Assignment No : 4 DL

Q1. Short note on autoencoder?

→ An autoencoder is a type of neural network used primarily for unsupervised learning. Its goal is to learn an efficient, compressed representation of input data (encoding), and then reconstruct the original data from this compressed version (decoding)

An autoencoder consists of two main parts :

1. Encoder : This part compresses the input into a lower-dimensional representational (latent space), effectively learning the most important features of the data.

2. Decoder : The decoder takes the compressed data and tries to reconstruct the original input as closely as possible.

Autoencoder are widely used for:

• Dimensionality reduction : Similar to PCA, but more powerful as they can learn non-linear relationship.

• Denoising : Autoencoders can remove noise from data by training the network to ignore the irrelevant features.

• Image Compression and generation : They are often used in ~~lat~~ tasks like image generation or compression, where the N/W learns to represent images with fewer bits.

Q2. With diagram explain architecture of autoencoder?

→ An autoencoder's architecture typically consists of three main parts :

1. Encoder : This compresses the input data into a lower-dimensional representation (latent space).

2. Bottleneck (Latent Space) : This is the compressed, lower-dimensional representation of the input data.

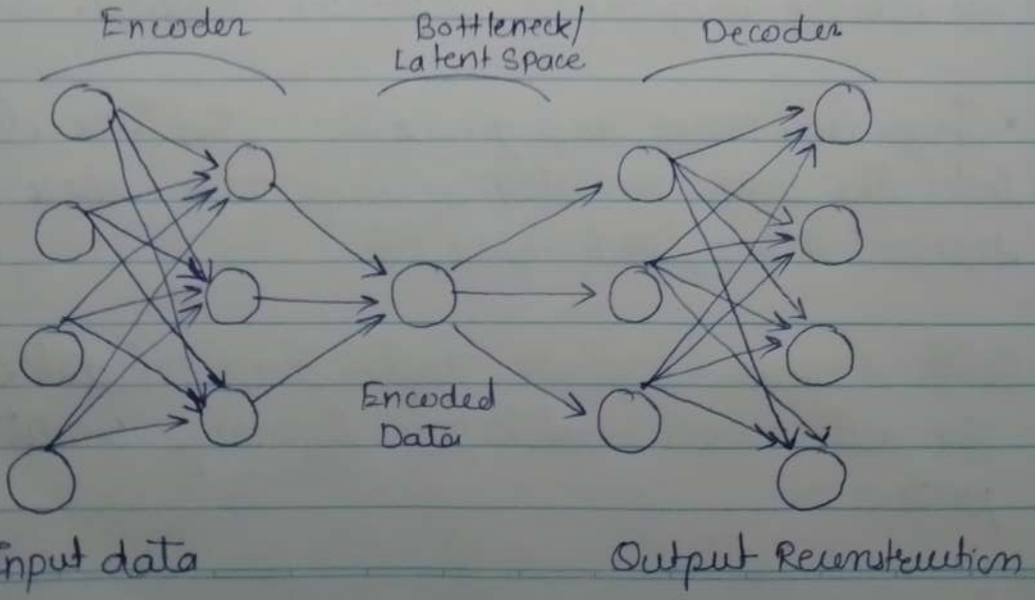
3. Decoder : This reconstructs the input data from the compressed representation.

Features:

- Input layer : Takes the original data as input.
- Hidden layer (Encoder) : These layer reduce the dimensionality of the input data.
- Bottleneck (Layer) :- The middle layer containing the compressed representation of the input data.

- Hidden layer (Decoder) : These layers reconstruct the original data from the compressed version.

- Output layers : Produces the reconstructed data.



Page No.
Date

- Input layer: This is the layer where the original data is fed into the network.

- Encoder: A series of layers that reduce the dimensionality of the input data step by step. each hidden layer in this section applies transformations (usually nonlinear) to extract the essential features of the data.

- Bottleneck (Latent Space): The central layer in the network that holds the compressed representation. This layer has the smallest number of neurons, as it is the most compact version of the input data. The bottleneck forces the autoencoder to learn efficient data encoding.

- Decoder: The layers that mirror the encoder, but in reverse, gradually expanding the compressed data back to its original dimensions. It reconstruct the input as accurately as possible.

- Output Layers: The final layer that attempts to recreate the original input from the compressed data.

Q3 What are the parameter that you need to define while training an autoencoder?

→ ① Learning Rate:
- Controls how much to adjust the weights with each updates.
- A small learning rate ensures slow and steady convergence, while a larger one speeds up training but might overshoot

optimal values.

