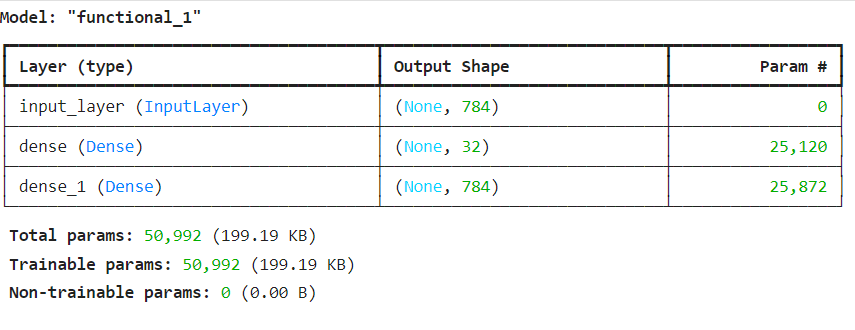
|  |  |
| --- | --- |
| **Ex No: 7**  **Date: 23rd September 2024** | **Implementation of Simple, Deep, CNN and Denoising Autoencoders** |

**Objective:** The main goal of this report is to explore and implement various types of autoencoders for dimensionality reduction, denoising, and image reconstruction. The autoencoders are applied to datasets like MNIST and Fashion MNIST, with different architectures: simple, deep, convolutional, and denoising.

**Description:**

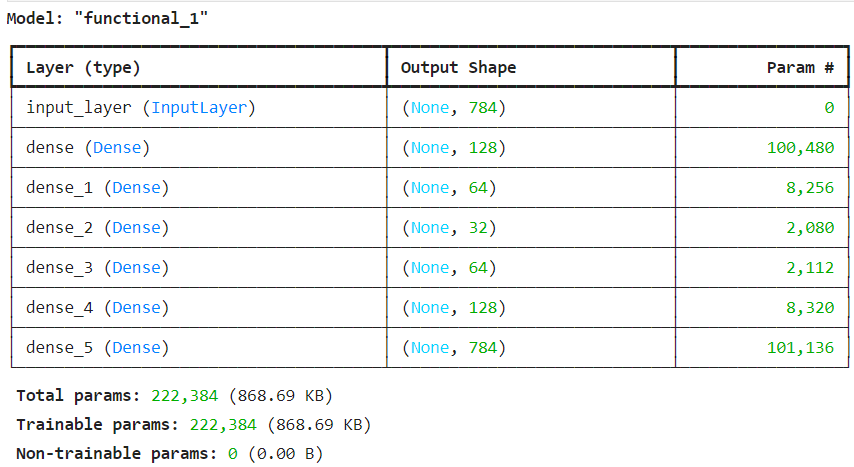
**1. Simple Autoencoder:** A simple autoencoder compresses input data into a latent representation and then reconstructs the original input from this compressed form.

* **Dataset:** MNIST (handwritten digits)
* **Process:** The encoder compresses the 28x28 pixel images into a lower-dimensional space (e.g., 32 units), and the decoder reconstructs the image from this compressed representation.
* **Key Concept:** The network learns the most critical features of the data in an unsupervised manner.



**2. Deep Autoencoder:** A deep autoencoder has multiple layers in both the encoder and decoder, allowing the network to learn complex, hierarchical representations.

* **Dataset:** MNIST
* **Process:** The encoder consists of several hidden layers, progressively reducing the dimensionality. The decoder mirrors this structure, gradually reconstructing the input from the latent space.
* **Key Concept:** The deeper architecture enables the network to learn more abstract and non-linear features of the data, allowing for better compression and reconstruction.



**3. CNN Autoencoder:** CNN autoencoders replace fully connected layers with convolutional layers, which are particularly effective for image data.

* **Dataset:** Fashion MNIST (images of clothing items)
* **Process:** The encoder uses convolutional layers to capture spatial features, and the decoder uses deconvolutional layers to reconstruct the image.
* **Key Concept:** CNNs excel at capturing spatial relationships in the data, making this type of autoencoder ideal for image processing tasks.



**4. Denoising Autoencoder:** A denoising autoencoder is designed to remove noise from data by training the model to reconstruct a clean version from corrupted input.

* **Dataset:** MNIST or Fashion MNIST
* **Process:** The input images are corrupted by adding noise, and the autoencoder is trained to recover the original clean images.
* **Key Concept:** The network learns to filter out noise and irrelevant information, improving the quality of the reconstructed data.

**Building the parts of the algorithm**

**1. Encoder:** The encoder reduces the input data into a compressed latent space. Depending on the autoencoder type:

* **Simple/Deep Autoencoder:** Uses fully connected (dense) layers to reduce the dimensionality.
* **CNN Autoencoder:** Uses convolutional layers to capture spatial hierarchies in the input image.
* **Denoising Autoencoder:** The encoder compresses noisy data into a cleaner latent space representation.

**2. Decoder:** The decoder reconstructs the original input from the compressed latent space.

* **Simple/Deep Autoencoder:** The decoder mirrors the encoder using fully connected layers to recreate the original image.
* **CNN Autoencoder:** The decoder uses deconvolutional (transposed convolution) layers to up-sample and reconstruct the image.
* **Denoising Autoencoder:** The decoder reconstructs the clean version of the input by learning how to reverse the corruptions applied to the data.

**3. Loss Function:** Autoencoders typically use Mean Squared Error (MSE) as the loss function, which measures the reconstruction error between the original input and the reconstructed output.

**4. Training Process:** The training process involves minimizing the reconstruction error using backpropagation. Optimizers such as Stochastic Gradient Descent (SGD) or Adam are employed to update the model’s weights.

**Conclusion:**

Autoencoders are powerful tools for dimensionality reduction, feature extraction, and noise removal, with different architectures catering to specific tasks. Simple and deep autoencoders excel at compressing data while retaining essential features, making them useful for efficient data storage and analysis. Deep autoencoders capture more complex, hierarchical patterns, while CNN autoencoders are particularly suited for image-based tasks due to their ability to capture spatial features. Denoising autoencoders further extend their utility by effectively reconstructing clean data from noisy input. Overall, autoencoders are versatile, enabling efficient data representation and preprocessing in a variety of machine learning applications.

**GitHub Link:** [**https://github.com/aryapg/DeepLearning.git**](https://github.com/aryapg/DeepLearning.git)