Autoencoder: Issues, Challenges and Future Prospect

Anega Maheshwari, Priyanka Mitra, and Bhavna Sharma

Abstract As of more recently, deep learning-based models have demonstrated considerable potential, as they have outperformed all traditional practices. When data becomes high dimensional, extraction of features and compression of data become progressively significant. In this paper, we describe the autoencoder deep learning algorithm. Autoencoder is primarily a neural network-based feature extraction methodology that accomplishes outstanding victory in producing highlights of high-dimensional data. Autoencoder assumes a principal job in unsupervised learning which targets to rework inputs into outputs with minimal reconstruction error.

Keywords Autoencoder · Convolutional autoencoder · Denoising autoencoder · Unsupervised learning

1 Introduction

While unaided learning of a mapping that generates "good" intermediary illustrations of the input appears to be the key, little is comprehended in regards to what comprises "good" illustrations for initializing deeper network architectures, or what unequivocal criteria may direct learning such intermediary illustrations. Gaining insights from an expansive sum of information could be a difficult job. Several dimensionality reduction strategies that are all around concentrated in the writing intend to acquire an understanding of an easier representation from the dataset.

Although numerous methods have been proposed to deal with dimensionality reduction, outcomes from models dependent on deep architectures are encouraging. Autoencoder has been taken to the forefront of generative modeling with the emergence of deep learning science. Autoencoders are basic learning systems that intend

A. Maheshwari · P. Mitra (✉) · B. Sharma
Department of Computer Science, Jaipur Engineering College and Research Centre, Jaipur, Rajasthan, India
e-mail: priyankamitra.cse@jecrc.ac.in

A. Maheshwari
e-mail: anegamaheshwari.cse20@jecrc.ac.in

M. Vashista et al. (eds.), *Recent Innovations in Mechanical Engineering*,
Lecture Notes in Mechanical Engineering, https://doi.org/10.1007/978-981-16-9236-9_24

to recast inputs into outputs with the slightest possible distortion. The notion of autoencoders has been widespread within the discipline of neural networks for a considerable length of time.

However, recently, the idea of autoencoders has been extensively utilized for mastering generative models of data. Autoencoders are commonly trained with a solo layer encoder and a solo layer decoder; however, utilizing deep autoencoders offers favorable points of interest.

In this paper, we proposed an approach to an autoencoder model which can extract features based on both data itself and the correlation among the data. Provided data relationships, reconstruction error can be diminished and, therefore, generate more powerful features. This paper presents a comprehensive survey on autoencoder architecture and its various variants. The various issues and challenges have been addressed for autoencoders.

We organize the paper as described. Section II describes the literature study of autoencoders. Section III begins with a simple autoencoder architecture and its basic components. Variants of autoencoder are explained in Section IV. Section V presents challenges and issues of autoencoders. Section VI presents conclusions and future work.

2 Literature Review

Autoencoders grasp a condensed representation of data by mapping the data into a smaller spatial dimension. The foremost precept of autoencoder follows from the name: "auto" favors that this technique is unsupervised and "encoder" signifies it learns encodings of data. Autoassociator or diabolo network are other names of autoencoder. In particular, encoded latent space features of data are learned by an autoencoder, which tries to limit the error between original data and output decoded from the encodings. The latent space representation carries fundamental characteristics of data.

In [1], autoencoder was developed as a neural network tool and that could be a successful solution for unsupervised learning using neural networks. This was presented as a strong replacement for the current approaches that were conducting a function close to that of principal component analysis (PCA) at the time. An analytical model [2] has been developed based on the minimal description duration (MDL) concept for the training of autoencoders. Their purpose was to lessen the knowledge needed to describe both the vector of code vector and the error in reconstruction.

In [3], the authors suggested the usage of deep autoencoder as a promising method for minimizing dimensionality. Experimentally, they demonstrated that the compression of deep autoencoders is significantly stronger than the equivalent shallow or linear autoencoders. In [4], the authors defined a novel algorithm for learning sparse representations and compared it with a related probabilistically trained machine, namely restricted Boltzmann machine (RBM), theoretically and experimentally. In [5], autoencoder is proposed as a means of directing supervised learning by the usage

of autoencoding in conjunction with supervised learning as a tuning denoising mechanism. The authors discussed a deep networking approach focused on piling layers of denoising autoencoders that were trained to denoise noisy forms of the inputs. This prompted them to address a conceptual flaw of conventional autoencoders, namely their failure to understand effective overcomplete code. The resulting algorithm of stacked denoising autoencoder for deep network training has proven capable of bridging the functional gap with deep belief networks, resulting in equal or finer classification results.

