RIZVI COLLEGE OF ENGINEERING

DEEP LEARNING

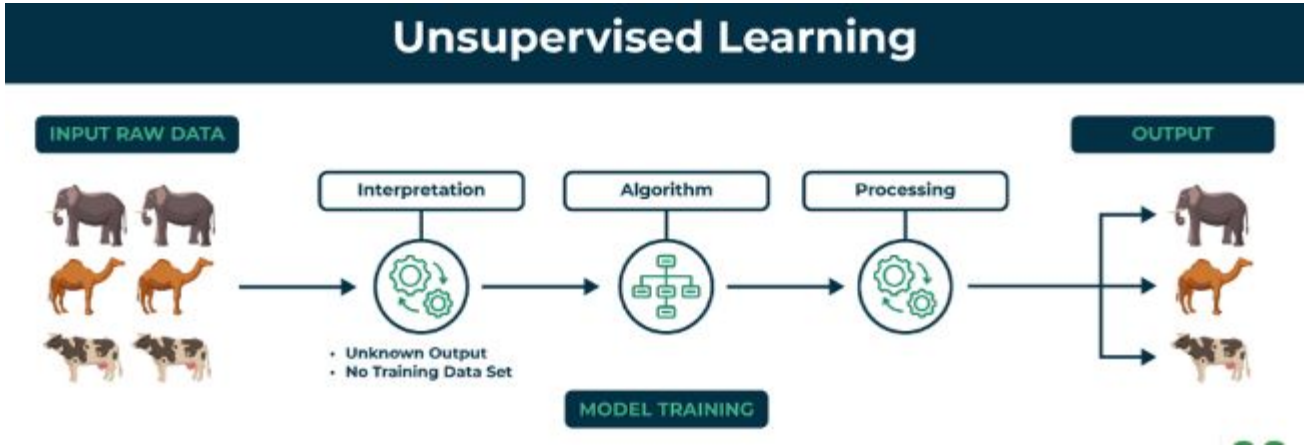
MODULE 3

TOPICS

Module 3		Autoencoders: Unsupervised Learning
	3.1	Introduction, Linear Autoencoder, Undercomplete Autoencoders, Overcomplete Autoencoders. Regularization in Autoencoders
	3.2	Denoising Autoencoders, Sparse Autoencoders, Contractive Autoencoders
	3.3	Application of Autoencoders: Image Compression

UNSUPERVISED LEARNING

- *Unsupervised learning* is a type of machine learning in which models are trained using unlabeled dataset and are allowed to act on that data without any supervision. Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data. The goal of unsupervised learning is to find the underlying structure of dataset, group that data according to similarities, and represent that dataset in a compressed format.
- Example: Dataset input of images of cats and dogs. Unsupervised learning algorithm will perform this task by clustering the image dataset into the groups according to similarities between images.
- Input raw data Interpretation Suitable algorithm Divide data in groups Output.



WHAT IS UNSUPERVISED LEARNING?

- Unsupervised learning is a type of machine learning that learns from unlabeled data. This means that the data does not have any pre-existing labels or categories. The goal of unsupervised learning is to discover patterns and relationships in the data without any explicit guidance.
- Unsupervised learning is the training of a machine using information that is neither classified nor labeled and allowing the algorithm to act on that information without guidance. Here the task of the machine is to group unsorted information according to similarities, patterns, and differences without any prior training of data.
- Unlike supervised learning, no teacher is provided that means no training will be given to the machine. Therefore the machine is restricted to find the hidden structure in unlabeled data by itself.
- You can use unsupervised learning to examine the animal data that has been gathered and distinguish between several groups according to the traits and actions of the animals. These groupings might correspond to various animal species, providing you to categorize the creatures without depending on labels that already exist.

KEY POINTS

- Unsupervised learning allows the model to discover patterns and relationships in unlabeled data.
- Clustering algorithms group similar data points together based on their inherent characteristics.
- Feature extraction captures essential information from the data, enabling the model to make meaningful distinctions.
- Label association assigns categories to the clusters based on the extracted patterns and characteristics.

Example:

- Imagine you have a machine learning model trained on a large dataset of unlabeled images, containing both dogs and cats. The model has never seen an image of a dog or cat before, and it has no pre-existing labels or categories for these animals. Your task is to use unsupervised learning to identify the dogs and cats in a new, unseen image.
- For instance, suppose it is given an image having both dogs and cats which it has never seen.
- Thus the machine has no idea about the features of dogs and cats so we can't categorize it as 'dogs and cats '. But it can categorize them according to their similarities, patterns, and differences, i.e., we can easily categorize the above picture into two parts. The first may contain all pics having dogs in them and the second part may contain all pics having cats in them. Here you didn't learn anything before, which means no training data or examples.
- It allows the model to work on its own to discover patterns and information that was previously undetected. It mainly deals with unlabelled data.

UNSUPERVISED LEARNING can be further divided into two types of problems:

- **Clustering**: Clustering is a method of grouping the objects into clusters such that objects with most similarities remains into a group and has less or no similarities with the objects of another group. Cluster analysis finds the common factors between the data objects and categorizes them as per the presence and absence of those factors.
- **Association**: An association rule is an unsupervised learning method which is used for finding the relationships between variables in the large database. It determines the set of items that occurs together in the dataset. Association rule makes marketing strategy more effective. Ex: People who buy X item are also tend to purchase Y item. A typical example of Association rule is Market Basket Analysis.

