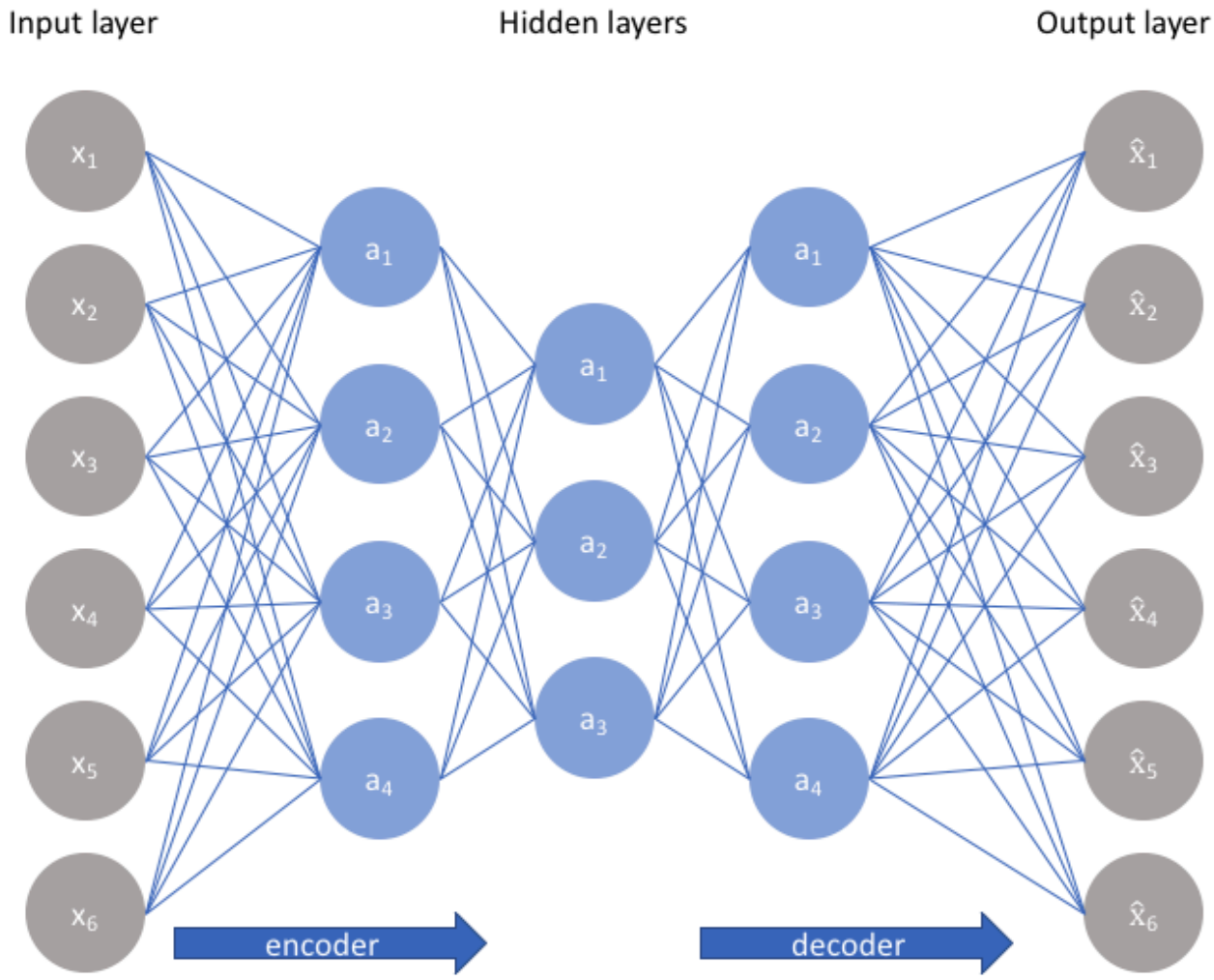
Auoencoder Notes

2024-08-02

Let’s dive into autoencoders with an easy-to-understand explanation, including the basic math behind them and the different types of autoencoders.

Basic Concept and Mathematics of Autoencoders

Architecture



1. **Encoder**: Maps input data to a latent representation.
2. **Latent Space (Bottleneck)**: Holds the compressed, encoded representation.
3. **Decoder**: Reconstructs the original data from the latent representation.

Mathematical Notation

- **Input Data**: Let $\mathbf{x}$ represent the input data, which could be an image, a vector, etc.
- **Encoder Function**: The encoder maps the input to the latent space using a function f, typically parameterized by weights and biases in a neural network.

  $$\mathbf{z} = f(\mathbf{x})$$

  Here, $\mathbf{z}$ represents the encoded representation (latent code).
- **Latent Representation**: The latent space $\mathbf{z}$ is usually of lower dimension than $\mathbf{x}$. The dimensionality reduction forces the network to learn compact features.