In [6], the authors demonstrated that a gradient of an autoencoder offers estimation to restricted Boltzmann machines' analogous divergence preparation. The authors in [7] proposed an innovative method to train deterministic autoencoders. They demonstrated that by applying a well-selected penalty term to the cost function of reconstruction, results can be obtained that equal or outclass those produced by denoising autoencoders and other regularized autoencoders on a number of datasets. In addition, they demonstrated that this penalty term has connection with both regularized and denoising autoencoders, and it can be interpreted as a bridge between deterministic and non-deterministic autoencoders.

In [8], a very efficient sparse coding system named k-sparse autoencoder was introduced by Makhzani and Frey, which attains good sparsity in the hidden representation. They also addressed how to use the k-sparse autoencoder for pretraining architectures that are shallow and deep. Bowman et al. [9] developed and analyzed a generative model focused on RNN variational autoencoder, which implements distributed latent space representations of complete sentences and can be utilized for natural language sentences. In [10], the authors described autoencoders and its variants and introduced a deep neural generative model that would merge variational autoencoders (VAEs) with comprehensive attribute discriminators to efficiently implement semantic structures.

In [11], the authors used deep autoencoders and feedforward networks for anomaly detection. NSL-KDD dataset has been considered for testing and comparing of the models.

In the literature, autoencoders have been utilized for information retrieval and dimensionality reduction. Backpropagation learning is employed for updating the weights within the network. Optimization algorithms such as stochastic gradient descent, root mean square prop, and Adam are employed for the learning of autoencoders.

The literature review recommends that autoencoders are acceptable candidates to find out spatial and temporal features and determine anomalies. There have also been numerous experiments undertaken on current state-of-the-art deep learning architectures present in the literature. Table 1 presents preliminary prominent contributions to the proposed architecture. To sum up the literature review concerning autoencoders, the accompanying table is offered for ease of access:

Table 1 Literature review on autoencoders

Contribution	Papers
Autoencoders	Hinton and Zemel (1994)
Denoising autoencoders	Gallinari et al. (1987), Vincent and Larochelle (2008)
Convolutional autoencoders	Erhan et al. (2010), Masci et al. [7], Du et al. (2017)
Sparse autoencoders	Olshausen and Field (1997), Ranzato (2007), Mairal et al. (2009), Makhzani and Frey (2014)
Stacked autoencoders	Bengio et al. (2007), Vincent et al. (2010)
Contractive autoencoders	Rifai et al. (2011)

3 Architecture

This section begins with a description of the simple autoencoder architecture followed by its basic components.

Basic Autoencoder and its Components. An autoencoder is a class of neural networks that attempts to recreate the output relative to the input by estimating the identity function. To attenuate the reconstruction error which can be evaluated using loss functions, the model parameters are optimized.

Autoencoder [3] focuses on learning a compact and distributed representation for a collection of data. Using a fewer hidden units than inputs powers the autoencoder to memorize a condensed approximation.

Autoencoder comprises of three main components: an encoder network, a code and a decoder network. The encoder encodes the input image as a compressed representation in a reduced dimension; the code resembles the latent space encoding, i.e., the compressed input fed to the decoder; the decoder rebuilds input back to the initial dimension from the code (Fig. 1).

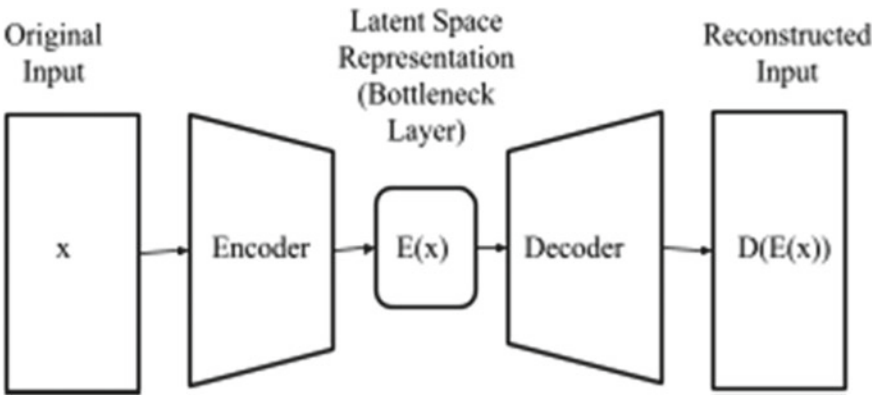

Fig. 1 Basic components of an auto encoder

When the size of the hidden layer in a basic autoencoder is larger than the size of the input layer, this is considered an overcomplete autoencoder. When the size of the hidden layer in a basic auto encoder is less than the size of the input layer, it is considered an undercomplete autoencoder.

