the sum of them equals to 1. And we can take these activations as the probabilities of an information source. Then, the sparsity degree is defined as:

$$\alpha = 1 - \frac{H(p)}{\log_2 n} = 1 - \frac{-\sum\limits_{i=1}^{n} p_i \log_2 p_i}{\log_2 n} \tag{13}$$

where $H(p)$ is the entropy of the normalized activations in a pooling region.

For a given n, $H(p)$ reaches its maximum and equals $\log_2 n$ when all the p_i are equal in the most uncertain situation. Oppositely, $H(p)$ equals zero when all the p_i but one are zero. In this case we are certain of the outcome [27]. Thus, $H(p)$ is rather small and α is close to 1 when the features are very sparse. $H(p)$ gets close to $\log_2 n$, and α will approach zero in the opposite situation. In other words, α varies from 0 to 1 according to the sparsity degree of the features in a pooling region. Therefore, we can choose α as a reasonable measure of the feature sparsity degree in a pooling region. And we can use it as a reference value when we are selecting pooling methods. For example, max pooling should be taken in to account if α is rather large.

In order to validate the above theory, we propose an adaptive pooling scheme which combines max pooling and average pooling using the sparsity degree α:

$$s_j = \alpha \max_{i\in R_j} a_i + (1\text{-}\alpha)\frac{1}{n}\sum_{i\in R_j} a_i \tag{14}$$

In this manner, the pooling layer can automatically switch between max pooling and average pooling. In theory, the adaptive method should be well suited to the separation of features in all circumstances. And this

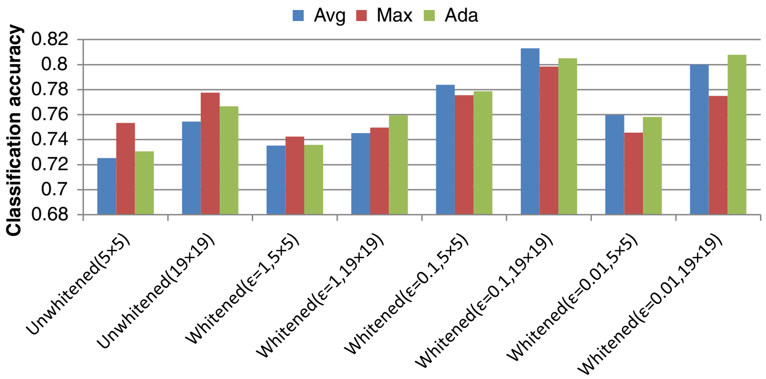


Figure 7 Classification results on reduced STL-10 dataset without overlapping pooling.

Table 2 Classification results on reduced STL-10 dataset with overlapping pooling

Pooling size	Pooling stride	Pooling type	Unwhitened	Whitened ($\varepsilon = 0.1$)
19 × 19	10	Avg	75.80%	81.34%
		Max	78.12%	80.22%
		Ada	77.34%	81.06%
	5	Avg	76.06%	81.78%
		Max	78.56%	80.50%
		Ada	77.84%	81.25%

algorithm will be applied to convolutional autoencoders both with and without whitening processing in the next section.

6 Experiments

We mainly tested our conjectures by running thorough experiments on the STL-10 dataset, which provides a large set of unlabeled examples for unsupervised feature learning [12]. Additionally, we conducted experiments on the CIFAR-10 [13] dataset to evaluate the performance of convolutional autoencoders with various pooling approaches. Since our main purpose is to test the effect of whitening transformation in different conditions, we selected to use convolutional autoencoders with only one hidden layer in order to avoid interference from other factors.

6.1 STL-10 dataset

We used a sparse autoencoder with 400 hidden units to learn features on a set of 100,000 small 8 × 8 patches sampled from the STL-10 dataset. We first trained the autoencoder without whitening processing. Then, we whitened the image patches with a regularization term $\varepsilon = 1, 0.1, 0.01$ respectively and repeated the training several times. All the autoencoders were trained using the sparsity parameter $\rho = 0.03$, sparsity weight $\beta = 5$, and the weight decay parameter $\lambda = 0.003$. In Figure 5, we select the first 100 image patches to demonstrate the effect of whitening processing in different conditions. And Figure 6 shows the corresponding features learned from the image patches.