② Number of Layers:
- The depth of the encoder and decoder, which determines the complexity of the model.
- Deeper architecture may learn more complex features but might also required more data and computational power.

③ Bottleneck Size (Latent Space Dimension):
- This determines how much the input is compressed.
- A small bottleneck forces the model to learn a more efficient representation, but if its too small important information may be lost.

④ Activation Functions:
- Activation functions like ReLU or sigmoid are applied after each layer to introduce non-linearity.
- Common choices includes:
  - RELU for hidden layers.
  - Sigmoid or tanh for bottleneck or output layers (especially for bounded data).

⑤ Loss Function:
- Measures the difference between the input and the reconstructed output.
- Mean Squared error is commonly used for continuous data, while Binary Cross Entropy is often applied for binary data or data in the range [0,1].

Q.4 Applications of auto encoder ?
→

① Dimensionality Reduction:
- Autoencoders are often used to reduce the number of features in a dataset while retaining the most important information. This is similar to principal component analysis, but autoencoders can capture complex, nonlinear relationships.

② Anomaly Detection:
- Autoencoders can detect anomalies by reconstructing normal data patterns. If the model fails to accurately reconstruct new data, it could indicate an anomaly.

- Example :- Fraud detection in financial transactions or identifying defective products in manufacturing.

③ Image Compression:
- Autoencoders can compress high-dimensional image data into a smaller representation, which can then be used to reconstruct the image with minimal loss.

Example : Image Compression for more efficient storage & transmission.

④ Feature Extraction :
- Autoencoders are used to automatically extract meaningful features from data. The compressed latent space can serve as a set of features for other machine learning algorithms.

Example :- Extracting features from images to use in tasks like classification or clustering.

Date

⑤ Medical Imaging:
- Autoencoders are used for tasks like reconstructing high-quality images from lower-quality medical scans, detecting abnormalities or segmenting regions of interest

- Ex: Enhancing MRI, CT, or X-ray images for better diagnosis

⑥ Face Recognition:
- Autoencoders can compress face images into smaller representation which can then be used for face recognition by comparing by comparing the latent representations.

- Ex: Identifying individuals in security or authentication systems.

Q5 Short note on ~~a)~~
a) Undercomplete autoencoder.
An undercomplete autoencoder is an autoencoder where the dimensionality of the latent space (bottleneck) is smaller than the input data. The objective is to force the model to learn a compressed, meaningful representation of the data by limiting the bottleneck size. The smaller latent space prevents the model from simply compying the input data, which ensures that the autoencoder captures the most important patterns or features.

key ideas: Compression forces the model to learn important features, reducing the input to fewer dimensions.

Risk: If the bottleneck is too small, the model may lose critical information, leading to poor reconstruction.

Page No.
Date

b) Regularized Autoencoder:
A regularized autoencoder adds a constraint or regularization term to the learning process to prevent overfitting & encourage the autoencoder to learn more robust representation. Unlike undercomplete autoencoders, the latent space can be as large as (or ~~but~~ larger than) the input data, but the regularization a specific behavior in the latent representation.

Common Regularization Techniques:

- Sparse Autoencoder: Encourages sparsity in the latent space by forcing many activations to be zero, resulting in a more efficient respresentation.

- Denoising Autoencoder: Trains the autoencoder to reconstruct the original input from a noisy version, improving its robustness to noise.

- Variational Autoencoder (VAE): Adds a probabilistic constraint on the latent space, forcing it to follow a certain distribution, which enables generation of new data.

- key idea: Regularization prevents overfitting & helps the autoencoders learn more generalizable features.

Q.10 Short note on contractive autoencoder?
→ - The main goal of Contractive autoencoder (CAE) is to have a robust learned representation that is less sensitive to small variation in the data.

- A penalty term is applied to the loss function so as to make the representation robust.

- In order to make the derivatives of f to be as small as possible, contractive autoencoder introduces an explicit regularizer on the code $h = f(x)$

- The penalty term is Frobenius norm of the Jacobian matrix which is calculated with respect to input for the hidden layer. Frobenius norm of the Jacobian matrix is the sum of square of all elements.

$$L = |x - g(f(x))| + \lambda \| J_f(x) \|_F^2$$

$$\| J_f(x) \|_F^2 = \sum_{i,j} \left(\frac{\partial h_j(x)}{\partial x_i} \right)^2$$

- Contractive autoencoder is similar to denoising autoencoder in a sense that in presence of small Gaussian noise the denoising reconstruction error is equivalent to a contractive penalty on the reconstruction function that maps x to $r = g(f(x))$.

