Texas A&M University - Commerce

Department of Computer Science

Comparative Analysis of Denoising Autoencoder and Convolutional Neural Networks for MNIST Classification

Bharath Muthuswamy Paran

Supervisor: Derek Harter, Ph.D.

A report submitted in partial fulfilment of the requirements of
Texas A&M University - Commerce for the degree of
Master of Science in *Computer Science*

April 25, 2024

Declaration

I, Bharath Muthuswamy Paran, of the Department of Computer Science, Texas A&M University - Commerce, confirm that this is my own work and figures, tables, equations, code snippets, artworks, and illustrations in this report are original and have not been taken from any other person's work, except where the works of others have been explicitly acknowledged, quoted, and referenced. I understand that if failing to do so will be considered a case of plagiarism. Plagiarism is a form of academic misconduct and will be penalised accordingly.

I give consent to a copy of my report being shared with future students as an exemplar.

I give consent for my work to be made available more widely to members of TAMUC and public with interest in teaching, learning and research.

Bharath Muthuswamy Paran
April 25, 2024

Abstract

In order to enchance MNIST classification this research conducts an in-depth comparative analysis between two powerful architectures: the Denoising Autoencoder (DAE) and the Convolutional Neural Network (CNN). Both models are well-regarded for their distinct capabilities — DAE excels in unsupervised learning and feature extraction, while CNN is tailored for image-based tasks, particularly classification. The purpose of this study is to enhance the performance of MNIST digit classification, this study navigates through the architectures of DAE and CNN, analyzing their impact on classification accuracy and noise resilience. The research extends beyond traditional performance metrics, delving into the interpretability of learned representations and assessing the models' ability to handle noisy input data. The research problem centers on deciphering how these architectures influence MNIST classification under varying levels of noise, providing insights into their respective strengths and limitations. By addressing this research problem, the study aims to contribute to the advancement of image classification techniques, offering nuanced guidance on selecting models for MNIST classification tasks, especially in scenarios with noisy input data. The significance of this research lies in its potential to improve the resilience of classification models to real-world noise, thereby enhancing their applicability in practical settings. The findings are expected to benefit practitioners and researchers alike, guiding them in the selection and optimization of models for noise-affected MNIST classification scenarios.

Keywords: Deep Learning, Denoising Autoencoder, Convolutional Neural Networks, Classification, Accuracy.

Contents

List of Figures

List of Tables

List of Abbreviations

SMPCS	School of Mathematical, Physical and Computational Sciences

Chapter 1

Introduction

The realm of image classification, a complex perceptual task involving the categorization of objects from images, has witnessed significant advancements over the past decade. Referred to as the process of categorizing objects, image classification relies on the sophisticated analysis of multispectral data, utilizing the underlying multispectral pattern of each pixel as a quantitative basis for classification (Lillesand et al., 2015). Notably, there has been a remarkable improvement in classification accuracy, reflecting the evolution of image classification models. In recent times, these models are increasingly applied across diverse fields, showcasing their versatility. Applications range from tasks such as handwritten digit recognition (Ahlawat et al., 2020) and Vehicle detection and classification (Tsai et al., 2018), deep learning approach to pneumonia classification (Stephen et al., 2019), and military object detection (Janakiramaiah et al., 2023). The existing models are broadly categorized into unsupervised and supervised modes, reflecting the diverse approaches employed in addressing the intricate challenges of image classification.

1.1 Background

This research project is focused on enhancing the classification accuracy of the MNIST digit dataset, a widely used benchmark in the field of machine learning. The motivation behind this study is to address the importance of improving the robustness of MNIST digit classification, particularly in scenarios with noisy input data.

The project conducts an in-depth comparative analysis between two powerful neural network architectures: the Denoising Autoencoder (DAE) and the Convolutional Neural Network (CNN). Both models are chosen for their distinct capabilities, where the DAE excels in unsupervised learning and feature extraction, and the CNN is specifically tailored for image-based tasks, including classification.

The research delves into the impact of these architectures on classification accuracy and noise resilience, going beyond traditional performance metrics. It explores the interpretability of learned representations and evaluates the models' ability to handle noisy input data. Algorithms such as Denoising Autoencoders and Convolutional Neural Networks are the core components under investigation.

The project's significance lies in its potential to advance image classification techniques, providing nuanced guidance for selecting models in MNIST classification tasks, especially when dealing with noisy input data. The findings are expected to benefit both practitioners and researchers, offering insights that can guide the selection and optimization of models for real- world, noise-affected MNIST classification scenarios.

1.2 Research Question

The research questions are:

1. What differences exist between denoising autoencoder (DAE) models' performance indicators when it comes to handwritten digit prediction?
2. What are the contributing elements to these variations, especially with respect to the hyperparameters?
3. What role do different searches for meta parameters play in maximising the performance of these two neural networks?

The central research problem revolves around understanding how the DAE and CNN architectures influence MNIST digit classification under varying levels of noise, providing valuable insights into their respective strengths and limitations. The main goal is to identify the aspects that affect accuracy, with a particular emphasis on the effects of noise levels.

1.3 Aims and objectives

Aims: This study's main goal is determine how well a CNN—which is optimised for image- related tasks—can classify handwritten digits in the MNIST dataset when compared to a DAE—which is specifically made for denoising and feature extraction. Also, provide insightful information on the advantages and disadvantages of both architectures on real world noisy images. The main objectives of this study include:

- Gather the MNIST dataset to make it ready for training models.
- Train a convolutional Neural Network (CNN) on the MNIST data, focusing on improving accuracy, precision, recall, F1-score, and understanding confusion between numbers.
- Train a denoising autoencoder (DAE) on the same MNIST data, emphasizing its ability to learn features without labeled information.
- Adjust various parameters of both CNN and DAE to get the best performance.
- Compare how well CNN and DAE perform after adjusting parameters, understanding which one is better for predicting numbers in the MNIST dataset.
- Explore how well both models understand and handle noisy data.
- Test the models' ability to work well when there is some noise in the MNIST dataset.

1.4 Solution approach

The study uses a step-by-step approach, including setting up models, preparing data, splitting the data, adjusting key parameters, and evaluating performance. The research achieves its goals by thoroughly exploring different ways to find the best model hyperparameters.

