Feature Article

Anomaly detection with autoencoder

Hidetaka Nambo (Kanazawa University)

1. Introduction

With the recent development of AI technology, especially machine learning technology such as deep learning, the application of AI in various situations has been studied and many practical examples have been reported. In particular, the improvement of accuracy in image classification and detection has been remarkable. Since AlexNet [(1)], which uses convolutional neural networks, won the ILCVRC with high accuracy in 2012, higher-performance algorithms have been developed one after another, and the accuracy has also been improved.

On the other hand, AI technologies are also being introduced in the field of anomaly detection. For the problem of anomaly detection, because the occurrence of anomalies is rare, it is difficult in terms of collecting data of anomalies. Therefore, it is not possible to apply the method of creating a classification model by collecting and learning data for both normal and anomaly conditions which is usually applied for image classification problems. For this reason, generative deep learning algorithms are used usually.

In this article, an autoencoder [(2)], one of the deep generative networks, is explained shortly, and an anomaly detection method using autoencoders is introduced.

2. Algorithms for generative systems

Various deep learning algorithms for generative systems have been proposed.

Generative Adversarial Network (GAN) [(3)] is one of the generative algorithms. Images generated by GAN or variants of GAN are indistinguishable from real objects. In addition to generating images, there are other types of GANs, such as StackGAN [(4)], which generates images from sentences with little information and learned content.

Autoencoder is a method that was proposed before GAN, and it extracts and learns features to reproduce the target. Also, it can be used as a generative algorithm by using the part to be reproduced for generation. The next section explains details.

3. Autoencoder

An autoencoder is a form of neural network that consists of an encoder part that encodes the input and extracts features, and a decoder part that decodes the features and generates the output. A structure of an autoencoder is shown in Figure 1.

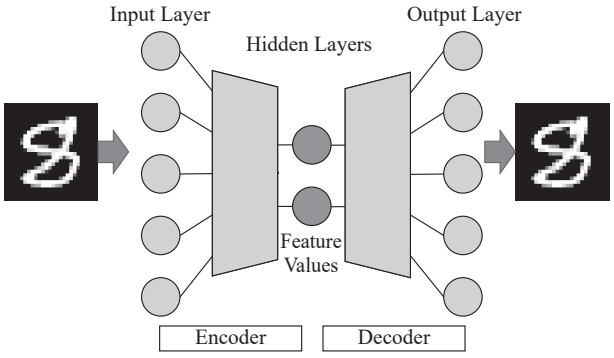


Figure 1: Structure of autoencoder

In learning an autoencoder, it is done so that the input and output are the same. This means that the features extracted from the input will be sufficient elements to reproduce the input as the output. Therefore, the autoencoder also has the aspect of a dimension reduction to represent the input with fewer dimensions.

Various forms have been proposed for the encoder and decoder parts. Stacked autoencoders, which consist of multiple layers like conventional neural networks, and convolutional autoencoders, which use convolutional layers, have been proposed. Other types such as a variational autoencoder (VAE) [(5)] and a conditional variational autoencoder (CVAE) [(6)] have also been proposed.

4. Data generation with autoencoder

Section 3 explained how the autoencoder compresses the input and encodes the features.

The decoder part can also generate the input data based on the features. Then, it is possible to consider giving arbitrary