An autoencoder makes use of a series of "recognition weights" to transform a vector of input to a vector of code. It then makes use of a collection of "generative weights" to transform the vector of code into an estimated reconstruction of the vector of input.

The autoencoder calculation and its profound variant has been a remarkable accomplishment as of late. An autoencoder's training procedure is based on cost-function optimization. The autoencoders are nonlinear by nature and can learn more intricate relations and more powerful features.

4 Variants of Autoencoder

Denoising Autoencoder. The basic principle behind denoising autoencoder (DAE) is to compel the autoencoders to no longer gain proficiency from the identity function, but more robust features, by reestablishing the input from a corrupted version of itself. An autoencoder can retrace a distorted input by capturing the statistical dependencies between the inputs.

Training denoising autoencoders requires learning to recover clean input from an adulterated sample, a task called denoising. Denoising autoencoders anticipate that well denoising can be accomplished if the model obtains highlights that seize valuable structure in the input distribution.

Denoising autoencoder's key idea is to corrupt a portion of the input characteristics of a given dataset X before submitting it to an autoencoder model, train the network to rebuild a fair "restored" input from the adulterated input X' and then reduces the error between the reconstructed Y and original X (Fig. 2).

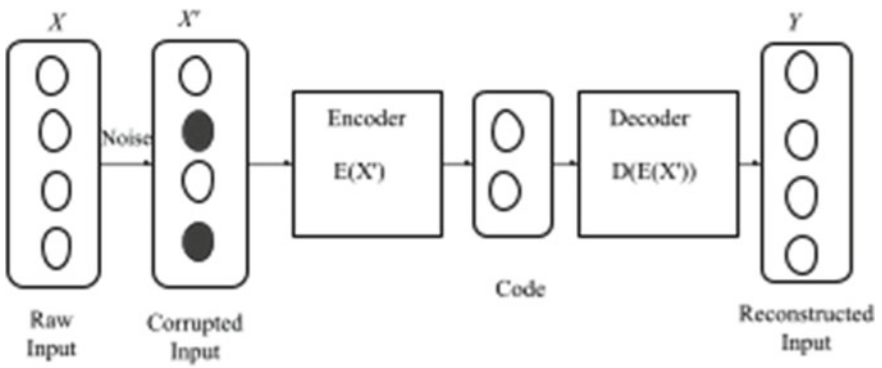

Fig. 2 Denoising autoencoder

The training operation of a denoising autoencoder works as follows:

- Stochastic mapping is bring into play to contaminate the initial input X into X'.
- The corrupted input X' is then mapped to a hidden encoding by using function $E(X')$.
- From the hidden encoding, the model reestablishes the input through mapping $D(E(X'))$.

The loss function minimizes error not from the original but the corrupted input and takes the following form:

$$L\big(X, D\big(E\big(X'\big)\big)\big) \tag{1}$$

where $D(E(X'))$ is the decoder output, $E(X')$ is the encoder output encoded from the corrupted input X'.

Convolutional Autoencoder. Convolutional autoencoders (CAEs) are focused on convolutional neural networks (CNN). A CNN comprises of convolutional, pooling, and deconvolutional layers optionally accompanied by a fully connected layer. The convolutional operation is applied by extracting native receptive fields around the entire image to create an activity map from the inputs. Each ensuing layer increases the intricacy of the realized activity map. This activity map is also known as a feature map. The middle subsampling layer is connected to the decoder in unsupervised learning and then proceeds to restore the input image again with the convolutional process.

The convolution operator permits filtering an input signal to extract a considerable phase of its content. Convolutional autoencoders utilize the convolution operator to take advantage of the survey that a signal can be seen as an entirety of other signals. Convolutional autoencoders figure out how to encipher the contribution of inputs to a set of basic signals and afterward attempt to remake the contribution from those signals. We let the convolutional autoencoder model gain proficiency with optimal filters that lessen the error in the reconstruction. Once learned, these filters are applied to related input to retrieve information.