1.4.1 Data Description:

The MNIST dataset (Tensorflow, 1994) is a widely used collection of handwritten digit images commonly employed in the field of machine learning and computer vision. The dataset consists of digits in the images range from 0 to 9, and each image is labeled with its corresponding digit. This dataset serves as a fundamental benchmark for developing and testing image classification algorithms, particularly those focused on recognizing and classifying handwritten digits.

1.4.2 Data Preprocessing:

Data preprocessing plays a crucial role in optimizing the MNIST dataset for effective deep learning model training. Here are the few steps involved in preparing the MNIST dataset:

1.4.3 Handling Missing or Noisy Data:

Checking for any missing or corrupted data points within the dataset. Addressing any noise or outliers that might affect the model's performance.

1.4.4 Reshaping and Normalization:

Reshaping the images to a standard format. For MNIST, this often involves converting 28x28 pixel images into a flattened array of 784 pixels. Normalize the pixel values to a range between 0 and 1. This ensures consistent scaling and aids in faster convergence during model training.

1.4.5 Model Implementation:

Convolutional Neural Network (CNN) Training:

Implementing and training a standard CNN on the labeled MNIST dataset. Focused on optimizing key performance metrics: accuracy, precision, recall, F1-score, and confusion matrix.

Denoising Autoencoder (DAE) Training:

Training a competitive Denoising Autoencoder on the same MNIST dataset. Leverage DAE's unsupervised learning and feature extraction capabilities.

1.4.6 Performance Evaluation:

Evaluating and compare the performance metrics of CNN and DAE models. Assess strengths and limitations, providing insights into model suitability for MNIST digit classification.

1.4.7 Interpretability and Noisy Data Handling:

Investigate the interpretability of learned representations from both models. Assess models' resilience in handling noisy input data.

1.5 Summary of contributions and achievements

Describe clearly what you have done/created/achieved and what the major results and their implications are.

Chapter 2

Literature Review

This section explores the existing existing body of knowledge surrounding the Comparative Analysis of Denoising Autoencoder (DAE) and Convolutional Neural Networks (CNN) for MNIST Classification. Recent advancements in deep machine learning have significantly improved feature learning in image processing tasks, particularly in image classification, making it imperative to explore and understand the capabilities and limitations of different models. Notably, Convolutional Neural Networks (CNNs) have emerged as powerful tools, showcasing superior performance in image classification tasks. However, the inherent challenge of noise in real-world data poses a significant constraint on the performance of deep neural networks, prompting the exploration of denoising techniques. The literature review will examine prior studies focusing on image denoising methods, including traditional approaches and those leveraging deep learning, with a specific emphasis on Denoising Autoencoders. Understanding the existing landscape is crucial for contextualizing the current research and identifying gaps that the Comparative Analysis seeks to address.

2.1 Review of state-of-the-art

The (Guo et al., 2017) explores the application of deep learning, specifically Convolutional Neural Networks (CNNs), in image classification. The authors discuss various models commonly used in deep learning, such as Auto Encoder, sparse coding, Restricted Boltzmann Machine, Deep Belief Networks, and CNNs. They emphasize the effectiveness of CNNs in image classification, particularly showcasing their high performance. The study involves building a simple CNN for image classification, with experiments conducted on benchmark datasets like minist and cifar-10. The authors delve into the fundamental components of CNNs, including Convolutional layers, pooling layers, and fully-connected layers. The study focuses on learning rate strategies and optimization algorithms for solving optimal parameters in image classification. Different strategies are compared, demonstrating the impact on recognition rates during training and testing. The experimental results and

The (Bajaj et al., 2020) proposes an efficient image denoising technique using autoencoders based on deep learning models. The motivation behind image denoising is to eliminate noise from images, which can be caused by various factors such as defects in camera sensors, transmission in noisy channels, or faulty memory locations in hardware. The proposed model utilizes autoencoders, a type of artificial neural network, for image denoising. Autoencoders learn noise patterns from training images and attempt to eliminate noise from novel images. The proposed network architecture consists of convolutional denoising autoencoder (CDA) blocks, each containing internal layers like convolution, pooling, deconvolution, and upsampling. Skip connections are employed to enhance the model's ability to capture finer image details and facilitate back-propagation during training. The paper discusses related work, highlighting methods such as non-local means, Block Matching and 3D Matching (BM3D), and denoising autoencoders. STL-10 dataset for training and the SET5 standard image dataset for testing. Performance metrics, such as Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM), are used to evaluate the proposed model's effectiveness in comparison to baseline models. The results show that the proposed model outperforms Convolutional Denoising Deep Neural Network (CDDNN) and Residual Encoder Decoder with 30 layers (RED30) in terms of PSNR, indicating superior denoising capabilities.

2.2 Convolutional Neural Networks

CNNs are a variant of multi-layered neural networks designed for processing two-dimensional data, particularly well-suited for classifying visual patterns like pixel images with minimal pre-processing. Feature extraction from observed data occurs at each layer by applying digital filtering techniques, allowing information to propagate through multiple layers.

THe CNN architecture, introduced by LeCun et al. in 1998, are applied to classify images and perceive visual patterns directly from pixel images. Information propagation throughout multiple layers enables feature extraction using digital filtering techniques. CNNs perform two main processes: convolution and subsampling. The convolution involves applying a small-sized kernel over the input feature map (IFM) to produce a convolved feature map (CFM). CFMs are computed by applying the convolutional operation over the original input image using a kernel, which represents weights and biases. Weights are shared among positions, preserving spatial locality during the convolution process. The subsampling simplifies the feature map gained from convolution by selecting significant features and discarding the rest. Max-pooling, a subsampling method used in experiments, involves taking the maximum value over non-overlapping sub-regions.

2.3 Denoising Auto-Encoder

Denoising Autoencoder (DAE) is a specialized type of artificial neural network designed to clean up noisy or corrupted data. It belongs to the family of autoencoders, which are neural networks trained to learn efficient representations of input data. DAEs, in particular, excel in removing noise and capturing essential features from corrupted input.

The encoder is the first part of the DAE. Its role is to transform the input data into a compressed representation. This compressed representation should ideally retain the essential features of the input while filtering out noise. One distinctive feature of DAE is the intentional introduction of noise into the input data. This could involve adding random variations or distortions to simulate real-world noise or corruption.The decoder is the second part of the DAE. It takes the compressed representation from the encoder and reconstructs the clean version of the original input data. The decoder's task is to recover the essential features and remove the injected noise.