It is obvious from Figures 5 and 6 that whitening transformation does affect image patches and corresponding features learned by sparse autoencoders. For example, more edge features can be learned when whitening processing is adopted. In addition, the value of ε also has an effect on the image patches and learned features when whitening pre-processing is used. As shown in Figure 6, the learned features look rather noisy if the value of ε is set to 0.01 and the data will be blurred if ε is set to 1. The best value of ε is 0.1 in our experiments.

Having learned features from image patches, we constructed convolutional neural networks for image classification on the reduced STL-10 dataset which consists of 64 × 64 images from four classes (airplane, car, cat, and dog). We selected to use the reduced STL-10 dataset for simplicity because we are chiefly concerned with the relationship between whitening and pooling. Therefore, experiments were made in an attempt to evaluate the performance of each pooling approach both with and without whitening transformation. In each case, we tested the performance for both 19 × 19 and 5 × 5 pooling sizes. Classification accuracy under each condition is presented in Table 1 and Figure 7, where Avg denotes average pooling, Max denotes max pooling, and Ada denotes the adaptive pooling algorithm proposed in 'Pooling operation' section.

As shown in Table 1 and Figure 7, whitening with a proper regularization term (e.g., 0.1) can greatly improve the classification accuracy no matter what method is adopted in the pooling layer. However, things will change when the value of ε is either too low or too high.

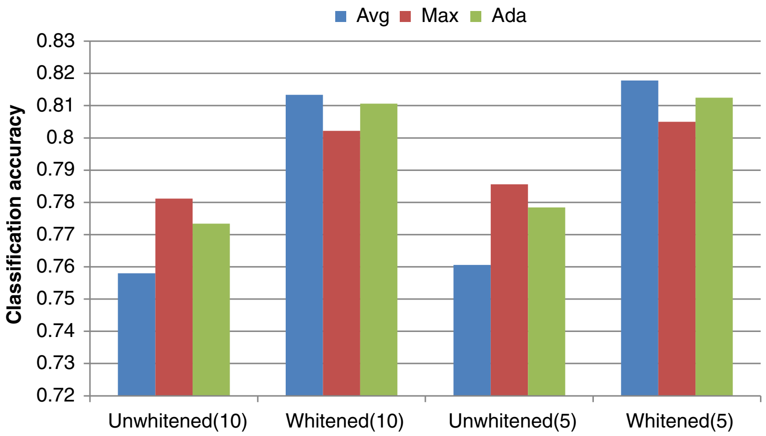


Figure 8 Classification results on reduced STL-10 dataset with overlapping pooling.

Table 3 Classification results on CIFAR-10 dataset

Pooling size	Pooling type	Unwhitened	Whitened ($\varepsilon = 0.1$)
8 × 8	Avg	59.84%	64.81%
	Max	60.55%	63.62%
	Ada	60.67%	65.48%
	Sto	61.06%	64.44%
5 × 5	Avg	57.34%	64.38%
	Max	59.56%	63.06%
	Ada	58.66%	63.59%
	Sto	58.60%	63.25%

If the value of ε is 1 and the pooling region size is 19 × 19, the classification accuracy even drops slightly compared to that without whitening. This is consistent with the results in Figure 6 that features learned by autoencoders become blurred and edge information is lost when ε is set to 1.

Moreover, we can observe that max pooling performs better than average pooling when whitening transformation is not applied to image patches. On the contrary, average pooling outperforms max pooling when whitening is adopted with a proper value of ε. The reason is that whitening transformation changes the distribution of image patches and learned features. And the image features after convolution become less sparse (i.e., have higher probability of being active) when whitening preprocessing is used. In this case, average pooling is more likely to perform better than max pooling.

