Auto-encoder for generating a transform invariant descriptor and transform parameters

Tadashi Matsuo[1,a)] Nobutaka Shimada[1]

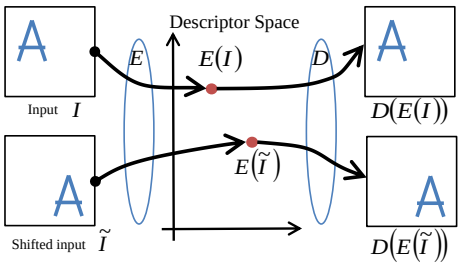

Fig. 1 Characteristics of an ordinary auto-encoder

Abstract

The auto-encoder method is a type of unsupervised dimensionality reduction method. However, an image and its spatially shifted version are encoded into different descriptors by an ordinary auto-encoder because each descriptor includes both a spatial pattern and its position in a window. This will be a problem when focusing on a pattern itself. To solve this, we proposed a transform invariant auto-encoder based on a new cost function. By the method, we can extract a transform invariant descriptor from an input, but we need an additional regressor to extract transform parameters required to restore the input. In addition, the cost function requires high computation cost by computational explosion when considering multiple types of transforms.

In this publication, we propose a novel auto-encoder that separates an input into a transform invariant descriptor and transform parameters. The proposed method does not require an additional regressor and it will overcome combinational explosion. The proposed method can be applied to various auto-encoders without requiring any special modules or labeled training samples. By applying it to shift transforms, we can achieve a spatial pattern descriptor and its relative position in a window. By some experiments, we demonstrate that the method can generate a pair of a transform invariant descriptor and a set of parameters for restoring the original input.

1. Introduction

The auto-encoder method [1], [2], [4] is a type of dimensionality reduction method. It can extract essential information from a vector via general non-linear mapping. Moreover, a mapping from a vector to a descriptor representing essential information can be automatically generated from a set of vectors without any supervising information.

When encoding images by the auto-encoder method, a descriptor of an image generally differs from that of a spatially shifted version of the image as shown in Fig. 1, because a pattern itself and its position are inseparably embedded into a descriptor. Although the denoising auto-encoder method [6] can extract desired components from an input including information to be ignored, it requires an ideal output for each training sample when training an auto-encoder. Therefore, to generate a descriptor representing a spatial subpattern in an image by such an auto-encoder, we need to normalize its spatial position in the images prior to training the auto-encoder. However, such a spatial normalization is generally difficult. For example, the normalization of the appearances of various hand–object interactions is not obvious and requires a pattern recognition technique to automatically find the standard for each image.

We have proposed a transform invariant auto-encoder that outputs a descriptor invariant with respect to a set of transforms[5]. By considering spatial shifts, the method can generate a shift invariant auto-encoder, which extracts a typical spatial subpattern without regard to its relative position in a window (Fig. 2). It can be applied to various auto-encoders without requiring any special modules or labeled training samples. By using the method, we can encode a spatial pattern itself even if target images are difficult to label or normalize, for example, the appearances of hand–object interactions. However, it ignores a position of the pattern. To estimate the position of the pattern, we had to introduce an additional inference model.

In this paper, we propose a novel auto-encoder that separates an input into a transform invariant descriptor and transform parameters. It consists of a transform invariant encoder, the corresponding decoder and a regressor of transform parameter as shown in 3. The encoder, decoder and regressor can be trained simulteneously and an external additional regressor is not required. The proposed method can be applied to various auto-encoders without requiring any special modules or labeled training samples. In addition, the proposed method will overcome combinational explosion, which occurs a problem when training a transform invariant auto-encoder for very widely various transforms.