Application of Unsupervised learning

- Non-supervised learning can be used to solve a wide variety of problems, including:
- Anomaly detection: Unsupervised learning can identify unusual patterns or deviations from normal behavior in data, enabling the detection of fraud, intrusion, or system failures.
- Scientific discovery: Unsupervised learning can uncover hidden relationships and patterns in scientific data, leading to new hypotheses and insights in various scientific fields.
- Recommendation systems: Unsupervised learning can identify patterns and similarities in user behavior and preferences to recommend products, movies, or music that align with their interests.
- Customer segmentation: Unsupervised learning can identify groups of customers with similar characteristics, allowing businesses to target marketing campaigns and improve customer service more effectively.
- Image analysis: Unsupervised learning can group images based on their content, facilitating tasks such as image classification, object detection, and image retrieval.

Advantages of Unsupervised learning

- It does not require training data to be labeled.
- Dimensionality reduction can be easily accomplished using unsupervised learning.
- Capable of finding previously unknown patterns in data.
- Unsupervised learning can help you gain insights from unlabeled data that you might not have been able to get otherwise.
- Unsupervised learning is good at finding patterns and relationships in data without being told what to look for. This can help you learn new things about your data.

SUPERVISED VS. UNSUPERVISED MACHINE LEARNING

Parameters	Supervised machine learning	Unsupervised machine learning
Input Data	Algorithms are trained using labeled data.	Algorithms are used against data that is not labeled
Computational Complexity	Simpler method	Computationally complex
Accuracy	Highly accurate	Less accurate
No. of classes	No. of classes is known	No. of classes is not known
Data Analysis	Uses offline analysis	Uses real-time analysis of data
Algorithms used	Linear and Logistics regression, Random forest, Support Vector Machine, Neural Network, etc.	K-Means clustering, Hierarchical clustering, Apriori algorithm, etc.

Output	Desired output is given.	Desired output is not given.
Training data	Use training data to infer model.	No training data is used.
Complex model	It is not possible to learn larger and more complex models than with supervised learning.	It is possible to learn larger and more complex models with unsupervised learning.
Model	We can test our model.	We can not test our model.
Called as	Supervised learning is also called classification.	Unsupervised learning is also called clustering.
Example	Example: Optical character recognition.	Example: Find a face in an image.

DATA COMPRESSION

Data Compression is the process of encoding, restructuring or modifying the input data(images, sounds etc.) and converting it into a smaller representation with reduced size to facilitate storage or transmission.

This representation can be recreated to the original by the process of Decompression.

Data Compression can be achieved by dimensionality reduction.

Other than Compression there are 2 methods: Feature Selection and Feature Extraction.

Feature Selection: Used to find subset. The existing features in dataset are ranked according to the importance and the ones with less importance are discarded. It is a process of selecting the most relevant features and discarding noise in data.

***Feature Extraction*:** Transforms data from high dimensional space into a space with lower dimension. In feature extraction the number of features in dataset are reduced by creating new features from that of the existing ones. The original features are discarded.

The reduced feature set summarizes most of the information from the original feature set.

AUTOENCODERS

An *autoencoder* is a unsupervised learning technique. It is a artificial neural network used to learn data encodings of unlabeled data or task of representational learning.

Autoencoders are a specific type of feed forward neural networks trained to copy input to the output.

A bottleneck is imposed on the network to represent compressed knowledge of the original input.

The input is compressed into a lower dimensional code and then the output is reconstructed from this representation.

The code is also called as latent space representation which is a compact summary or compression of the input.

The general structure of autoencoder shows mapping of input Y to an output R through an internal representation or code h.

WHY COPYING THE INPUT TO THE OUTPUT?

- If the only purpose of autoencoders was to copy the input to the output, they would be useless.
- Indeed, we hope that, by training the autoencoder to copy the input to the output, the latent representation h will take on useful properties.
- This can be achieved by creating constraints on the copying task. One way to obtain useful features from the autoencoder is to constrain h to have smaller dimensions than x, in this case the autoencoder is called undercomplete.
- By training an undercomplete representation, we force the autoencoder to learn the most salient features of the training data. For our reference let's assume we are dealing with images, the job of an encoder is to extract features from the image, thereby reducing the image in height and width but simultaneously growing it in depth i.e an encoder makes a latent representation for the image. Now the job of the decoder is to decode the latent representation and form an image that satisfies our given criterion.

ARCHITECTURE OF AUTOENCODER

An Autoencoder is a type of neural network that can learn to reconstruct images, text, and other data from compressed versions of themselves.

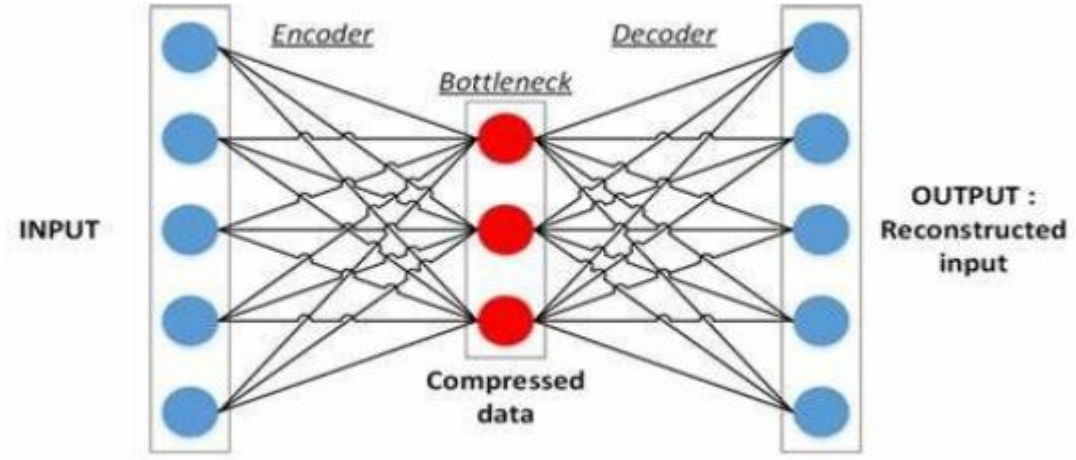
An Autoencoder consists of three layers:

1. Encoder

2. Code

3. Decoder

The Encoder layer compresses the input image into a latent space representation. It encodes the input image as a compressed representation in a reduced dimension.