2.4 Critique of the review

The literature review provides a comprehensive overview of the existing body of knowledge on the Comparative Analysis of Denoising Autoencoder (DAE) and Convolutional Neural Networks (CNN) for MNIST Classification. The review is structured logically, beginning with an introduction to the significance of the topic and progressing to a review of state-of-the-art research.

- Clear Problem Statement: The literature review effectively establishes the problem by highlighting the challenges posed by noise in real-world data for deep neural networks. The motivation to explore denoising techniques is well justified in the context of advancements in deep machine learning and the superior performance of CNNs in image classification.
- Thorough State-of-the-Art Review: The review of state-of-the-art research is well-detailed, discussing relevant models in deep learning for image classification, such as Auto Encoder, sparse coding, Restricted Boltzmann Machine, Deep Belief Networks, and CNNs. The exploration of fundamental components of CNNs, learning rate strategies, and optimization algorithms adds depth to the understanding of the field.
- Comprehensive Denoising Technique Overview: The literature appropriately includes a discussion on denoising techniques, with a specific emphasis on the proposed model using autoencoders. The incorporation of related work, including traditional methods like non-local means and modern approaches like denoising autoencoders, enriches the background information.
- Methodology Explanation: The detailed description of the proposed image denoising technique using convolutional denoising autoencoder (CDA) blocks, skip connections, and performance metrics (PSNR and SSIM) adds transparency to the methodology.
- Effective Comparison of Models: The comparative analysis between the proposed model and existing models like Convolutional Denoising Deep Neural Network (CDDNN) and Residual Encoder Decoder with 30 layers (RED30) is valuable. The use of performance metrics provides a quantitative basis for comparison.

2.5 Summary

In this literature review, the Comparative Analysis of Denoising Autoencoder (DAE) and Convolutional Neural Networks (CNN) for MNIST Classification is explored within the context of recent advancements in deep machine learning. Convolutional Neural Networks (CNNs) have shown superior performance in image classification tasks, but the challenge of noise in real-world data necessitates the investigation of denoising techniques. The review critically examines prior studies on image denoising, with a specific focus on Denoising Autoencoders.

The state-of-the-art review discusses the application of CNNs in image classification, emphasizing their effectiveness. It covers various deep learning models, including Auto Encoder, sparse coding, Restricted Boltzmann Machine, and Deep Belief Networks, with experiments conducted on benchmark datasets. Another study proposes an image denoising technique using autoencoders, demonstrating its effectiveness through metrics like Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM). The proposed model outperforms existing models, showcasing superior denoising capabilities. The comprehensive overview of denoising techniques, methodology transparency, and effective comparison of models contribute to the overall strength of the literature review. This groundwork sets the stage for the Comparative Analysis, aiming to address gaps in understanding and provide insights into the capabilities and limitations of DAEs and CNNs for MNIST Classification.

Chapter 3

Methodology

3.1 Dataset Overview and Data Exploration

The MNIST dataset serves as a fundamental benchmark in the field of machine learning research, particularly in tasks related to image classification and recognition. This dataset consists of a collection of grayscale images of handwritten digits (0-9), each accompanied by its corresponding label indicating the digit it represents. Researchers widely utilize the MNIST dataset to evaluate the efficacy of various algorithms and techniques for digit recognition tasks.

The MNIST dataset comprises 70,000 grayscale images, divided into a training set of 60,000 images and a test set of 10,000 images. Each image is a 28x28 pixel square, resulting in a total of 784 pixels per image. These pixels represent the grayscale intensity, ranging from 0 (black) to 255 (white). Additionally, each image is associated with a label, providing ground truth information about the digit depicted in the image.

Table 3.1: Dataset Attributes and Data Types for MNIST Dataset

Attribute	Description	Data Type
Image	Handwritten digit image	Image (28x28 pixels, grayscale)
Label	Digit label	Integer (0 to 9)

3.1.1 Loading the Dataset

The MNIST dataset will be retrieved from reputable sources, such as TensorFlow or PyTorch libraries, ensuring data integrity and consistency. Both the training and testing datasets will be loaded into the research environment for further exploration.

3.1.2 Visualizing Sample Images

A random selection of sample images from the dataset will be visualized to gain insights into the characteristics of handwritten digits. These images will be displayed alongside their corresponding labels, facilitating an understanding of the dataset's content.