[1] Ritsumeikan University
[a)] matsuo@i.ci.ritsumei.ac.jp

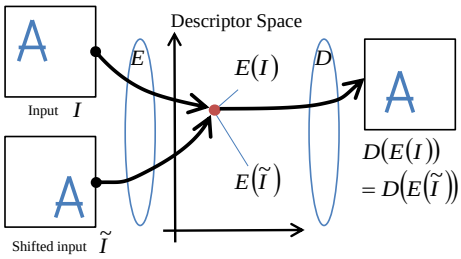

Fig. 2 Characteristics of a shift invariant auto-encoder

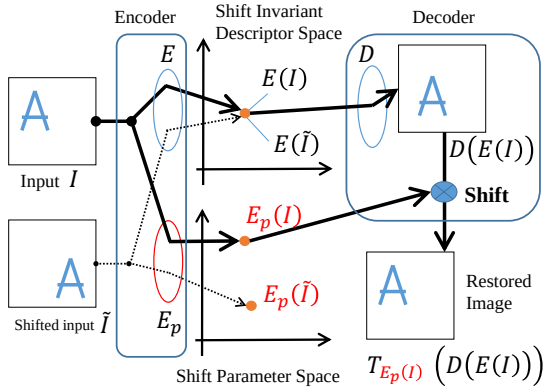

Fig. 3 Characteristics of a proposed auto-encoder

By applying it to shift transforms, we can achieve a spatial pattern descriptor and its relative position in a window. By an experiment, we demonstrate that the method can generate a pair of a transform invariant descriptor and a set of parameters for restoring the original input.

2. Ordinary auto-encoder

In general, an auto-encoder is so trained that the encoder–decoder combination approximately restores an input in a certain input set. It is formulated as a problem minimizing a cost function $C_{\mathrm{ord}}(E, D)$ defined as

$$C_{\mathrm{ord}}(E, D) = \sum_{I \in S} \|I - D(E(I))\|_2^2 , \tag{1}$$

where S, $E(\cdot)$, $D(\cdot)$, and $\|\cdot\|_p$ denote a set of inputs, the encoder, the decoder, and the ℓ^p norm, respectively.

To minimize $C_{\mathrm{ord}}(E, D)$, the decoder should be able to approximately restore an original vector I from its descriptor $E(I)$, which has a lower dimensionality than I. By training the encoder E and the decoder D by minimizing $C_{\mathrm{ord}}(E, D)$, information sufficient to restore an original vector can be extracted as a descriptor by the encoder. In this way, the auto-encoder method can construct descriptors of vectors from just a set of training vectors.

However, a descriptor of an image from an ordinary auto-encoder includes both a spatial pattern and its position.

If images have a common spatial pattern at different positions, their descriptors are different.

3. Transform invariant auto-encoder

We have proposed the transform invariant auto-encoder method [5]. It is trained by minimizing the following cost function;

$$\begin{aligned} C_{\mathrm{old}}(E, D) = \sum_{I \in S} \lambda_{\mathrm{inv}} \sum_i \|D(E(I)) - D(E(T_{\theta_i}(I)))\|_2^2 \\ + \lambda_{\mathrm{res}} \min_i \|D(E(I)) - T_{\theta_i}(I)\|_2^2 . \\ + \lambda_{\mathrm{spa}} \left(\frac{\|E(I)\|_1}{\|E(I)\|_2} \right)^2 , \end{aligned} \tag{2}$$

where S and T_θ denote a set of training inputs and a transform operator in the ignored transforms, respectively.

By minimizing the above cost function, we can achieve an auto-encoder that is transform invariant and can restore a pattern accurately. However, calculation of the function may require high computation cost for various transforms because the function includes minimization for the transform parameter θ.

4. Proposed method

We propose a new auto-encoder that separates an input into a transform invariant descriptor and transform parameters. The basic idea is relaxation of the minimization of the restoration term (the third term in (1)) for the transform invariant auto-encoder. To calculate the restoration term, it is required to find the transform parameter θ giving the minimum. However, it is generally difficult when a transform parameter is continuous and high-dimensional. So, we propose a method to avoid searching the concrete minimum on the whole transform parameter space by using a weight function, which indicates a transform parameter near to the minimum. The weight function can be used as a regressor of a transform parameter for an input. The weight function can be optimized simultaneously with the transform invariant encoder and the corresponding decoder.