Encoder

- Input layer take raw input data
- The hidden layers progressively reduce the dimensionality of the input, capturing important features and patterns. These layer compose the encoder.
- The bottleneck layer (latent space) is the final hidden layer, where the dimensionality is significantly reduced. This layer represents the compressed encoding of the input data.

Decoder

- The bottleneck layer takes the encoded representation and expands it back to the dimensionality of the original input.
- The hidden layers progressively increase the dimensionality and aim to reconstruct the original input.
- The output layer produces the reconstructed output, which ideally should be as close as possible to the input data.

The loss function used during training is typically a reconstruction loss, measuring the difference between the input and the reconstructed output. Common choices include mean squared error (MSE) for continuous data or binary cross-entropy for binary data.

During training, the autoencoder learns to minimize the reconstruction loss, forcing the network to capture the most important features of the input data in the bottleneck layer.

After the training process, only the encoder part of the autoencoder is retained to encode a similar type of data used in the training process. The different ways to constrain the network are: –

Keep small Hidden Layers: If the size of each hidden layer is kept as small as possible, then the network will be forced to pick up only the representative features of the data thus encoding the data.

Regularization: In this method, a loss term is added to the cost function which encourages the network to train in ways other than copying the input.

Denoising: Another way of constraining the network is to add noise to the input and teach the network how to remove the noise from the data.

Tuning the Activation Functions: This method involves changing the activation functions of various nodes so that a majority of the nodes are dormant thus, effectively reducing the size of the hidden layers.

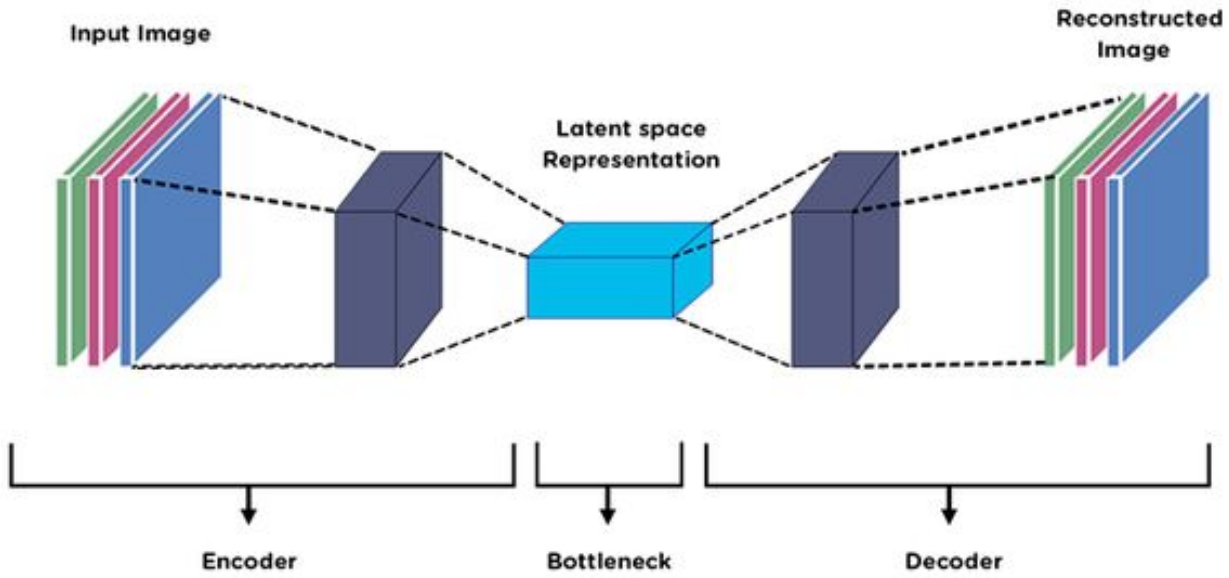
For your understanding:

The compressed image is a distorted version of the original image.

The Code layer represents the compressed input fed to the decoder layer.

The decoder layer decodes the encoded image back to the original dimension. The decoded image is reconstructed from latent space representation, and it is reconstructed from the latent space representation and is a lossy reconstruction of the original image.

Information is lost because it goes from a smaller to larger dimensionality. The loss function equation helps to find out the value of how much information is lost. This measure tells us how effectively the decoder has learned to reconstruct the input image. For this the output of the decoder should be of same size as the original image. Because if the size of image is different, there is no way to calculate the loss.