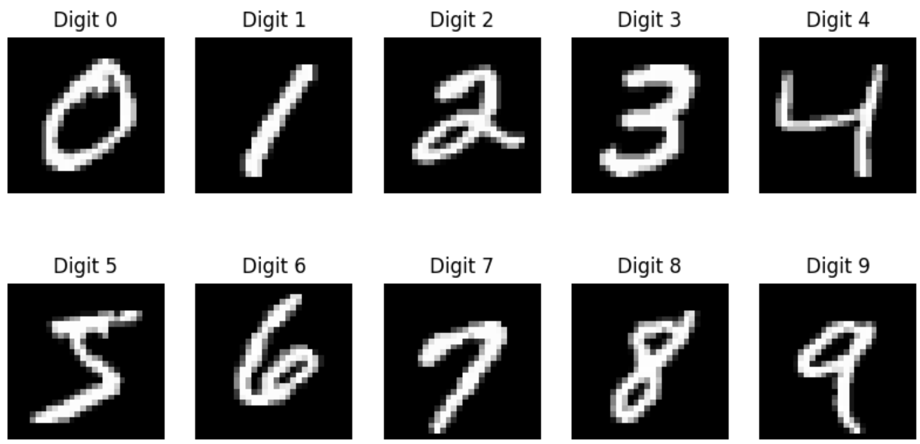


Figure 3.1: MNIST Example Images

3.1.3 Exploring Label Distribution

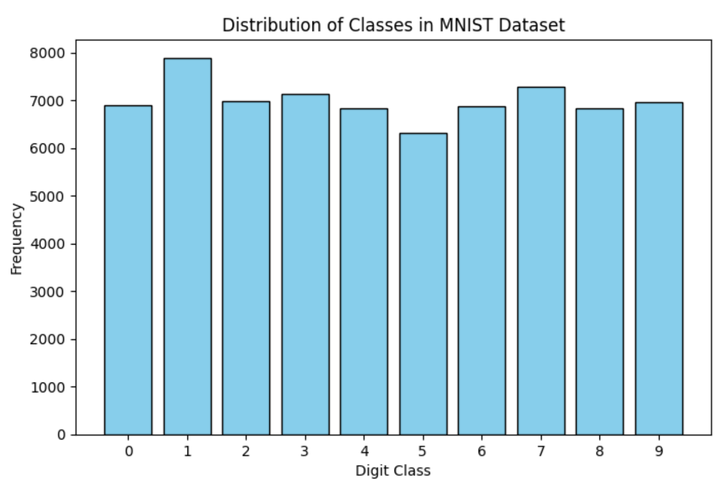


Figure 3.2: Distribution of Classes in MNIST Dataset

A histogram is plotted to visualize the distribution of digit labels within the training set. It is imperative to ensure that the dataset exhibits balanced label distributions, with a similar number of samples for each digit class.

3.2 Data Preprocessing

3.2.1 Flatten the image data

In the MNIST dataset, each image represents a handwritten digit and is initially structured as a 2-dimensional array (28x28 pixels). However, most machine learning algorithms, including neural networks, require input data to be in a flat, one-dimensional format. Flattening the image data involves reshaping each image into a single vector by stacking the rows of pixels one after another. This transformation simplifies the representation of the image data and allows us to treat each pixel as a separate feature. Flattening the data ensures compatibility with various machine learning models and facilitates efficient computation during training and prediction processes.

3.2.2 Normalize the pixel values:

The pixel values in the MNIST dataset represent the intensity of grayscale pixels ranging from 0 to 255. Normalizing the pixel values involves scaling them to a standard range, typically between 0 and 1 or -1 and 1. Normalization helps in stabilizing and speeding up the training process of machine learning models. It prevents certain features from dominating others due to differences in scale, thereby improving the convergence of optimization algorithms. Moreover, normalization ensures that the model's performance is not sensitive to the absolute values of the input features, making it more robust and generalizable across different datasets.

3.2.3 Split the dataset into training and testing sets:

Splitting the dataset into training and testing sets is crucial for evaluating the performance of machine learning models. The training set is used to train the model, while the testing set is used to evaluate its performance on unseen data. Additionally, it is common practice to further split the training set into a training subset and a validation set. The training subset is used for actual model training, while the validation set is used to tune hyperparameters and monitor the model's performance during training. This split helps prevent overfitting by providing an independent dataset for model evaluation and parameter tuning. Overall, splitting the dataset ensures that the model's performance estimates are reliable and generalizable to unseen data.

3.2.4 Adding Noise to MNIST Images

The add noise function provides a convenient way to induce varied levels of Gaussian noise into the MNIST dataset. This function takes two parameters: images, which represents the array of MNIST images, and noise level, which specifies the variance of the Gaussian noise to be added. The higher the noise level, the more intense the noise added to the images.

Internally, the function generates Gaussian noise with the specified variance and adds it to the input images. It ensures that the pixel values remain within the valid range of [0, 255] by clipping the values after adding noise. By incorporating this function into the preprocessing pipeline, we can create augmented versions of the MNIST dataset with different levels of noise. These augmented datasets can help improve the model's performance, especially in scenarios where the input data may exhibit variability or uncertainty.

3.3 Model Implementation

This is the critical step where we build the model to predict handwritten digits.

3.3.1 Convolutional Neural Network

It is used for accurately classifying handwritten digits in the MNIST dataset due to their ability to effectively capture spatial patterns within images.

Algorithm 1 Handwritten Digit Classification using Convolutional Neural Networks

Input: Images X and corresponding labels y for training data
Output: Performance metrics including accuracy, confusion matrix, precision, recall, and F1-score
1: Create a convolutional neural network model with default architecture
2: Train the model using the training data (images X and labels y)
3: Evaluate the model's performance metrics such as accuracy, confusion matrix, precision, recall, and F1-score
4: Predict the labels for the testing data using the trained model
5: Assess the model's performance on the testing data
6: Calculate the performance metrics including accuracy, confusion matrix, precision, recall, and F1-score.

3.3.2 Noise Function

Algorithm 2 Add Gaussian Noise to MNIST Images

1: **Input:** MNIST images array X_{clean}, noise level σ^2
2: **Output:** Noisy MNIST images array X_{noisy}
3: Initialize the noise array $N \sim \mathcal{N}(0, \sigma^2)$ with the same shape as X_{clean}
4: $X_{\text{noisy}} \leftarrow X_{\text{clean}} + N$ ▷ Add Gaussian noise to the clean images
5: $X_{\text{noisy}} \leftarrow \text{clip}(X_{\text{noisy}}, 0, 1)$ ▷ Clip values to ensure they remain between 0 and 1
6: **return** X_{noisy}

3.3.3 Handwritten Digit Classification using Denoising Auto-encoder

It is employed for enhancing the robustness of feature extraction in MNIST dataset classification tasks by reconstructing clean images from noisy inputs.

Algorithm 3 Denoising Autoencoder

Input: Noisy images X_{noisy} and clean images X_{clean} for training data
Output: Reconstructed clean images from noisy inputs
1: Create a denoising autoencoder model with encoder and decoder architecture
2: Train the denoising autoencoder using the noisy images X_{noisy} and their corresponding clean images X_{clean}
3: Evaluate the performance of the denoising autoencoder by measuring reconstruction error
4: Use the trained denoising autoencoder to reconstruct clean images from noisy inputs

3.4 Evaluation Metrics

In Deep learning, evaluation metrics are used to measure the performance of models, particularly in tasks like classification and regression. These metrics help in understanding how well a model is performing and are crucial for comparing different models or tuning the same model with different hyperparameters. Below is a brief description of some common evaluation metrics used in classification tasks.

3.4.1 Accuracy

Accuracy measures the fraction of predictions our model got right. It is the simplest metric for evaluation but can be misleading if the classes are imbalanced.

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

3.4.2 Confusion Matrix

A confusion matrix is a table used to describe the performance of a classification model. It provides insights not only into the errors being made but also the types of errors that are occurring.