CAEs are a subset of CNNs: the key distinction between the frequent understanding of CNN and CAE is that the former are qualified end-to-end to master filters and integrate features to categorize their input. In addition, CNNs are generally classified as supervised learning algorithms [6]. CAEs are only equipped to comprehend optimal filters that can extract features and can be used for input recreation.

Convolutional autoencoder is a conventional autoencoder stacked with convolution layers. Convolutional autoencoder broadens the standard autoencoder's fundamental structure by altering the fully connected layers to convolution layers. The encoder network is modified to convolutional layers and the decoder network is modified to transpose convolutional layers in convolutional autoencoders. In the event that we use image data, the convolutional layer is smarter to catch the spatial statistics in the image (Fig. 3).

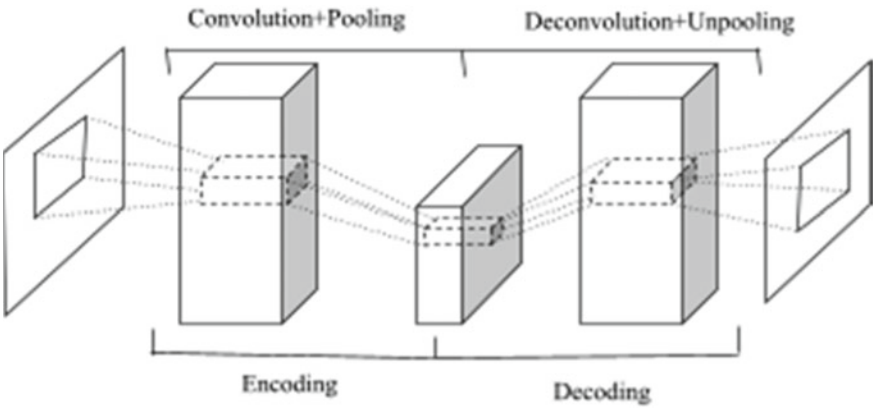

Fig. 3 Convolutional autoencoder

As opposed to utilizing one hidden layer in a basic autoencoder, a convolutional autoencoder uses numerous layers to extract significantly high-level features appearing in the image. The quantity of unbound parameters in the fully connected layer is bigger than the quantity of unbound parameters in a multi-convolution layer, which makes the basic autoencoder arduous to train and time exorbitant.

Convolutional autoencoders can scale rigorously to realistic-sized high-dimensional images as a result of their convolutional character. Therefore, convolutional autoencoders are well acknowledged as "Feature Extractors" for general purpose functionality.

Sparse Autoencoder. A sparse autoencoder (SAE) incorporates additional hidden activation nodes than input nodes, but only a few nodes are dynamically active at once. The loss function penalizes the activation nodes present in the hidden layers, with the end goal that only a few nodes are actuated when the network is trained.

It is considered that if a sparsity penalty is upheld on the hidden layer activation nodes, at that point, the autoencoder would still discover captivating designs and statistical patterns within the data. The sparsity constraint permits the sparse autoencoder to figure out distinctive characteristics and features within the dataset. By enforcing constraint on a number of hidden layer activations, one can flatten the input to an optimally contracted latent space.

Sparse autoencoder is mathematically constrained, such limitations are added to the learning optimization handles using regularizers. Sparsity is extended to autoencoders as a mathematical constraint by means of several conceptualizations. It has been noted that sparse regularization prompts higher learning performance in many applications by improving the standard of encoding.

The network is driven to acquire representations in which only a minimal number of neurons are triggered. There are two different ways to impose sparsity constraint:

- L1 Regularization—It adds "absolute value of magnitude" of coefficients as a penalty term. A term can be added to the loss function that penalizes the absolute value of the vector of activations within the hidden layers.
- Kullback Leibler Divergence—It is the degree of contrast between two probability distributions. A sparsity parameter ρ is defined, which indicates the average activation of a node.

The k-sparse autoencoder [8] is an improvement over sparse autoencoder. Here, k nodes with the most elevated activation functions are picked.

5 Challenges and Issues

When addressing real-world data, existing algorithms dependent on autoencoders experience the ill effects of various issues which hinder their robustness and ease-to-use, such as

Weights Initialization—with huge introductory weights, autoencoders usually realize poor local minima; with tiny initial weights, the gradients within the early layers are little, creating it infeasible to train autoencoders with many hidden layers. Instantiating the weights with random values can add randomness to the obtained results.