4.1 Cost function

Searching the minimum can be replaced with optimization of the weight function $W(\theta)$ as follows;

$$\min_{\theta \in \Theta} f(\theta) = \min_{W(\theta) \geq 0, \int_\Theta W(\theta) d\theta = 1} \int_\Theta f(\theta) W(\theta) d\theta, \tag{3}$$

where

$$f(\theta) \stackrel{\mathrm{def}}{=} \|D(E(I)) - T_\theta(I)\|_2^2 . \tag{4}$$

If the integral in the right side of (3) is near to the minimum, the weight function $W(\theta)$ will have a value near to 1 on a small neighborhood of the minimum and almost zero otherwise. This means that the weight function indicates the parameter giving the minimum. Moreover, the weight function W can be optimized by gradient method even if it is difficult to differentiate $f(\theta)$ itself. In addition, the weight function $W(\theta)$ may depend on each input I. Therefore, the function W can be minimized simultaneously with the transform invariant encoder E and the corresponding decoder D.

By considering continuous parameters, we can rewrite the cost function for the transform invariant auto-encoder as following;

$$\sum_{I\in S}\lambda_{\text{inv}}\int_{\Theta}\left\|D\left(E\left(I\right)\right)-D\left(E\left(T_{\theta}\left(I\right)\right)\right)\right\|_2^2 d\theta + \lambda_{\text{res}}\min_{W}\int_{\Theta}\left\|D\left(E\left(I\right)\right)-T_{\theta}\left(I\right)\right\|_2^2 W(I,\theta)d\theta + \lambda_{\text{spa}}\left(\frac{\left\|E\left(I\right)\right\|_1}{\left\|E\left(I\right)\right\|_2}\right)^2, \quad (5)$$

where W is optimized under the condition that $0 \leq W(I,\theta)$ and $\int_{\Theta} W(\theta)d\theta = 1$. By optimizing W for the total value instead for only the restoration term, we can define a new cost function as following;

$$\begin{aligned} &C\left(E,D,W\right) \stackrel{\text{def}}{=} \\ &\sum_{I\in S}\lambda_{\text{inv}}\int_{\Theta}\left\|D\left(E\left(I\right)\right)-D\left(E\left(T_{\theta}\left(I\right)\right)\right)\right\|_2^2 d\theta \\ &+\lambda_{\text{res}}\int_{\Theta}\left\|D\left(E\left(I\right)\right)-T_{\theta}\left(I\right)\right\|_2^2 W(I,\theta)d\theta \\ &+\lambda_{\text{spa}}\left(\frac{\left\|E\left(I\right)\right\|_1}{\left\|E\left(I\right)\right\|_2}\right)^2. \end{aligned} \quad (6)$$

We train the transform invariant encoder E, the corresponding decoder D and the transform parameter weight function W so that they minimize the proposed cost function $C\left(E,D,W\right)$.

4.2 Calculation

For convenience of calculation, we suppose that the transform parameter weight function $W(I,\theta)$ is a Gaussian function on the transform parameter space as follows;

$$W\left(I,\theta\right) = \frac{1}{\left(2\pi\right)^{\frac{D}{2}}\left|\frac{1}{2}\Sigma_I\right|^{\frac{1}{2}}} e^{-\frac{1}{2}\left(\theta-\mu_I\right)^T\left(\frac{1}{2}\Sigma_I\right)^{-1}\left(\theta-\mu_I\right)}, \quad (7)$$

where D denotes the dimension of a transform parameter and Σ and μ denotes the scaled covariance matrix and the mean, respectively.

We use the Monte Carlo method to calculate integrals in the cost function (6). First, we define a utility function $w(I,\theta)$ as follows;

$$\begin{aligned} w\left(I,\theta\right) &\stackrel{\text{def}}{=} e^{-\frac{1}{2}\left(\theta-\mu_I\right)^T\Sigma_I^{-1}\left(\theta-\mu_I\right)}, \\ p_I\left(\theta\right) &\stackrel{\text{def}}{=} \frac{1}{\left(2\pi\right)^{\frac{D}{2}}\left|\Sigma_I\right|^{\frac{1}{2}}} e^{-\frac{1}{2}\left(\theta-\mu_I\right)^T\Sigma_I^{-1}\left(\theta-\mu_I\right)}, \end{aligned} \quad (8)$$

where p means a probability density function of a Gaussian distribution. $W\left(I,\theta\right)$ can be represented as

$$W\left(I,\theta\right) = 2^{\frac{D}{2}} w\left(I,\theta\right) p_I\left(\theta\right). \quad (9)$$