	Predicted Positive	**Predicted Negative**
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

3.4.3 Reconstruction Mean Squared Error

Reconstruction MSE is a common metric used to evaluate the performance of image reconstruction models, particularly in contexts such as denoising or super-resolution. It measures the average squared difference between the original clean images and their reconstructed counterparts.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(X_{\text{original},i} - X_{\text{reconstructed},i})^2$$

3.5 Summary

The methodology section outlines the research approach, beginning with an overview of the MNIST dataset and exploratory data analysis. It covers data preprocessing steps like flattening and normalizing image data, along with splitting the dataset. Model implementation details for Convolutional Neural Networks (CNNs) and Denoising Autoencoders (DAEs) are provided, emphasizing their roles in digit classification and image denoising, respectively. Overall, it serves as a roadmap for conducting the research, detailing the steps involved in data preparation, model implementation, and hyperparameter optimization for accurate classification and denoising of MNIST handwritten digits.

Chapter 4

Results

4.1 Results for CNN

Evaluating the performance of a Convolutional Neural Network (CNN) subjected to varying levels of Gaussian noise, using accuracy as the primary metric. The analysis uses three datasets: training, validation, and testing, to observe how the CNN copes with noise during different phases of model usage.

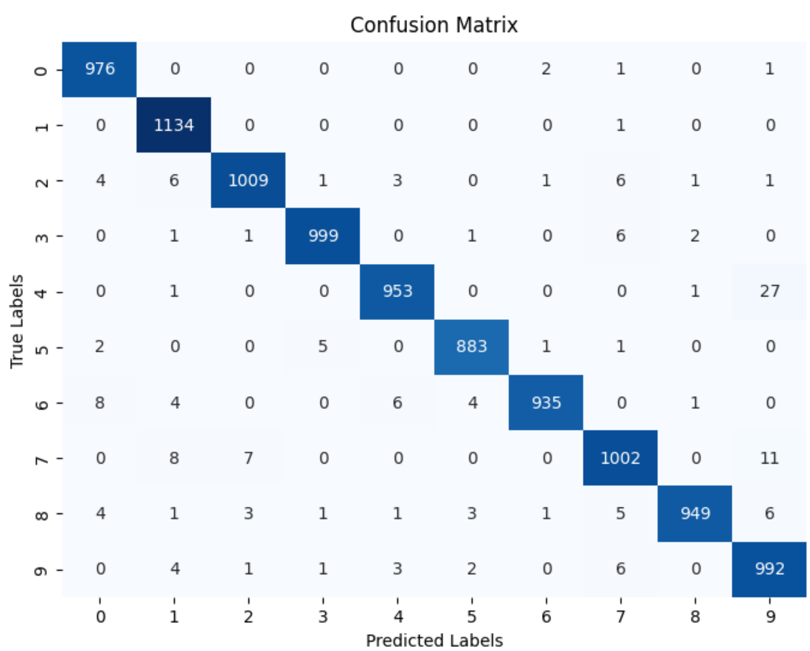


Figure 4.1: Confusion Matrix for Noise level 0.5

The testing accuracy starts at a high of 98.99% with minimal noise and remains robust up to a noise level of 0.5.

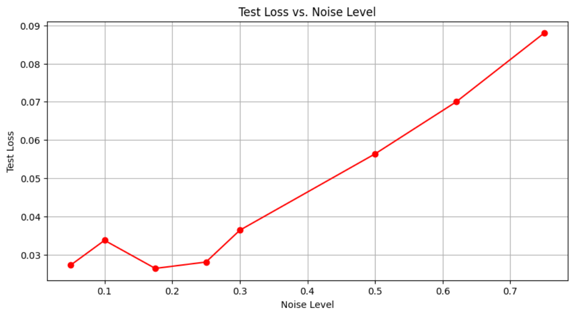


Figure 4.2: CNN Test Loss

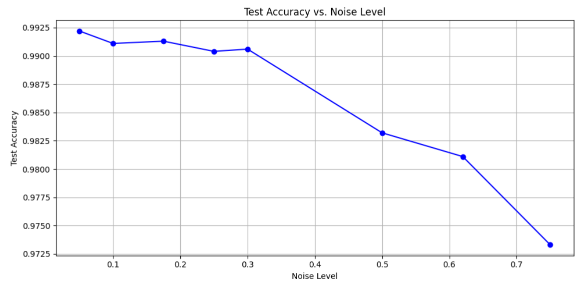


Figure 4.3: CNN Test Accuracy

CNN	Noise Level							
	0.05	0.1	0.175	0.25	0.3	0.5	0.62	0.75
Training	99.56%	99.55%	99.65%	99.42%	99.18%	98.35%	97.17%	93.88%
Validation	98.65%	98.77%	99.02%	98.07%	98.08%	96.28%	94.46%	89.99%
Testing	98.99%	99.07%	98.98%	98.91%	98.84%	98.54%	98.46%	97.74%

Table 4.1: CNN Accuracy Across Different Noise Levels

4.2 Results for DAE

The performance of a Denoising Autoencoder (DAE) when exposed to varying levels of Gaussian noise, evaluating its efficiency using reconstruction mean squared error (MSE) for assessing the quality of image reconstruction and training and validation losses to gauge model learning performance.