Model Architecture—the model's configuration, i.e., number of layers and their width, causes the network to seek a specific portrayal of the data while retaining the relevant details.

Hyperparameters—there are a few essential hyperparameters that are tough to be set; learning rate, weight-cost, dropout fraction, batch size, the number of epochs, the number of layers, the number of nodes in each of the encoding layers, kind of activation functions, number of nodes in each of the decoder layers, network weight initialization, optimization algorithms, and the number of nodes in the bottleneck layer.

Studying autoencoders in depth empowers one to achieve a general understanding of autoencoders, characterize key properties that are shared by different autoencoders and that should be checked consistently in any new form of autoencoder. The utilization of autoencoder encodings might offer a decent solution for errands that require more steady performance under the noise.

The key drawback of the autoencoders lies in the fact that they are data-driven or data-specific, and thus, their utility is restricted to the data that share similar properties with the training data.

Another principal constraint is that the compression by an autoencoder is lossy. Autoencoders do not flawlessly restore the original information. This essentially implies that compression and decompression operation degrades the output of the network, generating a less precise representation contrasted with its input. It should be noted that autoencoders do not perform noticeably better than the JPEG algorithm at encoding images.

6 Conclusion and Future Work

Autoencoders are broadly utilized for a diversity of tasks such as information retrieval, reverse image search, classification, anomaly detection, dimensionality reduction, image compression, image denoising, image blending, feature extraction, and plenty more. A smart idea for future work is to apply autoencoders in medical image analysis. Medical imaging including X-rays, MRI, CT, etc., are susceptible to noise. Reasons incorporate the use of different image acquisition techniques. Image denoising is a crucial preprocessing step in medical image analysis. Denoising autoencoders can be utilized here for efficient denoising of medical images.

Another potential future application is an encoder–decoder model capable of capturing temporal structure, such as long short-term memory (LSTM)-based autoencoders, which can fix machine translation issues. It can be used to determine a video's next frame. Besides the simple autoencoder, denoising autoencoder and convolutional autoencoder; variational autoencoder (VAE) is another autoencoder variant that is worth investigating. It would be fascinating to explore its potential in future work.

In short, autoencoders create a representation at the intermediate bottleneck layer to preserve only the components that are useful, and to reject not useful segments and noise. In the modern era, autoencoders are turning up to be a hot field of research and exploration in numerous aspects. This paper has successfully introduced autoencoders driven by the objective of learning intermediate representations of the input that are robust to small irrelevant changes in the input and much better suited for subsequent learning tasks.

References

1. Hornik K, Baldi P (1989) Neural networks and principal component analysis: learning from examples without local minima. Neural Netw 1(2):54
2. Hinton G (1990) Connectionist learning procedures. In: Machine learning. Morgan Kaufmann, pp 555–610
3. Hinton GE, Osindero S, Teh YW (2006) A fast learning algorithm for deep belief nets. Neural Comput 18(7):1527–1554
4. Ranzato M, Boureau Y, LeCun Y (2007) Sparse feature learning for deep belief network. In: Advances in neural information processing system, vol 20 (NIPS 2007)
5. Vincent P, Larochelle H, Bengio Y, Manzagol P-A (2008) Extracting and composing robust features with denoising autoencoders (Technical report 1316). Université de Montréal, Department IRO
6. Bengio Y (2009) Learning deep architectures for AI. Found Trends Mach Learn 2(1):1–127
7. Masci J et al (2011) Stacked convolutional auto-encoders for hierarchical feature extraction. In: Artificial neural networks and machine learning–ICANN 2011, pp 52–59
8. Makhzani A, Frey B (2013) k-sparse autoencoders
9. Bowman SR, Vilnis L, Vinyals O et al (2016) Generating sentences from a continuous space. In: CONLL, pp 10–21

10. Zhai J, Zhang S, Chen J, He Q (2018) Autoencoder and its various variants. In: 2018 IEEE international conference on systems, man, and cybernetics (SMC), Miyazaki, Japan, 2018, pp 415–419. https://doi.org/10.1109/SMC.2018.00080
11. Albahar MA, Binsawad M (2020) Deep autoencoders and feedforward networks based on a new regularization for anomaly detection. Security Commun Netw