By using the utility functions, we can approximately calculate the restoration term, the second term in (6) as follows;

$$\begin{aligned} &\int_{\Theta}\left\|D\left(E\left(I\right)\right)-T_{\theta}\left(I\right)\right\|_2^2 W\left(I,\theta\right)d\theta \\ =&2^{\frac{D}{2}}\int_{\Theta}\left\|D\left(E\left(I\right)\right)-T_{\theta}\left(I\right)\right\|_2^2 w\left(I,\theta\right)p_I\left(\theta\right)d\theta \\ \approx&\frac{2^{\frac{D}{2}}}{N}\sum_{\theta_n\sim N_D\left(\mu_I,\Sigma_I\right)}\left\|D\left(E\left(I\right)\right)-T_{\theta_n}\left(I\right)\right\|_2^2 w\left(I,\theta_n\right), \end{aligned}$$

where $N_D\left(\mu_I,\Sigma_I\right)$ denotes the D-dimensional Gaussian distribution and N denotes the number of sampled parameters $\{\theta_n\}$. We minimize the cost function (6) by optimizing the parameters μ_I and Σ_I as functions of an input I. By using the Monte Carlo method, we can avoid combinational explosion when searching the minimum from whole possible transform parameters. Similarly, we can calculate the invariance term, the first term in (6), by the Monte Carlo method with the uniform distribution of possible transform parameters.

5. Experiments

We demonstrate the effectiveness of the proposed method by experiments with a shift invariant auto-encoder. On the experiments, we supposed that a transform parameter θ consisted of two values θ_x and θ_y and the shift operator T_θ was defined as

$$\left(T_{\theta}\left(I\right)\right)\left(x,y\right) = I(x+\theta_x, y+\theta_y), \quad (10)$$

where $I(x,y)$ denotes the value of the image I at the position (x,y). As a range of shifts, we supposed that $|\theta_x| \leq 4$ and $|\theta_y| \leq 4$.

As a transform invariant encoder, we used a neural network consisting of a single CNN with 9×9 filter kernels and 16-channel outputs following a max pooling with stride 2 and a three-layer fully connected neural network (NN), where each layer has 1500, 150, 30 outputs respectively. As a decoder corresponding to the encoder, we used a three-layer fully connected NN, where each layer has 150, 1500, 1024 outputs, respectively. As a regressor of μ_I and Σ_I, which are parameters of a transform parameter weight function, we used a four-layer fully connected NN, where each layer has 256, 64, 16 and 5 outputs, respectively. The 2 outputs of the final 5 outputs are used as μ_I and the rest 3 outputs are used for generating Σ_I. In addition, we used a hyperbolic tangent as an activation function, which is placed between each pair of layers.

Here, we demonstrate shift invariant property of the proposed method using experiments for digit patterns.

We generated two auto-encoders. One was trained as a shift invariant auto-encoder by minimizing (2) and the other was trained as a proposed auto-encoder by minimizing (6) for digit images of training images in the MNIST database [3]. Both auto-encoders were trained by stochastic gradient descent (SGD) [3] with learning rate 1.0×10^{-3}, and both were updated with every 100 samples that were randomly extracted from the training images (60k samples) in the MNIST database. We used auto-encoders that were updated 20,000 times ($\approx$ 33 epochs). Training the shift invariant auto-encoder took more than 6 and a half hours and training the proposed auto-encoder took a little less than 6 hours. The proposed auto-encoder could be trained in less time, even though it includes a regressor of transform parameters in addition to an encoder of transform invariant components.

