Jurnal Elektro
p-ISSN: 1979-9780
e-ISSN: 2746-4288

Jurnal Elektro Vol.18 No.1 April 2025
journal homepage: https://ejournal.atmajaya.ac.id/index.php/JTE

Performance Analysis of Deep Neural Network to Noisy Digit Dataset

Karel Octavianus Bachri[1]

[1]Graduate School Electrical Engineering, School of Bioscience, Technology, and Innovation, Atma Jaya Catholic University of Indonesia, Jakarta 12930, INDONESIA

Article Info

Article history:

Received
26-05-2025

Accepted
04-06-2025

Keywords:
Active White Gaussian Noise
Deep Neural Network
Generalization Ability
Noisy digit dataset
System Robustness

Abstract

This work investigates the impact of noise on model performance by training a neural network on a digit dataset with varying Signal-to-Noise Ratios (SNR) to assess its resilience and generalization ability. The experimental setup involved training the model on datasets with noise levels ranging from clean images to highly distorted ones (SNR 5%–25%), analyzing accuracy, mini-batch loss, and training time. Results indicate that while the model achieves high accuracy (96.88%) at mild noise levels (SNR 5%), performance declines significantly at higher noise levels, with accuracy dropping to 78.91% at SNR 25%. The analysis of mini-batch loss and training time reveals that noise slows convergence and increases computational complexity. The confusion matrix further confirms that while the model effectively distinguishes between classes, noise-induced misclassifications become more frequent at lower SNRs. These findings emphasize the importance of noise reduction techniques and data preprocessing to improve model robustness in real-world applications.

Info Artikel

Histori Artikel:

Diterima:
26-05-2025

Disetujui:
04-06-2025

Kata Kunci:
Active White Gaussian Noise
Deep Neural Network
Generalization Ability
Noisy digit dataset
System Robustness

Abstrak

Makalah ini bertujuan menyelidiki bagaimana jaringan saraf yang dilatih pada dataset digit dengan berbagai tingkat noise dapat mengenali gambar, dengan tujuan mengevaluasi ketahanan dan kemampuan generalisasinya. Eksperimen dilakukan dengan melatih model pada dataset dengan Signal-to-Noise Ratio (SNR) yang bervariasi, mulai dari gambar normal hingga kondisi dengan berbagai Tingkat noise (SNR 5%–25%), kemudian menganalisis akurasi, mini-batch losses, dan waktu pelatihan. Hasil menunjukkan bahwa model mencapai akurasi tinggi (96,88%) pada SNR 5%, tetapi kinerjanya menurun seiring meningkatnya noise, dengan akurasi turun menjadi 78,91% pada SNR 25%. Analisis juga menunjukkan bahwa noise memperlambat konvergensi dan meningkatkan kompleksitas komputasi, yang terlihat dari waktu pelatihan yang lebih lama dan kehilangan yang lebih tinggi pada SNR rendah. Matriks kebingungan mengonfirmasi bahwa meskipun model dapat mengklasifikasikan sebagian besar sampel dengan baik, misklasifikasi lebih sering terjadi pada tingkat noise yang lebih tinggi. Temuan ini menekankan pentingnya teknik reduksi noise dan prapemrosesan data untuk meningkatkan ketahanan model dalam aplikasi dunia nyata.

1. INTRODUCTION

Humans can recognize images even when they are noisy, incomplete, or distorted. This skill is a result of the brain's advanced pattern recognition system, which allows us to identify objects, faces, and scenes even under challenging visual conditions. Unlike computers that rely heavily on precise input, the human brain can make sense of fragmented or obscured images by filling in missing details and filtering out

*Corresponding author. Karel Octavianus Bachri
Email address: karel.bachri@atmajaya.ac.id

irrelevant noise. This capability is essential for daily life, enabling us to interpret blurry photographs, read partially covered text, or recognize familiar objects in low-light environments [1].

Another study explored how training deep neural networks (DNNs) with noisy images can enhance their alignment with human visual processing. Researchers found that DNNs trained on noisy data not only improved in object recognition tasks under noisy conditions but also exhibited neural activation patterns more closely resembling those observed in human brains. This suggests that incorporating noise during training can make artificial vision systems more robust and human-like in their processing [2].