FEATURES OF AUTOENCODERS

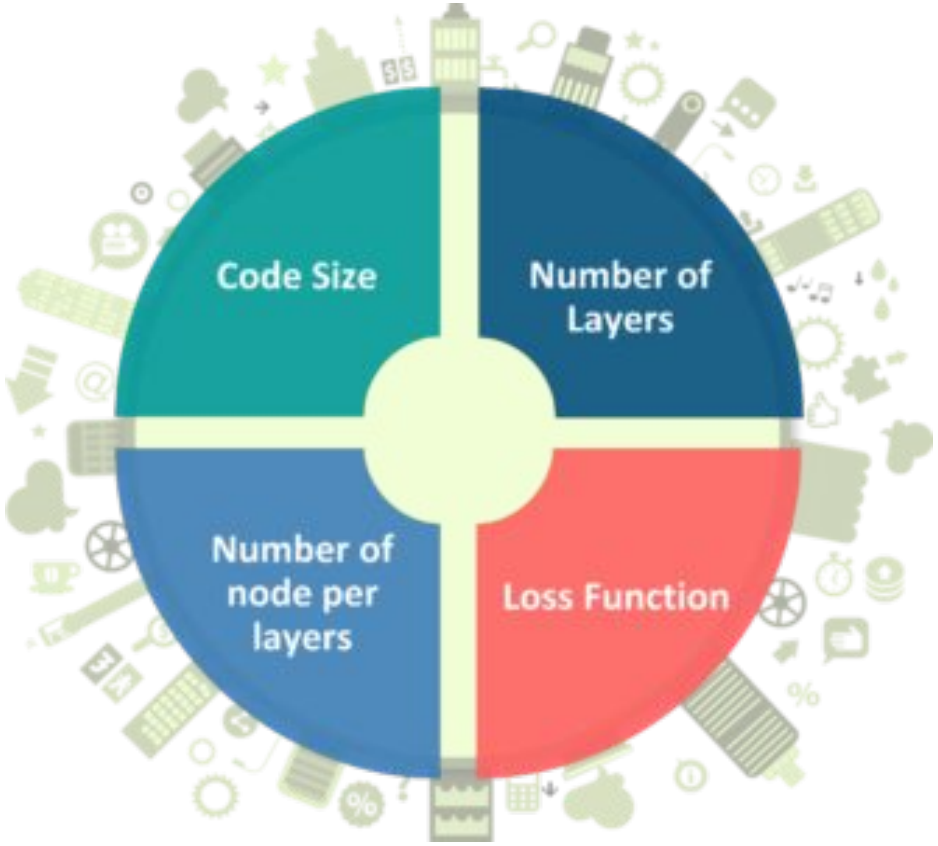
Data Dependent: Autoencoders are compression techniques where the model can be used only on data in which they have trained.

For example: Model of auto encoder which is used to compress images of houses cannot be used to compress human faces.

Lossy compression: Reconstruction of original data from compressed representation would result in a degraded output.

HYPERPARAMETERS IN AUTOENCODERS

Autoencoders are special case of feedforward networks, and may be trained with all of the same techniques, typically minibatch gradient descent following gradients computed by back-propagation.(for accurate reconstruction of input)



HYPER PARAMETERS

There are 4 hyperparameters need to set before training the network.

1. *Code size*. This represents the number of nodes in the middle part of the network. Smaller size results in more compression.

2. *Number of layers*. The autoencoder can be as deep as it wants to be.

3. *Loss function*. Either MSE or Binary Cross Entropy is used here.

4. *Number of nodes per layer*. Here the layer decreases with each subsequent layer of the encoder and increases back in the decoder. The decoder can be similar to the encoder in terms of layer structure.

LINEAR AUTOENCODER

A linear autoencoder is a type of artificial neural network used for unsupervised learning. It consists of two parts: an encoder and a decoder. The encoder takes an input and compresses it into a lower-dimensional representation, called the latent space. The decoder then takes the latent space representation and reconstructs the original input as accurately as possible.

Input: The autoencoder takes an input, which can be any type of data, such as an image, a text document, or a sensor reading.

Encoder: The encoder compresses the input into a lower-dimensional representation. This is done by passing the input through a series of linear layers. Each layer learns a set of weights that project the input onto a lower-dimensional space.

Latent space: The lower-dimensional representation is called the latent space. It contains the most important information about the input data.

Decoder: The decoder takes the latent space representation and reconstructs the original input. This is done by passing the latent space representation through a series of linear layers that are the mirror image of the encoder layers.

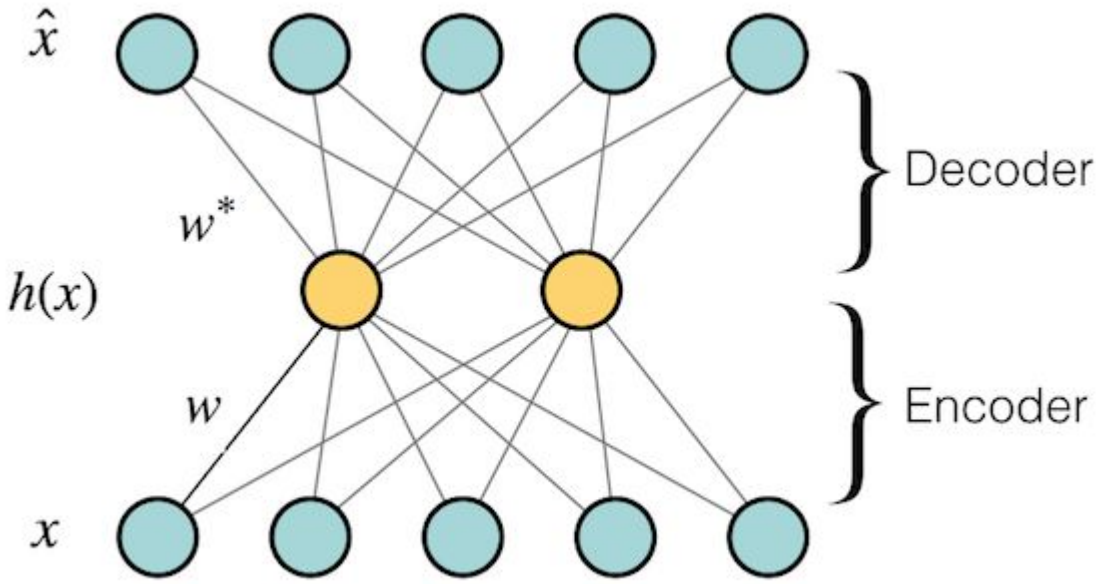
Output: The output of the autoencoder is a reconstruction of the original input.