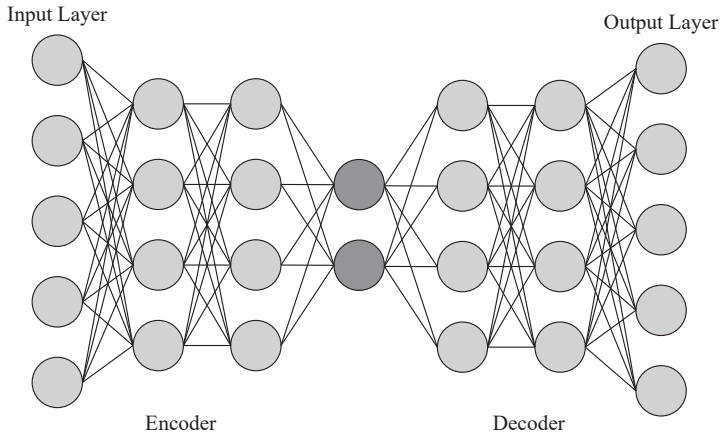


Figure 2: Structure of a stacked autoencoder

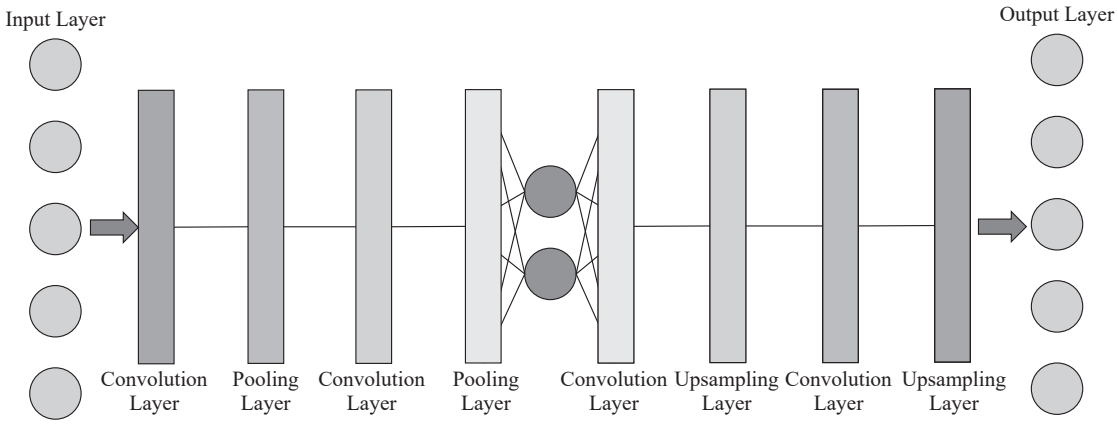


Figure 3: Structure of a convolutional autoencoder

features to the already trained autoencoder instead of the features encoded from the input. Since the decoder has already been trained, it can generate an image similar to the input image from the given features, as long as it does not deviate significantly from the range of features of input as training data.

In the following example, MNST dataset is used for a training of the autoencoder. The dataset is a dataset of handwritten digit images, where each image is represented by 28 × 28 pixels in 256 degree of gray-scale. Figure 4 shows a part of MNIST dataset.

Figure 4: Part of MNIST dataset
Source: Wikipedia, 2021.[7]

A simple stacked autoencoder shown in Figure 5 is used to learn the '0' and '1' images of MNIST dataset. Figure 5 shows the structure of the autoencoder used in this example.

As a result of the training, the image of '0' or '1' can be represented by two-dimensional features, because the hidden layer in Figure 5 consists from 2 nodes. Figure 6 shows the distribution of features when the image used for training is encoded by the encoder. Each image is converted into two-dimensional features. In the figure, it can be seen that the feature distribution is divided into two clusters. One is the cluster transformed