And it should be noted that the proposed adaptive pooling shows strong adaptability with stable performance in all circumstances no matter whether whitening operation is adopted. Its performance is not the worst in all cases. In some cases (e.g., $\varepsilon = 1$ and $\varepsilon = 0.01$), it even performs better than both max pooling and average pooling. Since we make adjustments according to the sparsity degree of feature maps in the adaptive pooling algorithm, the experimental results further confirm that whether a pooling method is suitable depends on the sparsity degree of features activations. Considering other techniques like whitening pre-processing can affect the distribution of features activations, we can draw the inference that whitening transformation indirectly influences the selection of pooling methods. This gives us a revelation that we should pay attention to the relationship between these tricks adopted in feature learning systems and not just put them together.

We also conducted experiments to determine whether the above results would hold up when overlapping pooling is adopted. We tested the performance of three pooling methods with a pooling size of 19 × 19 and pooling strides of 5 and 10. We set the value of ε to 0.1 in whitening processing as it has been shown to be a proper value in above experiments. Classification results are given in Table 2 and Figure 8. The results are consistent with the above experiments.

6.2 CIFAR-10 dataset

To further validate the relationship between whitening transformation and pooling operation, we repeated the above experiments on the CIFAR-10 dataset. It is a widely used benchmark dataset that consists of 60,000 32 × 32 color images in ten classes. For whitening processing, we only used a regularization term $\varepsilon = 0.1$ because it is the best value of ε in previous experiments. And pooling region sizes of 8 × 8 and 5 × 5 were selected in the pooling layer. Besides, we use all of the same parameters as for the STL-10 dataset. Additionally, we also tested the performance of the novel stochastic pooling

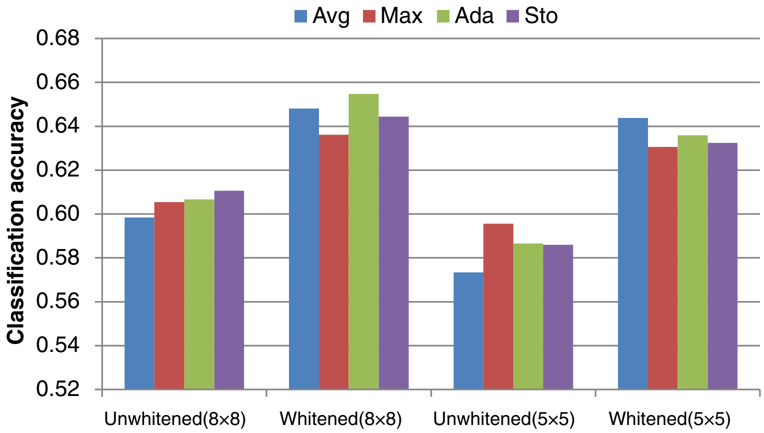


Figure 9 Classification results on CIFAR-10 dataset.

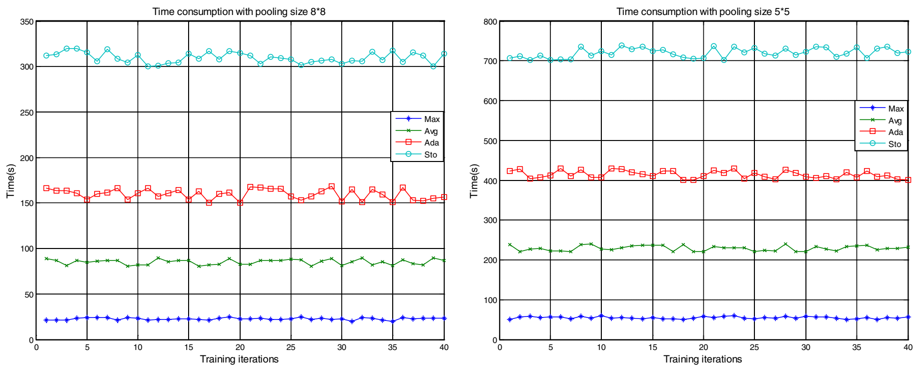


Figure 10 The results of time consumption for pooling operation with 40 iterations in the training process.

algorithm proposed in [20] which has been introduced in the 'Related work' section and made a comparison with conventional pooling methods and adaptive pooling we proposed. Classification results are given in Table 3 and Figure 9, where Sto denotes stochastic pooling.