LINEAR AUTOENCODERS ARE USEFUL FOR A VARIETY OF TASKS, INCLUDING:

Dimensionality reduction: Autoencoders can be used to reduce the dimensionality of data, which can be useful for tasks such as image compression and data visualization.

Feature learning: Autoencoders can be used to learn features from data, which can be used for tasks such as classification and regression.

Anomaly detection: Autoencoders can be used to detect anomalies in data, which can be useful for tasks such as fraud detection and network intrusion detection.



WHAT EXACTLY DOES THE ENCODE DO?

While reconstructing an image, we do not want the neural network to simply copy the input to the output. This type of memorization will lead to overfitting and less generalization power.

An autoencoder should be able to reconstruct the input data efficiently but by learning the useful properties rather than memorizing it.

There are many ways to capture important properties when training an autoencoder. Let's start by getting to know about under complete autoencoders.

TYPES OF ENCODERS

- Undercomplete Autoencoder
- Overcomplete Autoencoder
- Regularized Autoencoder.
- Denoising Autoencoder
- Sparse Autoencoder
- Contractive Autoencoder
- Convolutional Autoencoder
- Variational Autoencoder

UNDERCOMPLETE AUTOENCODERS:

- An autoencoder in which the dimension of the code is less than the dimension of the input is called as undercomplete encoder.
- Undercomplete autoencoder limits the amount of information that can flow through the network by constraining the number of hidden nodes.
- Ideally this encoding learns and describes the latent attributes of the input data.
- Undercomplete autoencoders is a sandwich architecture keeping the code size small. Training the autoencoder to perform the input copying task will result in code h taking on useful properties.

Learning an under complete representation forces the autoencoder to capture the salient features of training data and it won't be able to copy the inputs to the outputs.

Learning is described by loss function.

The objective of undercomplete autoencoder is to capture the most important features present in the data.

Undercomplete autoencoders have a smaller dimension for hidden layer compared to the input layer. This helps to obtain important features from the data. It minimizes the loss function by penalizing the g(f(x)) for being different from the input x.

Advantages-

Undercomplete autoencoders do not need any regularization as they maximize the probability of data rather than copying the input to the output.

Drawbacks-

Using an overparameterized model due to lack of sufficient training data can create overfitting.

OVERCOMPLETE AUTOENCODERS:

- Undercomplete autoencoders with code dimensions less than the input dimensions can learn the most salient features of data distribution . These autoencoders fail to learn anything useful if encoder and decoder are given too much capacity.
- A similar problem occurs if hidden code has dimension greater than the input. They are overcomplete autoencoders. In this case even a linear encoder and linear decoder can learn to copy the input to the output without learning anything useful about data distribution.
- An overcomplete autoencoder is when the encoded representation has more dimensions than the original input. This can capture a lot of details, but it might also memorize the data.

REGULARIZED AUTOENCODER:

- A regularized autoencoder is a variant of the autoencoder architecture that includes additional constraints or penalties during training. These constraints encourage the autoencoder to learn more meaningful and generalizable features, preventing it from overfitting or simply copying the input data.
- Ex: The regularized autoencoder helps the computer understand shapes better by focusing on the important parts and not getting distracted by tiny details.

SPARSE AUTOENCODER:

- A sparse autoencoder is a type of autoencoder that encourages the learned representations (the encoded values) to have fewer active neurons, meaning that only a few neurons should "fire" or be active for any given input.
- This sparsity constraint aims to extract more meaningful and distinctive features from the input data by forcing the autoencoder to focus on the most relevant information.
- Sparse autoencoders have hidden nodes greater than input nodes.
- Sparsity constraint is introduced on the hidden layer. This is to prevent output layer copy input data.

Sparse Autoencoder

This type of autoencoder typically contains more hidden units than the input but only a few are allowed to be active at once. This property is called the sparsity of the network. The sparsity of the network can be controlled by either manually zeroing the required hidden units, tuning the activation functions or by adding a loss term to the cost function.

Advantages

- The sparsity constraint in sparse autoencoders helps in filtering out noise and irrelevant features during the encoding process.
- These autoencoders often learn important and meaningful features due to their emphasis on sparse activations.