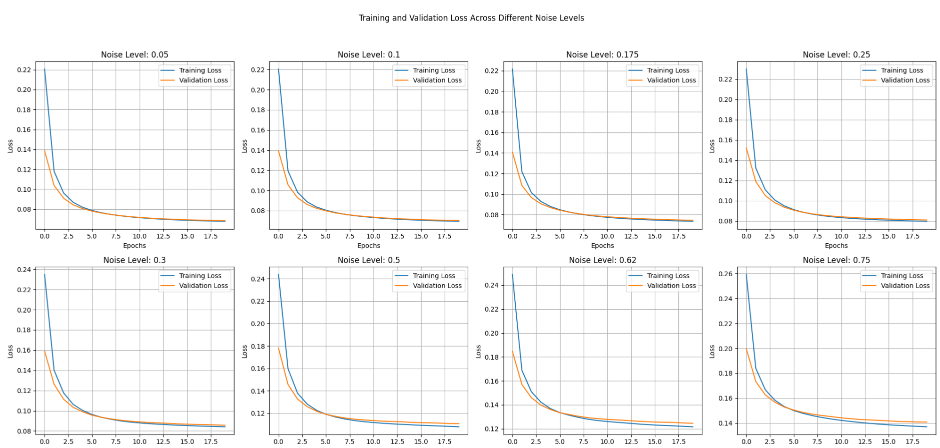


Figure 4.4: Training & Validation Loss on various noise levels

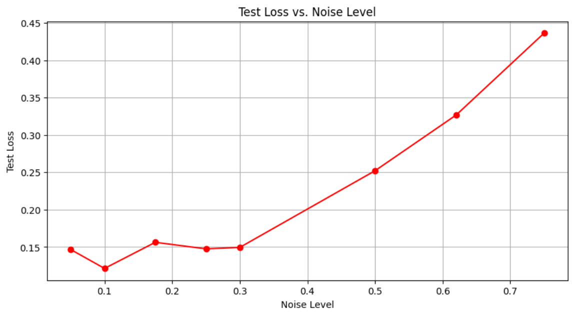


Figure 4.5: DAE Classifier Test Loss

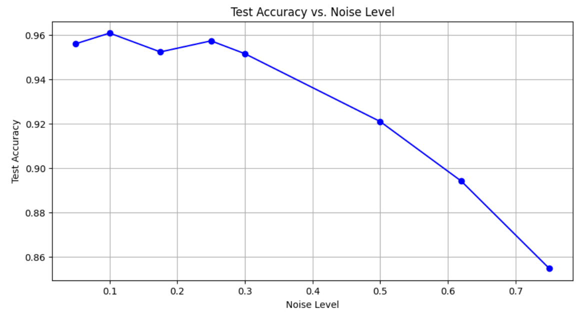


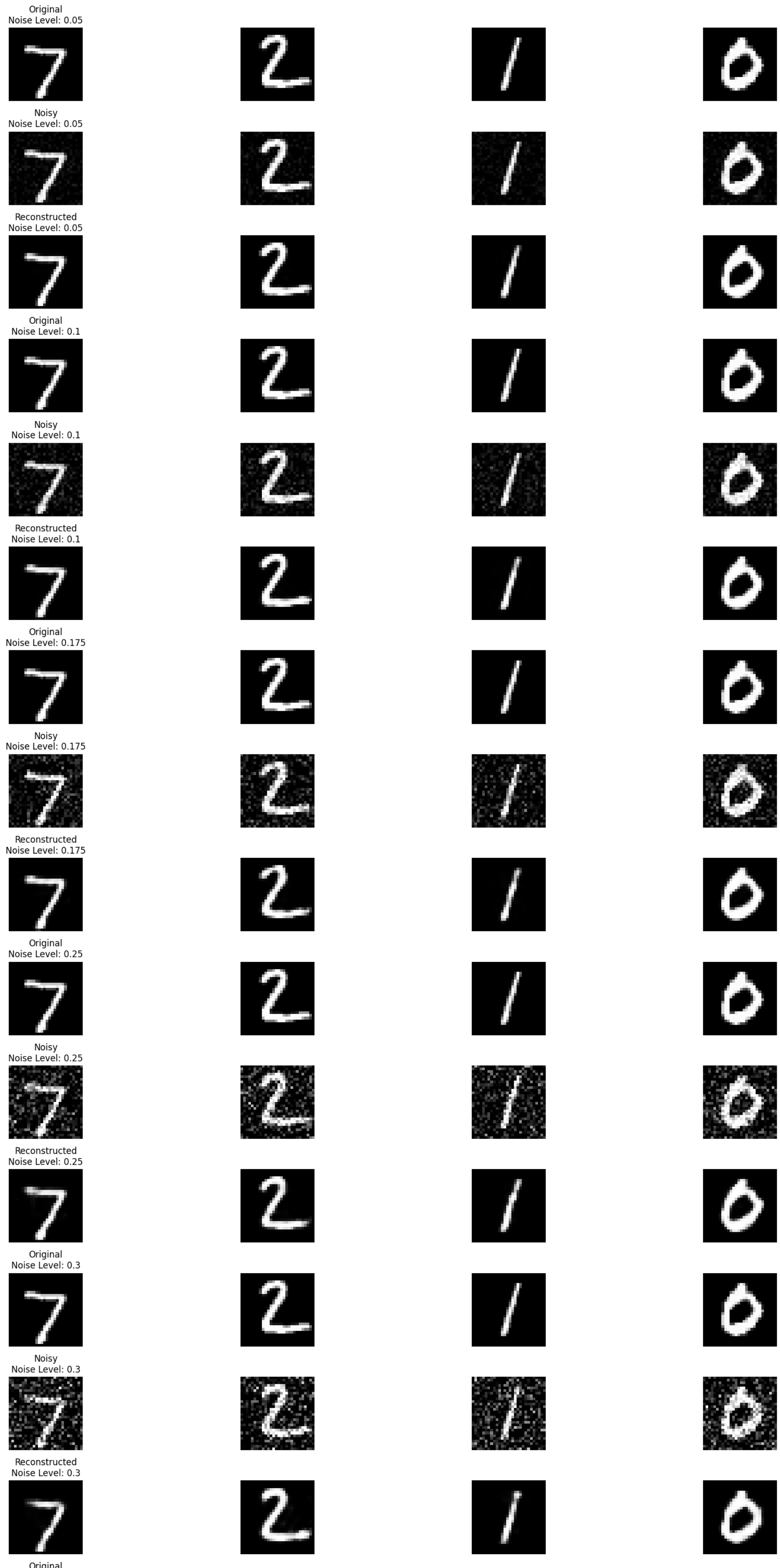
Figure 4.6: DAE Classifier Test Accuracy

Noise Level	Reconstruction MSE	Training Loss	Validation Loss
0.05	0.0021	0.0678	0.0684
0.10	0.0026	0.0695	0.0702
0.175	0.0038	0.0735	0.0746
0.25	0.0057	0.0796	0.0809
0.30	0.0072	0.0841	0.0857
0.50	0.0148	0.1079	0.1106
0.62	0.0195	0.1216	0.1245
0.75	0.0249	0.1386	0.1418

Table 4.2: Performance of Denoising Autoencoder (DAE) at Different Noise Levels

Image Reconstruction using DAE

The image showcases a Denoising Autoencoder's (DAE) performance in cleaning digit images across a spectrum of Gaussian noise intensities. The format is arranged in a descending order of noise levels, with original images at the top, followed by noisy versions, and concluding with the DAE's reconstructed outputs at the bottom of each set. At lower noise levels, the DAE effectively restores the digits to a state closely resembling the originals. However, as noise intensifies, the quality of reconstruction degrades, revealing the DAE's decreasing ability to accurately denoise the images, especially when the noise reaches the highest levels. This visual demonstration effectively highlights the DAE's strengths in mitigating noise and its challenges in handling extreme noise conditions.