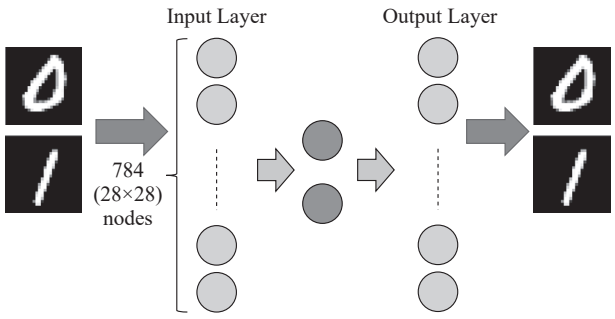


Figure 5: Structure of autoencoder in the example

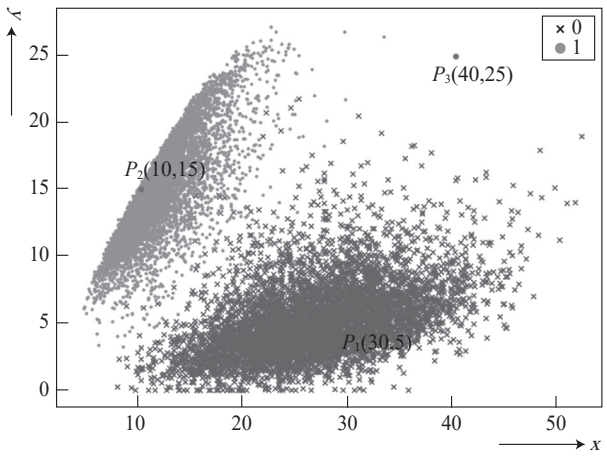
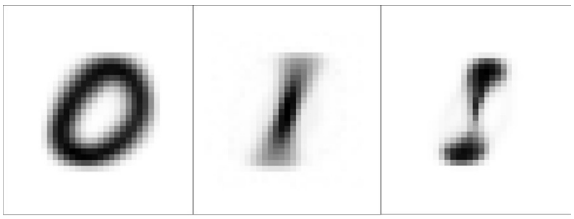


Figure 6: A feature distribution of digits of '0' and '1'

from the '0' image, and the other is the cluster transformed from the '1' image.

By determining feature values arbitrary, it is possible to generate images using the decoder part of the autoencode that has been trained. The image generated from the three points P_1(30,5), P_2(10,15) and P_3(40,25) in Figure 6 is shown in Figure 7. Figure 7 (a)-(c) shows a generated image from feature values of P_1, P_2 and P_3, respectively. The closer the features are to the '0' and '1' clusters, the more similar the respective images are generated. It can also be seen that for features that do not belong to either of the clusters, an image similar to neither of the images is generated.



(a) (b) (c)

Figure 7: Generated images from feature value P1, P2 and P3

Thus, it can be seen that it is possible to generate an image from any value in the feature space. It can also be seen that it is possible to arbitrarily generate images that are similar or dissimilar to the learned image, depending on the feature values used.

5. Anomaly detection with autoencoder

By using the feature of image generation, it is possible to detect anomalies by learning only normal images.

In anomaly detection, the problem in training is that there is not enough data of anomaly states. Therefore, we learn only the normal state and detect that the anomaly state is different from the normal state.

Figure 8 shows the generated images when images of '0' to '9' are input to the network trained in section 4. In Figure 8, pairs of images in rectangles represent an input and a generated image. The left image is an input image and the right one is a generated image. When an image of '0' or '1' is input, an image similar to the input is generated. However, since images other than '0' or '1' have not been learned, an image similar to the input is not generated, and an image similar to '0' or '1' is generated. Here, if '0' and '1' are considered as normal states and the rest as anomaly states, we can regard that the difference between the input and output becomes larger when the anomaly state is input to the autoencoder trained only normal states. Therefore, by calculating the difference between input and generated images, and judging the difference to be anomaly if it exceeds a certain level, it becomes possible to detect anomalies.

Figure 8: Input images and generated images from input

In this example of autoencoder, the output was generated from features by encoding the anomaly image. However, instead of encoding the input, searching in the feature space to generate an image similar to the input may be used in some other method. In such cases, the time required for the search may become a problem.

6. Conclusion

In this article, an anomaly detection method using deep generative network was introduced. In particular, the simplest structure of autoencoder and MNIST dataset was used as a case study to explain the essentials of the autoencoder-based anomaly detection method. This method can be used not only for detecting anomalies in images, but also for detecting anomalous values and outliers in general data. It may be able to be used to generate and use data other than the observed data by complementing the features of the existing data.

This autoencoder is very rudimentary. Practically, it is necessary to consider using a more sophisticated VAE, CVAE or

GAN and their variants.

Notes

(1) Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, Vol. 60, No. 6, 84-90.

(2) Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, Vol. 313, No. 5786, 504-507.

(3) Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial networks. arXiv:1406.2661.

(4) Zhang, H., Xu, T., Li, H., Zhang, H., Wang, X., Huang, X., and Metaxas, D. (2016). StackGAN: Text to photo-realistic image synthesis with stacked generative adversarial networks. arXiv:1612.03242.

(5) Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. arXiv:1312.6114.

(6) Kingma, D. P., Rezende, D. J., Mohamed, S., and Welling, M. (2014). Semi-supervised learning with deep generative models. arXiv:1406.5298.

(7) Wikipedia (2021). MNIST database (Retrieved April 25, 2021 from https://en.wikipedia.org/wiki/MNIST_database).