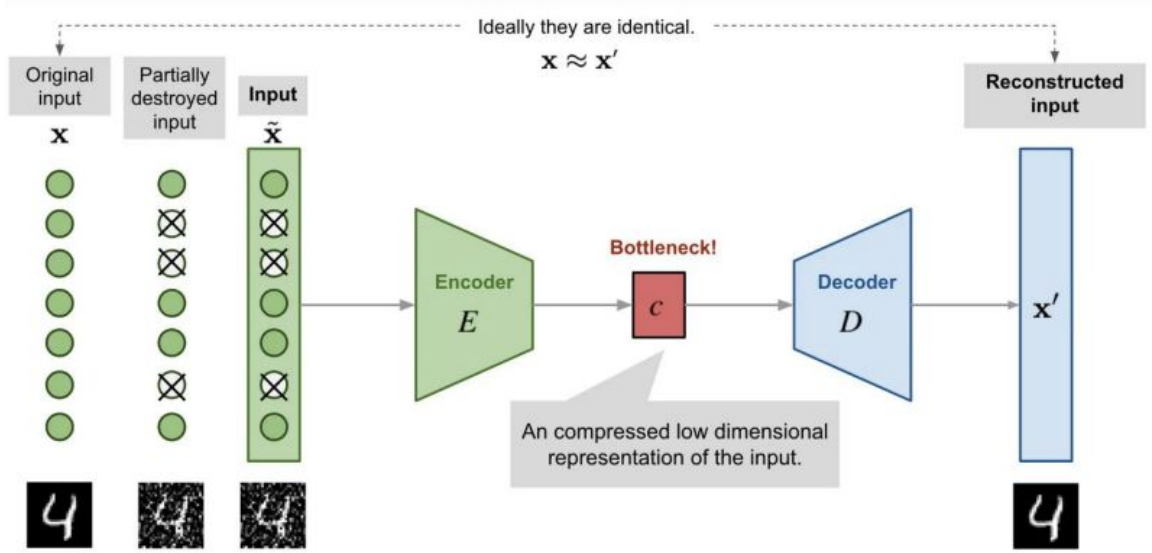
Disadvantages

- The choice of hyperparameters play a significant role in the performance of this autoencoder. Different inputs should result in the activation of different nodes of the network.
- The application of sparsity constraint increases computational complexity.

DENOISING AUTOENCODER:

A denoising autoencoder is a type of autoencoder designed to handle noisy data. It learns to remove noise from input data while trying to reconstruct the original, clean data. This helps the autoencoder to learn more robust and useful features by focusing on the important patterns in the data.

It cleans up messy pictures or sounds. the denoising autoencoder cleans up things that are hard to see or hear, so you can understand them better.



Denoising Autoencoder

Denoising autoencoder works on a partially corrupted input and trains to recover the original undistorted image. As mentioned above, this method is an effective way to constrain the network from simply copying the input and thus learn the underlying structure and important features of the data.

Advantages

This type of autoencoder can extract important features and reduce the noise or the useless features.

Denoising autoencoders can be used as a form of data augmentation, the restored images can be used as augmented data thus generating additional training samples.

Disadvantages

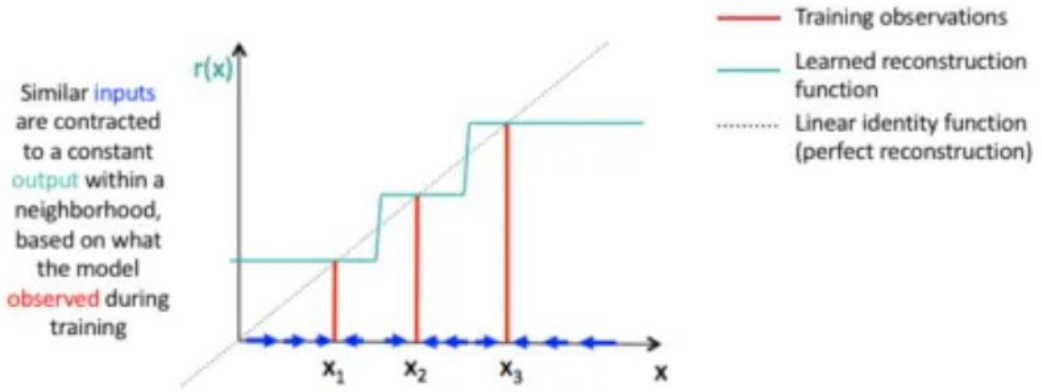
Selecting the right type and level of noise to introduce can be challenging and may require domain knowledge.

Denoising process can result into loss of some information that is needed from the original input. This loss can impact accuracy of the output.

CONTRACTIVE AUTOENCODER:

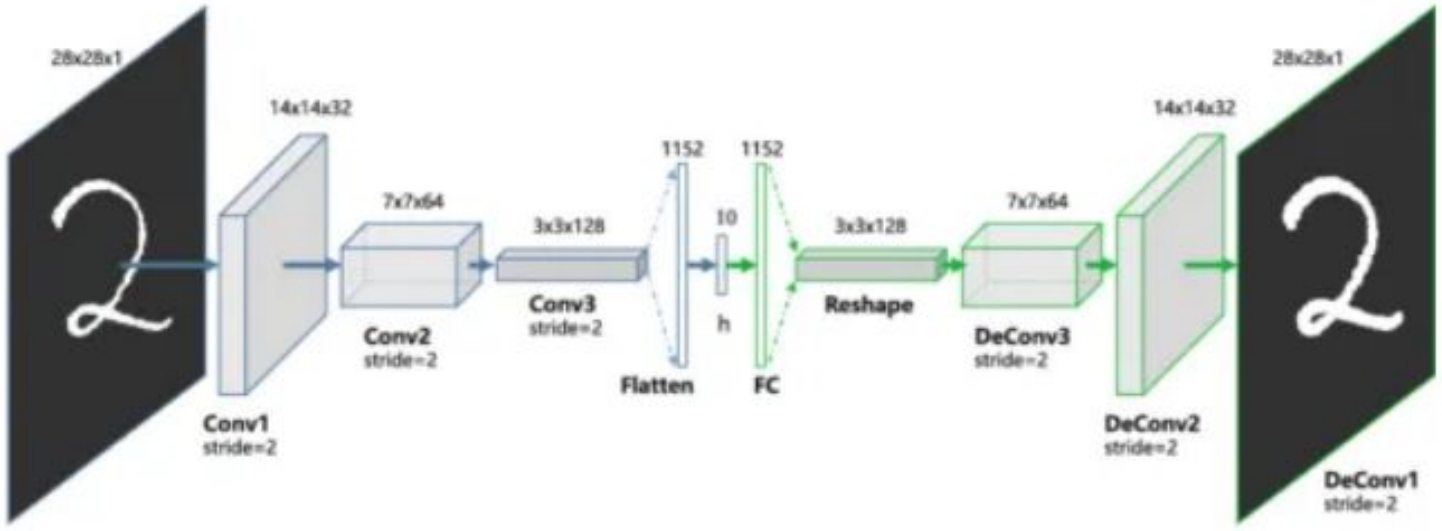
- A contractive autoencoder is like a curious explorer that learns to pay extra attention to the important parts of things.
- Contractive autoencoder simply targets to learn invariant representations to unimportant transformations for the given data.
- The main goal of contractive autoencoder is to have a robust learned representation that is less sensitive to small variation of data.
- A penalty term is applied to loss function so as to make representation robust.
- Ex: Imagine you're training a computer to recognize people's faces. However, you also want the computer to be good at recognizing the same person even if they make small changes, like wearing glasses or a hat.
- In the same way, a contractive autoencoder is a clever computer trick that learns to spot the essential features in data. It does this so well that even if some details change a little bit, it can still recognize what the data is about. Just like our curious explorer, it focuses on the special parts that define things and helps the computer understand them better.

A **contractive autoencoder** is an unsupervised deep learning technique that helps a neural network encode unlabeled training data. This is accomplished by constructing a **loss term** that penalizes large derivatives of our hidden layer activations with respect to the input training examples, **essentially penalizing** instances where a small change in the input leads to a large change in the encoding space.



CONVOLUTIONAL AUTOENCODER

Autoencoders in their traditional formulation does not take into account the fact that a signal can be seen as a sum of other signals. Convolutional Autoencoders use the convolution operator to exploit this observation. They learn to encode the input in a set of simple signals and then try to reconstruct the input from them, modify the geometry or the reflectance of the image.



VARIATIONAL AUTOENCODER

Variational autoencoder makes strong assumptions about the distribution of latent variables and uses the **Stochastic Gradient Variational Bayes** estimator in the training process.

Advantages

Variational Autoencoders are used to generate new data points that resemble the original training data. These samples are learned from the latent space.

Variational Autoencoder is probabilistic framework that is used to learn a compressed representation of the data that captures its underlying structure and variations, so it is useful in detecting anomalies and data exploration.

Disadvantages

Variational Autoencoder use approximations to estimate the true distribution of the latent variables. This approximation introduces some level of error, which can affect the quality of generated samples.

The generated samples may only cover a limited subset of the true data distribution. This can result in a lack of diversity in generated samples.

APPLICATIONS OF ENCODER

Dimensionality Reduction: Autoencoders can be used to reduce the dimensionality of high-dimensional data while preserving important features. This can be particularly useful in data visualization and compression. For instance, consider a dataset of images. By training an autoencoder on these images, you can create a compressed representation that captures the essential features of the images. This can be handy for reducing storage space or for speeding up further processing.

Image Denoising: Autoencoders can be trained to remove noise from images. During training, noisy versions of clean images are used as input, and the autoencoder learns to produce denoised versions of the same images as output. This is achieved by forcing the autoencoder to learn the underlying structure of the data and remove the noise. This technique is especially useful in fields like medical imaging, where clean images are crucial for accurate diagnosis.

Anomaly Detection: Autoencoders can be used for anomaly detection by training on normal instances and then detecting deviations from this learned normalcy. When presented with an unseen instance, the autoencoder will likely fail to reconstruct it accurately if it is an anomaly. This can be applied in various domains, such as fraud detection in finance or detecting defects in manufacturing processes.

Feature Learning: Autoencoders can be used to automatically learn useful features from raw data. Instead of manually engineering features, an autoencoder can be trained to extract meaningful representations from the input data. This is particularly valuable when working with complex data types like text or speech.

Style Transfer and Generation: Autoencoders can be used for style transfer, where the style of one input is transferred to the content of another. By training an autoencoder on a dataset containing pairs of images with different styles, you can create a model that can generate images in the style of the input images. This is widely used in creative applications such as generating artwork or altering the style of photographs.

Collaborative Filtering and Recommender Systems: Autoencoders can be employed to build personalized recommender systems. In collaborative filtering, an autoencoder can learn representations of users and items (e.g., movies, products) based on their interactions. By training on user-item interaction data, the autoencoder can generate recommendations for users by identifying similar users and items in the latent space.

Data Imputation: Autoencoders can be used to fill in missing values in data. For instance, in medical records, certain patient data might be incomplete. An autoencoder can be trained on the available data to predict the missing values, helping to create a more complete dataset for analysis.