Additionally, a study investigated the spatiotemporal neural dynamics of object recognition under noisy and ambiguous conditions. Using advanced neuroimaging techniques, researchers observed that the human brain employs specific temporal patterns to process and recognize objects, even when visual information is degraded. These findings provide valuable insights into the temporal aspects of how our visual system maintains recognition performance amidst challenging conditions [3], [4].

In real-world scenarios, recognizing images from noisy or incomplete data is crucial for various applications, including medical imaging, autonomous driving, and security systems. In medical diagnostics, for example, doctors often analyze scans that may contain noise due to limitations in imaging technology or patient movement. Similarly, self-driving cars rely on cameras and sensors to navigate their environment, even in poor weather conditions where images may be blurry or occluded. Security and surveillance systems must also accurately identify individuals or objects despite low-resolution footage or obstructions. The ability to process and recognize images under such conditions is essential for ensuring safety, accuracy, and reliability in these critical fields [5], [6].

This research aims to investigate the performance of a network trained on a noisy dataset, evaluating how well it can recognize images despite distortions. By analyzing how deep learning models handle degraded visual information, researchers can develop more robust and adaptive artificial intelligence systems. The study will explore different noise levels and types to assess the model's resilience and generalization ability. Understanding the network's response to noise can help improve its architecture and training methods, ultimately leading to enhanced real-world applications where image recognition under imperfect conditions is necessary.

Research on incomplete data has been conducted in several areas. In database systems, the models, the type, the causes, and the solution regarding such a problem have been investigated and solved [7]. In the area of computation and Big Data, Bayesian inference has been used to perform real-world learning tasks, which involve large dimensions of data using statistical methods [8]. In the cyber security area, Artificial Intelligence has been developed for several scenarios [2], [9], [10].

2. LITERATURE REVIEW

2.1 Deep Neural Network

Deep Neural Networks (DNNs) have become a cornerstone in modern artificial intelligence (AI), enabling advancements in computer vision, natural language processing, and various other domains. The proliferation of DNNs has been driven by increased computational power, large datasets, and improved training algorithms. This literature review provides an overview of key research studies, methodologies, and applications of DNNs.

DNNs are a class of artificial neural networks (ANNs) that consist of multiple hidden layers between the input and output layers. Each layer is composed of neurons that perform weighted summation and activation functions to learn representations of data. The development of deep learning frameworks such as TensorFlow and PyTorch has facilitated DNN implementation.

Training DNNs involves optimizing weights using backpropagation and stochastic gradient descent (SGD). Various optimizers, such as Adam, RMSprop, and AdaGrad, have been proposed to improve convergence. Batch normalization, dropout, and data augmentation are common regularization techniques that mitigate overfitting. Dropout is also introduced as effective regularization method to reduce underfitting [11], while the training speed and stability is improved by applying batch normalization [12].

2.2 Additive White Gaussian Noise (AWGN)

Additive White Gaussian Noise (AWGN) is a fundamental concept in signal processing and communication systems. It models the random noise that affects transmitted signals in various channels, serving as a standard benchmark for evaluating the performance of communication systems. This literature review explores key studies on AWGN, its mathematical modelling, impact on system performance, and mitigation techniques [13].

AWGN is characterized by its additive nature, Gaussian amplitude distribution, and constant power spectral density over all frequencies. Mathematically, it is represented as $y(t) = x(t) + n(t)$, where $x(t)$

the transmitted signal, $n(t)$ is the noise component, and $y(t)$ is the received signal. The noise $n(t)$ follows a normal distribution $N(0, \sigma^2)$, where σ^2 represents the noise variance.

3. METHODOLOGY

This work is conducted in several steps, data collection, preprocessing, data splitting, network building, training, and testing. This section discusses how each step is carried out.

3.1 Data collection

Data is collected from MNIST Digit Dataset [14]. Each sample is a grayscale image of 28x28 and are organized by folders. Data comprises single digits as shown in Table **1**, while the examples are shown in Figure **1**.

Table 1. Dataset composition

No.	Digit	Folder	#Samples	Nomor Files
1	1	"1"	1000	1 – 1000
2	2	"2"	1000	1001 – 2000
3	3	"3"	1000	2001 – 3000
4	4	"4"	1000	3001 – 4000
5	5	"5"	1000	4001 – 5000
6	6	"6"	1000	5001 – 6000
7	7	"7"	1000	6001 – 7000
8	8	"8"	1000	7001 – 8000
9	9	"9"	1000	8001 – 9000
10	0	"0"	1000	9001 – 10000

Figure 1. Samples of dataset.