- CAE ~~stup~~ surpasses results obtained by regularizing autoencoder using weight decay or by denoising. CAE is a better as compare to denoising autoencoder to learn useful feature extraction.

Q.6 Describe convolution auto encoder & sparse autoencoder.

Convolution autoencoder :-

- A convolution Autoencoder (CAE) is a type of autoencoder specifically designed to work with image data.
- It uses convolutional layers instead of fully connected layers to learn spatial hierarchies in the data.
- CAEs are especially effective for image-related tasks, as convolutional layers are adept at capturing spatial features like edges, textures and patterns.

Architecture:

• Encoder: The encoder part of a CAE consists of convolutional layers followed by pooling (or subsampling) layers to progressively reduce the spatial dimensions of the input, while preserving important image features.

• Bottleneck: The smallest representation of the input is obtained in the bottleneck layer, where the image data is compressed into a compact form.

• Decoder: The decoder consists of deconvolution (transposed convolution) layers, which progressively upsample the features maps to reconstruct the original image from the compressed data.

Applications:

• Image Denoising: CAEs are used to remove noise from images by learning to map noisy images to clean images.

• Image Compression: CAEs can compress images into smaller representations for efficient storage.

• Image Generation: Variations of CAEs are used in tasks like image generation or enhancement.

- A convolution Autoencoder (CAE) is a type of autoencoder specifically designed to work with image data.
- It uses convolutional layers instead of fully connected layers to learn spatial hierarchies in the data.
- CAEs are especially effective for image-related tasks, as convolutional layers are adept at capturing spatial features like edges, textures and patterns.

Architecture:

• Encoder: The encoder part of a CAE consists of convolutional layers followed by pooling (or subsampling) layers to progressively reduce the spatial dimensions of the input, while preserving important image features.

• Bottleneck: The smallest representation of the input is obtained in the bottleneck layer, where the image data is compressed into a compact form.

• Decoder: The decoder consists of deconvolution (transposed convolution) layers, which progressively upsample the features maps to reconstruct the original image from the compressed data.

Applications:

• Image Denoising: CAEs are used to remove noise from images by learning to map noisy images to clean images.

• Image Compression: CAEs can compress images into smaller representations for efficient storage.

• Image Generation: Variations of CAEs are used in tasks like image generation or enhancement.

Advantages :
• Spatial Features : CAEs efficiently capture spatial patterns & relationships in images.

• Computational Efficiency : Convolutional layers reduce the number of parameters compared to fully connected layers, making them more scalable for large images.

② Sparse Autoencoder :-

- A sparse Autoencoder introduces sparsity into the latent representation by forcing many neurons in the hidden layers to remain inactive (i.e, have output close to zero). This is achieved by adding a sparsity constraint or penalty during training. The goal is to learn a more efficient and compact representation of the data, where only a few neurons are "active" for any given input.

How it works:
• Sparsity Constraint : A regularization term is added to the loss function that ~~po~~ penalizes the network when too many neurons are activated. This ensures that only a small subset of neurons "fire" for a given input.

• Activation Function : Nonlinear activation functions like ReLU or sigmoid are typically used in the hidden layers to enforce sparsity.

• L1 Regularization : Often, an L1 penalty is added to the loss function to induces sparsity by minimizing the activation of unnecessary neurons

Page No.
Date

Applications:

• Features ~~Exto~~ Extraction : Sparse autoencoders are often used to extract meaningful features from data, especially in high-dimensional spaces like image or text data.

• Anomaly Detection : Sparse representation can make it easier to detect unusual patterns or outliers in the data.

• Pretraining : Sparse autoencoders can be used as a pretraining step for other machine learning models by learning useful features in an unsupervised manner.

Advantages :

• Efficient Representation : By enforcing sparsity, the autoencoder is forced to learn only the most important features of the data.

• Prevents Overfitting : Sparsity can act as a form of regularization, preventing the model from overfitting to the training data.

Q.7 Describe stacked autoencoder & deep autoencoder ?

→

① Stacked Autoencoder :

- A stacked Autoencoder is an autoencoder that ~~con~~ consists of multiple layers of encoders and decoders.

- Each encoder-decoder pair is stacked one on top of another ~~&~~ to form a deeper, hierarchical network.

- The output of one layer becomes the input to the next layer, allowing the autoencoder to capture more complex features and patterns from the data.