Original
Noise Level: 0.05
Noisy
Noise Level: 0.05
Reconstructed
Noise Level: 0.05
Original
Noise Level: 0.1
Noisy
Noise Level: 0.1
Reconstructed
Noise Level: 0.1
Original
Noise Level: 0.175
Noisy
Noise Level: 0.175
Reconstructed
Noise Level: 0.175
Original
Noise Level: 0.25
Noisy
Noise Level: 0.25
Reconstructed
Noise Level: 0.25
Original
Noise Level: 0.3
Noisy
Noise Level: 0.3
Reconstructed
Noise Level: 0.3
Original

As shown in 4.3 table presents the classification test accuracy of a Denoising Autoencoder (DAE) classifier at various noise levels, illustrating how the DAE's performance is affected as the noise level increases. Initially, at low noise levels (0.05 to 0.30), the DAE maintains high accuracy, slightly fluctuating but staying above 95%. This indicates the DAE's strong capability to accurately classify images with minimal to moderate noise. However, as the noise level rises to 0.50 and beyond, there is a clear downward trend in accuracy, with a notable drop to 85.47% at a noise level of 0.75. These figures underscore the challenge that higher noise levels pose to the DAE's classification ability, highlighting a decrease in performance as the noise approaches higher intensities.

Noise Level	Accuracy (%)
0.05	95.60%
0.10	96.09%
0.175	95.24%
0.25	95.74%
0.30	95.16%
0.50	92.10%
0.62	89.41%
0.75	85.47%

Table 4.3: Classification Test Accuracy of DAE Classifier at Various Noise Levels

4.3 Summary

The robustness and adaptability of both the Convolutional Neural Network (CNN) and the Denoising Autoencoder (DAE) when challenged by varying levels of Gaussian noise. The CNN maintains high accuracy across training, validation, and testing phases with minimal degradation until noise levels reach 0.5, demonstrating effective noise resilience. Conversely, the DAE, assessed through reconstruction MSE and training/validation losses, shows a gradual increase in error as noise levels rise, indicating a decline in its ability to reconstruct images precisely at higher noise intensities. Additionally, the classification accuracy of a DAE-based classifier also declines as noise increases, further illustrating the challenges faced by DAEs in maintaining performance under severe noise conditions. Together, these results highlight the strengths and limitations of both models in noise-affected environments, guiding potential improvements for noise robustness in neural network applications.

Chapter 5

Discussion and Analysis

5.1 Discussion on performance on noise levels

The performance of the Convolutional Neural Network (CNN) and the Denoising Autoencoder (DAE) across various Gaussian noise intensities reveals important insights into their operational resilience and constraints. The CNN showed commendable performance in maintaining high accuracy levels until a moderate noise intensity (0.5), beyond which a significant drop in accuracy was observed. This indicates that CNNs, with appropriate training, can efficiently process data in environments with considerable noise, which is essential for applications such as real-time surveillance or autonomous vehicle navigation.

On the other hand, the DAE demonstrated excellent initial performance in reducing noise at lower levels, as evidenced by low reconstruction MSE scores. However, as the noise intensity escalated, the effectiveness of the DAE diminished markedly, highlighted by rising MSE values. This decline might reflect the model's limitations in adapting to more complex or intense noise patterns, or a saturation in its data modeling capabilities.

5.2 Significance of the findings

The significance of these findings is underscored by demonstrating how various neural network architectures navigate the challenge of environmental noise, a prevalent issue in processing real-world data. Notably, the CNN's ability to sustain high accuracy up to moderate noise levels illustrates its suitability for critical applications such as autonomous vehicle navigation, where real-time image processing is essential and environmental conditions are unpredictable. This resilience suggests that CNNs could serve as a cornerstone technology for systems that demand consistent reliability across diverse sensory inputs.

For the Denoising Autoencoder (DAE), the notable reduction in reconstruction errors at lower noise intensities confirms its utility in enhancing image quality, critical for tasks such as digital photo restoration or improving the clarity of medical images, where precision is vital for accurate diagnosis. Additionally, the DAE's ability to improve image quality before further analytical processing can lead to more precise outcomes in subsequent image-processing tasks like object detection or classification.

5.3 Limitations

Despite the promising results, several limitations must be acknowledged, which could impact the generalizability and scalability of the findings:

- Noise Diversity: The study's focus on Gaussian noise does not fully encapsulate the complexity of real-world scenarios, which often involve a variety of noise types including Poisson, speckle, and motion blur. This limitation underscores the need to test these models against a broader spectrum of noise conditions to ensure their robustness across different environments.
- Model Scalability: The increasing reconstruction MSE observed in the DAE as noise levels rise suggests potential scalability issues under extreme noisy conditions. This could limit the DAE's applicability in environments where noise is not only high but also diverse in nature, potentially impacting the model's effectiveness in broader applications.
- Overfitting Concerns: The CNN's consistent performance up to a certain noise threshold might mask underlying overfitting issues, where the model is potentially tuned to specific noise characteristics rather than capturing more generalized patterns. Such overfitting could undermine the model's performance in new or varied operational settings where noise characteristics differ from the training data.
- Computational Efficiency: Both CNNs and DAEs, especially as the models increase in complexity to handle higher levels of noise, place significant demands on computational resources. This could pose challenges in deploying these models in resource-constrained environments, potentially limiting their usability in real-time applications or on edge devices.

5.4 Summary

The analysis of Convolutional Neural Networks (CNNs) and Denoising Autoencoders (DAEs) under varying levels of Gaussian noise revealed their respective strengths and limitations in noisy environments. CNNs maintained robust performance up to moderate noise levels, indicating their practical utility, but showed a performance drop at higher noise intensities. DAEs effectively reduced noise at lower levels but struggled with higher noise, pointing to scalability issues. Both models faced challenges including overfitting and computational efficiency, emphasizing the need for further research to enhance their adaptability and efficiency in diverse real-world settings.