- **Decoder Function**: The decoder maps the latent representation back to the input space using a function g.

$$\hat{\mathbf{x}} = g(\mathbf{z})$$

  $\hat{\mathbf{x}}$ is the reconstructed version of the input data.

Loss Function

The autoencoder is trained to minimize the difference between the input $\mathbf{x}$ and the output $\hat{\mathbf{x}}$. This difference is measured using a loss function, often Mean Squared Error (MSE):

$$\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}}) = ||\mathbf{x} - \hat{\mathbf{x}}||^2$$

The parameters of the encoder and decoder (weights and biases) are optimized to minimize this loss, meaning the network learns to reconstruct the input as accurately as possible.

Types of Autoencoders

1. Undercomplete Autoencoder

- **Description**: The size of the latent space $\mathbf{z}$ is smaller than the input $\mathbf{x}$.
- **Purpose**: To learn the most important features of the data by compressing the information.
- **Usage**: Dimensionality reduction, feature learning.

2. Overcomplete Autoencoder

- **Description**: The latent space $\mathbf{z}$ is larger than the input.
- **Challenge**: This setup risks learning an identity function, where the output $\hat{\mathbf{x}}$ simply copies the input $\mathbf{x}$ without meaningful compression.
- **Solution**: Regularization techniques are used to prevent overfitting and ensure that the model learns useful features.

3. Sparse Autoencoder

- **Description**: An undercomplete autoencoder with an added sparsity constraint on the latent representation $\mathbf{z}$.
- **How It Works**: It enforces that only a few neurons in the latent layer are active (non-zero) at a time.
- **Mathematics**: This is often achieved by adding a sparsity penalty to the loss function, such as the L1 regularization:

$$\mathcal{L}_{sparse} = \sum_i |\mathbf{z}_i|$$

- **Usage**: Feature extraction where only a few features are necessary, useful in natural language processing and image recognition.

4. Denoising Autoencoder

- **Description**: Designed to remove noise from the data.
- **How It Works**: The input data $\mathbf{x}$ is first corrupted with noise to form $\tilde{\mathbf{x}}$. The autoencoder is then trained to reconstruct the original data $\mathbf{x}$ from the noisy input $\tilde{\mathbf{x}}$.
- **Loss Function**: Similar to standard autoencoders but with a noisy input:

$$\mathcal{L}(\mathbf{x}, \hat{\mathbf{x}}) = ||\mathbf{x} - g(f(\tilde{\mathbf{x}}))||^2$$

- **Usage**: Image and audio denoising, data cleaning.

5. Variational Autoencoder (VAE)

- **Description**: Introduces a probabilistic approach to the latent space, learning a distribution rather than a fixed representation.
- **Mathematics**: The encoder outputs parameters for a probability distribution (usually a Gaussian), from which the latent vector $\mathbf{z}$ is sampled.
  - **Encoder**: Outputs mean μ and standard deviation σ for each dimension in the latent space.
  - **Latent Space Sampling**: $\mathbf{z} \sim \mathcal{N}(\mu, \sigma^2)$
  - **Loss Function**: Combines reconstruction loss with a regularization term (KL-divergence) to ensure that the learned distribution is close to a standard normal distribution.

$$\mathcal{L} = \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})] - D_{KL}(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))$$

- **Usage**: Generative modeling, anomaly detection.

6. Contractive Autoencoder

- **Description**: Adds a regularization term to the loss function to encourage the latent space to be robust to small changes in the input.
- **Mathematics**: The regularization term penalizes the Frobenius norm of the Jacobian matrix of the encoder activations with respect to the input:

$$\mathcal{L}_{contractive} = \lambda \|\nabla_{\mathbf{x}} f(\mathbf{x})\|_F^2$$

- **Usage**: Learning robust features, data compression.

Example Walkthrough: Simple Autoencoder for Image Compression

Imagine you have grayscale images of size 28x28 pixels (784 total pixels). The goal is to compress these images into a much smaller size using an autoencoder.

1. **Input Layer**: The image is fed as a 784-dimensional vector.
2. **Encoder**: The network reduces this vector down to, say, a 32-dimensional vector. This 32-dimensional vector represents the compressed features of the image.
3. **Latent Space**: This 32-dimensional vector is the latent representation.
4. **Decoder**: The network then takes this compressed vector and tries to reconstruct the original 784-dimensional image.
5. **Output Layer**: The final output is a 784-dimensional vector, ideally close to the original input image.

During training, the network adjusts its weights to minimize the difference between the original image and the reconstructed image, effectively learning how to compress and decompress the data.

This process helps the autoencoder learn to capture the most important features of the images, reducing the data's dimensionality while preserving its essential characteristics.

Links:

https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73 (https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73): Refer when learning GEN AI.

https://www.jeremyjordan.me/autoencoders/ (https://www.jeremyjordan.me/autoencoders/)