How it Works:

• Layer-Wise Training: It stacked autoencoders, training can be done one layer at a time (unsupervised pretraining). Each layer is trained as a simple autoencoder, and the learned features from layers are used as input for training the next layer.

• Fine-Turning: After the pretraining is completed, the entire network is fine-tuned with backpropagation using a supervised task (if applicable) or supervised ~~rec~~ reconstruction.

Architecture:

• Multiple Encoders: Each encoder reduces the dimensionality of the input data step by step, extracting higher-level features at each layers

• Multiple decoder: The decoder mirror the encoders, progressively ~~rec~~ reconstructing the original data from the learned features.

• Latent Space: At the bottleneck, the final compressed representation is used for reconstruction by the decoders.

② Deep Autoencoder:

- A deep Autoencoder is a Type of autoencoder with a deep architecture, meaning it has many layers (both in the encoder and decoder) to learn highly abstract representation of the data.
- Unlike traditional shallow autoencoders, deep autoencoders can extract more meaningful hierarchical features by learning complex relationships between the input and output data.

How it works :-

• Deep Network : Deep autoencoders have several hidden layers, each with non-linear activation functions. These allow the network to learn both low-level and high-level abstract representations of the data.

• Encoder : The encoder consists of multiple hidden layers that progressively reduce the dimensionality of the input until ~~eet~~ reaching a compressed, abstract representation (latent space).

• Bottleneck (latent space) : At the bottleneck, the dimensionality is significantly reduced, capturing the core features in a compact form.

• Decoder : The decoder mirrors the encoder, with multiple hidden layers progressively reconstructing the original input from the compressed representation.

Training :-

• End-to-End Training :- Deep autoencoders are often trained using backpropagation to minimize the reconstruction error between the input and output. This is typically done with a loss function like MSE or BCE, depending on the data.

• Pretraining and Fine-Tuning : For deep autoencoders with many layers, unsupervised layer-wise pretrained (similar to stacked autoencoders) can be applied. Afterward, the network is fine-tuned on the full architecture.

Page No.
Date

Q·8 Describe denoising autoencoder & variational autoencoder?
→

① Denoising Autoencoder (DAE):
- A Denoising Autoencoder is a type of autoencoder designed to reconstruct ~~a~~ clean data from a corrupted or noisy version. It is trained to map noisy inputs to their corresponding clean outputs, thereby learning robust features that are resilient to noise.

How It works :-
• Noise Addition : During Training, noise (e.g., Gaussian noise or making noise) is added to the input data.

• Objective : The autoencoder is trained to reconstruct the original, clean version of the data from the noisy ~~a~~ input. This forces the model to focus on capturining the essential structure of the data, rather than memorizing specific details.

Applications :

• Image Denoising : Removing noise from images by learning the the underlying clean image.

• Robust Features learning : Learning representation that are less sensitive to noise and outliners.

• Speech/Signal Processing : Denoising audio signals for clearer output.

Page No.
Date

2. Variational Autoencoder (VAE):

A Variational Autoencoder (VAE) is a generative model that extends the autoencoder architecture by introducing a probabilistic approach. Instead of learning a fixed encoding, VAEs learn a distribution over the latent space, allowing them to generate new data points by sampling from this distribution.

How It Works:

- Latent Space Distribution: The encoder maps the input to a probability distribution (usually Gaussian) in the latent space, characterized by a mean and variance. The decoder samples from this distribution to reconstruct the input.

- Loss Function: Combines a reconstruction loss (to ensure accurate reconstruction) and a KL divergence term (to regularize the latent space and ensure it follows a desired distribution, like a normal distribution).

Applications:

- Data Generation: Generating new data points that are similar to the training data (e.g, new images, text).

- Semi-supervised Learning: Leveraging the generative power of VAEs for learning with limited labelled data.

Q9 Short note stochastic autoencoder?

→ - A Stochastic Autoencoder introduces randomness or stochastic into the encoding process to learn more robust and generalized representation of the data.

- Unlike traditional autoencoders, where the encoders deterministically maps the input to a latent representation, stochastic autoencoders involve a probabilistic component, such as random noise or a probability distribution, in the latent space.

Features :

- Randomness : The encoder outputs a distribution (like in Variational Autoencoders) or adds noise to the encoding ensuring that the model does not memorize the input.

- Improved Generalization : By adding stochasticity, the model learns more flexible representations and is less prone to overfitting.

- Reconstruction : The decoder reconstructs the input from the stochastic latent representation, encouraging the model to focus on core features.