As an example, we encoded and decoded an test image of

the digit "0", which is not used in training auto-encoders. Input images are shown in Fig. 4, where the center image is the original image in the MNIST database and the others are its shifted versions. Images in Fig. 5 are restored from images in Fig. 4 by the shift invariant auto-encoder. Shift invariant image components restored by the proposed auto-encoder are shown in Fig. 6. Fig. 7 shows the restored images which are shifted according to μ_I, the estimated shift parameters.

The restored images in Fig. 5 have similar shapes on similar positions, though they are restored from inputs with different shifts. The images in Fig. 6 are also almost similar to each other. This means that the proposed auto-encoder can extract a pattern itself without regard to its position. The images in Fig. 7 have similar shapes and positions to those in Fig. 4. This means that the regressor in the proposed auto-encoder successfully estimate positions of patterns.

6. Conclusion

We proposed a novel auto-encoder that separated an input into a transform invariant descriptor and transform parameters. By utilizing a transform parameter weight function and the Monte Carlo method, we can avoid a problem of combinational explosion when training a proposed auto-encoder for various transforms. By an experiment, we showed that they can encode a pattern independently of its position and a background, respectively.

The framework of the proposed cost function can be applied to temporal patterns and other transforms such as dilation and rotation. The proposed auto-encoder will be able to independently encode typical motions in a video without regard to dilation and rotation. This will be useful for motion-based recognition.

References

[1] Baldi, P. and Hornik, K.: Neural networks and principal component analysis: Learning from examples without local minima, *Neural Networks*, Vol. 2, No. 1, pp. 53 – 58 (1989).

[2] Hinton, G. E. and Salakhutdinov, R. R.: Reducing the Dimensionality of Data with Neural Networks, *Science*, Vol. 313, No. 5786, pp. 504–507 (2006).

[3] Lecun, Y., Bottou, L., Bengio, Y. and Haffner, P.: Gradient-based learning applied to document recognition, *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2278–2324 (1998).

[4] Makhzani, A. and Frey, B. J.: k-Sparse Autoencoders, *CoRR*, Vol. abs/1312.5663 (2013).

[5] Matsuo, T., Fukuhara, H. and Shimada, N.: Transform invariant auto-encoder, *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, September 24-28, 2017*, IEEE, pp. 2359–2364 (2017).

[6] Vincent, P., Larochelle, H., Bengio, Y. and Manzagol, P.-A.: Extracting and Composing Robust Features with Denoising Autoencoders, *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, New York, NY, USA, ACM, pp. 1096–1103 (2008).

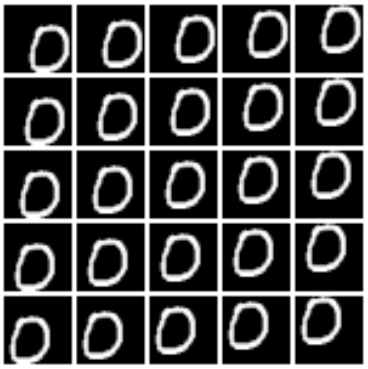

Fig. 4 An image in MNIST and its shifted versions

Fig. 5 Images restored by a shift invariant auto-encoder

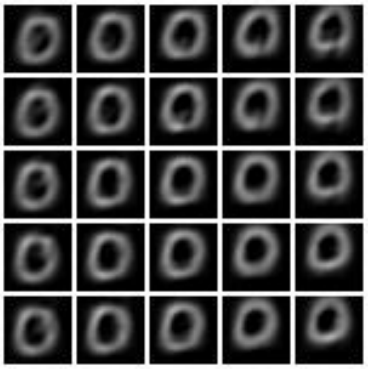

Fig. 6 Shift invariant images restored by a proposed auto-encoder

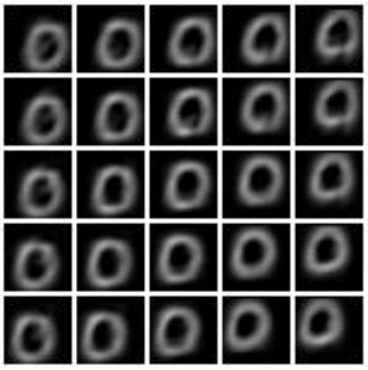

Fig. 7 Images restored by a proposed auto-encoder