Image Filters: Using filters on Instagram etc

Face Morphing: Matching or observing what your face will look like with extra features

Language Translation: Imagine you want to read a story in a language you don't understand. An autoencoder can learn how sentences are built in different languages and help translate the story for you

Predictive Typing: Suggestions during chatting on WhatsApp

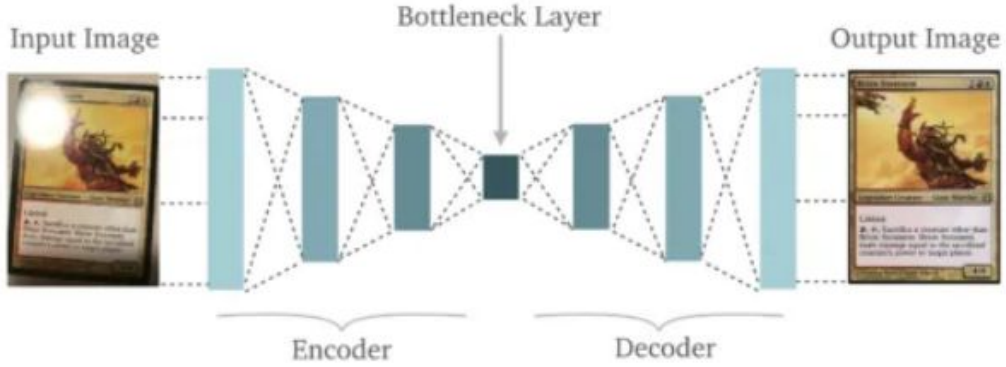
Robotics: Autoencoders can help robots understand their surroundings. They learn how objects and places look, so the robots can recognize things and move around without bumping into stuff.

Image Coloring

Autoencoders are used for converting any black and white picture into a colored image. Depending on what is in the picture, it is possible to tell what the color should be.

Feature variation

It extracts only the required features of an image and generates the output by removing any noise or unnecessary interruption.

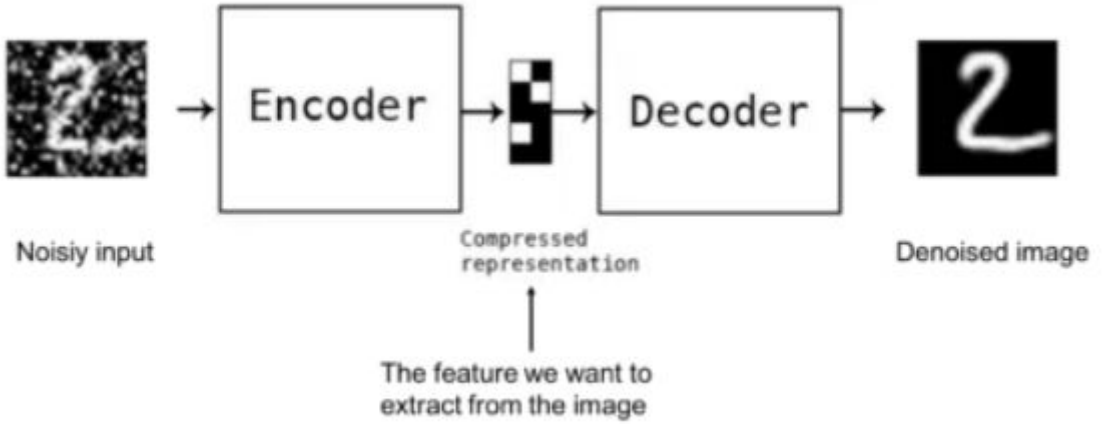


Dimensionality Reduction

The reconstructed image is the same as our input but with reduced dimensions. It helps in providing the similar image with a reduced pixel value.

Denoising Image

The input seen by the autoencoder is not the raw input but a stochastically corrupted version. A denoising autoencoder is thus trained to reconstruct the original input from the noisy version.



Watermark Removal

It is also used for removing watermarks from images or to remove any object while filming a video or a movie.

Now that you have an idea of the different industrial applications of Autoencoders, let’s continue our article and understand the complex architecture of Autoencoders.

IMAGE COMPRESSION WITH AUTOENCODERS

There are two types of image compression: lossless and lossy. Lossless compression methods preserve all of the data in the original image, while lossy compression methods discard some of the data to achieve higher compression rates.

Autoencoders can be used for both lossless and lossy compression. Lossless compression can be achieved by using a bottleneck layer that is the same size as the input data. In this case, the autoencoder essentially learns to encode and decode the input data without any loss of information.

Lossy compression can be achieved by using a bottleneck layer that is smaller than the input data. In this case, the autoencoder learns to discard some of the data to achieve higher compression rates. The amount of data that is discarded depends on the size of the bottleneck layer.

Here are some examples of image compression using autoencoders:

- A 512×512 color image can be compressed to a 64×64 grayscale image using an autoencoder with a bottleneck layer of size 64.
- A 256×256 grayscale image can be compressed to a 128×128 grayscale image using an autoencoder with a bottleneck layer of size 128.
- The effectiveness of autoencoder-based compression techniques can be evaluated by comparing the compressed and reconstructed images to the original images. The most common evaluation metric is the peak signal-to-noise ratio (PSNR), which measures the amount of noise introduced by the compression algorithm. Higher PSNR values indicate better compression quality.

IMAGE RECONSTRUCTION WITH AUTOENCODERS

Autoencoders are a type of neural network that can be used for image compression and reconstruction. The process involves compressing an image into a smaller representation and then reconstructing it back to its original form. Image reconstruction is the process of creating an image from compressed data.

The compressed data can be thought of as a compressed version of the original image. To reconstruct the image, the compressed data is fed through a decoder network, which expands the data back to its original size. The reconstructed image will not be identical to the original, but it will be a close approximation.

How autoencoders can be used for image reconstruction:

Autoencoders use a loss function to determine how well the reconstructed image matches the original. The loss function calculates the difference between the reconstructed image and the original image. The goal of the autoencoder is to minimize the loss function so that the reconstructed image is as close to the original as possible.