Since our main task is to explore the relationship between whitening and pooling, we have to avoid the potential influence of other factors. This will bring a slight decline in the overall performance because we did not adopt any other tricks. So the results show that we did not achieve higher performance compared to prior work [12]. However, the results are in accord with the experiments on the STL-10 dataset and provide further evidence for the relationship between whitening and pooling. Max pooling is perfectly suited for situations without whitening, and average pooling performs better when whitening is applied to image patches.

In addition, the adaptability of the pooling method we proposed is confirmed again for its good performance in every condition. And the proposed method even outperforms stochastic pooling and conventional pooling in whitened condition with pooling region size of 8 × 8. In general, the stochastic pooling has characteristics of randomness since the activation within the pooling region is randomly picked. For instance, the performance of stochastic pooling is the worst in whitened condition with pooling region size of 5 × 5. However, the adaptive pooling method shows robust performance because feature activation is picked in a less reckless approach.

In order to make a comprehensive comparison between these pooling methods in convolutional autoencoders, we further evaluated their time consumption on the CIFAR-10 dataset in the training process. We adopted max pooling, average pooling, stochastic pooling, and adaptive pooling with pooling region sizes of 8 × 8 and 5 × 5 on a system with a quad-core 3.30 GHz CPU and 8 GB RAM. The convolution and pooling were implemented ten features at a time to avoid running out of memory, and 40 iterations were needed to obtain the 400 feature maps. Figure 10 illustrates the results of time consumption in the 40 iterations, in each of which ten feature maps are pooled. It can be seen that the computational complexity of adaptive pooling is somewhat higher than that of average pooling and max pooling because extra calculation is needed. However, it is still far lower than that of stochastic pooling.

7 Conclusions

In this paper, we have focused on the effect of whitening transformation on the recognition performance of pooling operations in convolutional autoencoders. On the basis of theoretical analysis, we proposed an adaptive pooling approach which can automatically switch between two conventional pooling modes according to the sparsity degree of convolved features. Extensive experiments conducted on the benchmark datasets reveal that the performance of pooling operations is associated with the distribution of convolved features. On the other hand, whitening transformation can just change the distribution of image features. In this sense, we can say whitening transformation has a certain impact on the performance of pooling operation. Experimental results also show that depending on whether whitening processing is adopted, either max or average pooling performs better. So considering the effect of whitening and other factors on the distribution of image features, we can determine an appropriate pooling approach on a theoretical basis.

Competing interests

The authors declare that they have no competing interests.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under Grant 61202314, China Postdoctoral Science Foundation under Grant 2012 M521801, China Postdoctoral Science Foundation Special Project under Grant 2014 T70937, and the Science and Technology Innovation Engineering Program for Shaanxi Provincial Key Laboratories under Grant 2013SZS15-K02.

Received: 11 December 2014 Accepted: 29 March 2015
Published online: 14 April 2015