Chapter 6

Conclusions and Future Work

6.1 Conclusion

This investigation into the capabilities of Denoising Autoencoders (DAEs) and Convolutional Neural Networks (CNNs) within the scope of MNIST digit classification has yielded illuminating insights. CNNs have proven resilient, upholding accuracy in the face of noise, making them invaluable for real-world applications where data corruption is inevitable. DAEs have been successful in their primary role of image denoising at lower noise levels, asserting their importance in the initial stages of image analysis workflows. Nevertheless, the study has also highlighted critical limitations, such as the DAE's performance downturn at high noise levels and the CNN's vulnerability to overfitting, which may hinder performance in variable noise environments.

6.2 Future Directions for Research

After completing this project, i have identified several key areas that future research can expand upon, based on our experiences and the project's untapped potential:

- Testing with Different Noises: Our work primarily dealt with Gaussian noise, but there's a whole range of noises in real life that we haven't yet explored. Future projects could look at these other noises, like the random noise from camera sensors, to make our findings more useful for everyday situations.
- Improving the Models: While our DAEs and CNNs did a good job, they struggled with lots of noise. We could try out new types of networks that might do better in these tough conditions, or we could combine features from different models to create something even stronger.
- New Ways to Prevent Overfitting: We used some standard tricks to stop our CNNs from overfitting, but we think there could be better methods out there. Maybe we can take inspiration from how the human brain avoids overfitting to improve our models.

- Making Models Faster and Smaller: Our models are pretty big and slow, which could be a problem for using them on phones or other small devices. Future research could work on making the models more compact without losing their accuracy.
- Trying the Models in New Areas: We'd love to see how our models perform in different fields like medical imaging or factory quality control, where it's really important to get accurate results even when the images aren't perfect.
- Learning from Other Fields: There might be things we can learn from areas like psychology or neuroscience about how to handle noisy data better. By bringing in knowledge from these fields, we could create smarter and more efficient models.
- Adjusting to Noise on the Fly: Right now, our models don't do well if the noise in the images changes unexpectedly. We hope future projects will create models that can adjust themselves as the noise changes, keeping their performance up without needing help.
- Teaching Models to Keep Learning: Our models learn from the noise once and then stop. It would be great if we could teach them to keep learning over time so they can get better at dealing with noise the more they're exposed to it.
- Quickly Adapting to New Noises: Sometimes, models need to learn new types of noise fast. Research into meta-learning, which is about learning how to learn, could help our models pick up new noises quickly, which is especially handy when dealing with new or evolving types of data.

In short, I have learned a lot from this project, but there's still a lot I have not covered. Future research can pick up where I left off, helping to make our findings more applicable to the real world and improving the technology I have developed.

Chapter 7

Reflection

Write a short paragraph on the substantial learning experience. This can include your decision-making approach in problem-solving.

Some hints: You obviously learned how to use different programming languages, write reports in LaTeXand use other technical tools. In this section, we are more interested in what you thought about the experience. Take some time to think and reflect on your individual project as an experience, rather than just a list of technical skills and knowledge. You may describe things you have learned from the research approach and strategy, the process of identifying and solving a problem, the process research inquiry, and the understanding of the impact of the project on your learning experience and future work.

Also think in terms of:

- what knowledge and skills you have developed
- what challenges you faced, but was not able to overcome
- what you could do this project differently if the same or similar problem would come
- rationalize the divisions from your initial planed aims and objectives.

A good reflective summary could be approximately 300–500 words long, but this is just a recommendation.

Note: The next chapter is "**References**," which will be automatically generated if you are using BibTeX referencing method. This template uses BibTeX referencing. Also, note that there is difference between "References" and "Bibliography." The list of "References" strictly only contain the list of articles, paper, and content you have cited (i.e., refereed) in the report. Whereas Bibliography is a list that contains the list of articles, paper, and content you have cited in the report plus the list of articles, paper, and content you have read in order to gain knowledge from. We recommend to use only the list of "References."

References

Ahlawat, S., Choudhary, A., Nayyar, A., Singh, S. and Yoon, B. (2020), 'Improved handwritten digit recognition using convolutional neural networks (cnn)', *Sensors* **20**(12), 3344.

Bajaj, K., Singh, D. K. and Ansari, M. A. (2020), 'Autoencoders based deep learner for image denoising', *Procedia Computer Science* **171**, 1535–1541.

Guo, T., Dong, J., Li, H. and Gao, Y. (2017), Simple convolutional neural network on image classification, *in* '2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)', pp. 721–724.

Janakiramaiah, B., Kalyani, G., Karuna, A., Prasad, L. N. and Krishna, M. (2023), 'Military object detection in defense using multi-level capsule networks', *Soft Computing* **27**(2), 1045–1059.

Lillesand, T., Kiefer, R. W. and Chipman, J. (2015), *Remote sensing and image interpretation*, John Wiley & Sons.

Stephen, O., Sain, M., Maduh, U. J., Jeong, D.-U. et al. (2019), 'An efficient deep learning approach to pneumonia classification in healthcare', *Journal of healthcare engineering* **2019**.

Tensorflow (1994), 'Mnist dataset'. (accessed January 25, 2024).
URL: *https://www.tensorflow.org/datasets/catalog/mnist*

Tsai, C.-C., Tseng, C.-K., Tang, H.-C. and Guo, J.-I. (2018), Vehicle detection and classification based on deep neural network for intelligent transportation applications, *in* '2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)', IEEE, pp. 1605–1608.

Appendix A

An Appendix Chapter (Optional)

Some lengthy tables, codes, raw data, length proofs, etc. which are **very important but not essential part** of the project report goes into an Appendix. An appendix is something a reader would consult if he/she needs extra information and a more comprehensive understating of the report. Also, note that you should use one appendix for one idea.

An appendix is optional. If you feel you do not need to include an appendix in your report, avoid including it. Sometime including irrelevant and unnecessary materials in the Appendices may unreasonably increase the total number of pages in your report and distract the reader.

Appendix B

An Appendix Chapter (Optional)

...