References

1. M Ranzato, FJ Huang, YL Boureau, Y LeCun, Unsupervised learning of invariant feature hierarchies with applications to object recognition, in *Computer Vision and Pattern Recognition (CVPR)*, 2007, pp. 1–8
2. J Masci, U Meier, D Cireşan, J Schmidhuber, Stacked convolutional auto-encoders for hierarchical feature extraction, in *Artificial Neural Networks and Machine Learning - ICANN* (Springer, Berlin Heidelberg, 2011), pp. 52–59
3. AY Ng, *Sparse autoencoder. CS294A Lecture notes*, 2011, p. 72
4. Y LeCun, L Bottou, Y Bengio, P Haffner, Gradient-based learning applied to document recognition. Proc IEEE **86**(11), 2278–2324 (1998)
5. Y Bengio, Learning deep architectures for AI. Foundations Trends® Machine Learn **2**(1), 1–127 (2009)
6. P Sermanet, K Kavukcuoglu, S Chintala, Y LeCun, Pedestrian detection with unsupervised multi-stage feature learning, in *Computer Vision and Pattern Recognition(CVPR)*, 2013, pp. 3626–3633
7. A Makhzani, B Frey, *A Winner-Take-All Method for Training Sparse Convolutional Autoencoders. arXiv preprint arXiv:1409.2752*, 2014
8. DC Ciresan, U Meier, J Masci, L Maria Gambardella, J Schmidhuber, Flexible, high performance convolutional neural networks for image classification, in *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, 2011, p. 1237
9. A Krizhevsky, I Sutskever, GE Hinton, Imagenet classification with deep convolutional neural networks, in *Advances in neural information processing systems*, 2012, pp. 1097–1105
10. D Scherer, A Müller, S Behnke, Evaluation of pooling operations in convolutional architectures for object recognition, in *Artificial Neural Networks - ICANN* (Springer, Berlin Heidelberg, 2010), pp. 92–101
11. AY Ng, J Ngiam, CY Foo, Y Mai, Deep Learning, http://deeplearning.stanford.edu/wiki/index.php. Accessed 18 Nov 2014
12. A Coates, AY Ng, H Lee, An analysis of single-layer networks in unsupervised feature learning, in *International Conference on Artificial Intelligence and Statistics*, 2011, pp. 215–223
13. Krizhevsky, Master thesis. *Learning Multiple Layers of Features from Tiny Images*, University of Toronto, 2009
14. G Hinton, S Osindero, YW Teh, A fast learning algorithm for deep belief nets. Neural Comput **18**(7), 1527–1554 (2006)
15. C Poultney, S Chopra, Y LeCun, Efficient learning of sparse representations with an energy-based model, in *Advances in neural information processing systems*, 2006, pp. 1137–1144
16. I Goodfellow, H Lee, QV Le, A Saxe, AY Ng, Measuring invariances in deep networks, in *Advances in neural information processing systems*, 2009, pp. 646–654
17. P Vincent, H Larochelle, I Lajoie, Y Bengio, PA Manzagol, Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. J Machine Learn Res **11**, 3371–3408 (2010)
18. K Sohn, H Lee, *Learning Invariant Representations with Local Transformations. arXiv preprint arXiv:1206.6418*, 2012
19. K Jarrett, K Kavukcuoglu, M Ranzato, Y LeCun, What is the best multi-stage architecture for object recognition? in *Computer Vision, 2009 IEEE 12th International Conference on*, 2009, pp. 2146–2153
20. MD Zeiler, R Fergus, *Stochastic Pooling for Regularization of Deep Convolutional Neural Networks. arXiv preprint arXiv:1301.3557*, 2013
21. YL Boureau, F Bach, Y LeCun, J Ponce, Learning mid-level features for recognition, in *Computer Vision and Pattern Recognition(CVPR)*, 2010, pp. 2559–2566
22. YL Boureau, J Ponce, Y LeCun, A theoretical analysis of feature pooling in visual recognition, in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010, pp. 111–118
23. AJ Bell, TJ Sejnowski, Edges are the "independent components" of natural scenes, in *Advances in Neural Information Processing Systems*, 1997, pp. 831–837
24. QV Le, A Karpenko, J Ngiam, AY Ng, ICA with reconstruction cost for efficient overcomplete feature learning, in *Advances in Neural Information Processing Systems*, 2011, pp. 1017–1025
25. K Korekado, T Morie, O Nomura, H Ando, T Nakano, M Matsugu, A Iwata, A convolutional neural network VLSI for image recognition using merged/mixed analog-digital architecture, in *Knowledge-Based Intelligent Information and Engineering Systems*, 2003, pp. 169–176
26. DH Hubel, TN Wiesel, Receptive fields of single neurones in the cat's striate cortex. J Physiol **148**(3), 574 (1959)
27. CE Shannon, A mathematical theory of communication. ACM SIGMOBILE Mobile Comput Commun Rev **5**(1), 3–55 (2001)

Submit your manuscript to a SpringerOpen journal and benefit from:

- ► Convenient online submission
- ► Rigorous peer review
- ► Immediate publication on acceptance
- ► Open access: articles freely available online
- ► High visibility within the field
- ► Retaining the copyright to your article

Submit your next manuscript at ► springeropen.com