3.2 Data preprocessing

Data is preprocessed by adding Additive White Gaussian Noise (AWGN), which satisfies normal distribution, as shown in Figure **2**. X-axis indicates the variable distribution, while y-axis indicates the probability of each variable. The value x = 0 indicates the mean of the distribution.

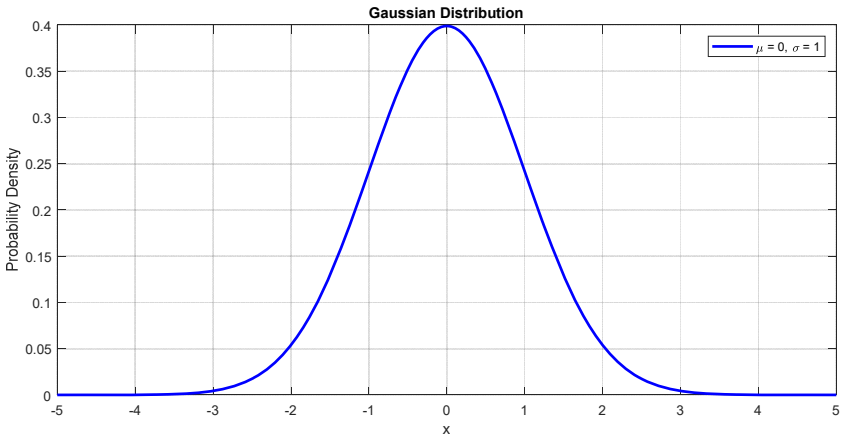

Figure 2. Gaussian distribution.

3.3 Data splitting

Data is split into train data and validation data with the proportion of 75% and 25% respectively. Given a total of 10,000 samples (10 categories × 1,000 samples each), the training set will contain 7,500 samples, and the test set will contain 2,500 samples. This split ensures that the model learns from enough data while reserving a meaningful portion for evaluation. The training set is used to teach model patterns and relationships within the data, while the test set serves as an unseen dataset to assess the model's performance, helping to identify potential overfitting or underfitting issues.

To maintain a balanced distribution of categories in both sets, stratified sampling is recommended. Stratified sampling ensures that each category maintains the same proportion in the training and test sets, preventing any imbalance that could bias the model. Since each category has 1,000 samples, a 75 – 25 split results in 750 training samples and 250 test samples per category.

By following this structured approach to data splitting, a robust foundation for model training and evaluation can be created. The test set remains independent, allowing for a reliable assessment of the model's generalization ability. This methodology supports fair comparisons between different models and hyperparameter tuning, ultimately leading to improved model performance and real-world applicability [15], [16].

3.4 Network building

Table 2. Network architecture

Layer	Name	Type	Size	Learning
1	imageinput	Image Input	28x28x1	-
2	conv_1	Convolution	28x28x8	Weights 3x3x1x8 Bias 1x1x8
3	batchnorm_1	Batch Normalization	28x28x8	Offset 1x1x8 Scale 1x1x8
4	relu_1	ReLU	28x28x8	-
5	maxpool_1	Max Pooling	14x14x8	-
6	conv_2	Convolution	14x14x16	Weights 3x3x8x16 Bias 1x1x16
7	batchnorm_2	Batch Normalization	14x14x16	Offset 1x1x16 Scale 1x1x16
8	relu_2	ReLU	14x14x16	-
9	maxpool_2	Max Pooling	7x7x16	-
10	conv_3	Convolution	7x7x32	Weights 3x3x16x32 Bias 1x1x32
11	batchnorm_3	Batch Normalization	7x7x32	Offset 1x1x32 Scale 1x1x32
12	relu_3	ReLU	7x7x32	-
13	fc	Fully Connected	1x1x10	Weights 10x1568 Bias 10x1
14	softmax	Softmax	1x1x10	-
15	classoutput	Classification Output	-	-

4. RESULT AND DISCUSSION

This section discusses the results and findings of this work. The results covers the modified dataset, network building result and

4.1 Adding noise to the dataset

Noise is added to the dataset as shown in Figure **1** and the result of this process is shown in Figure **3**.

a b c d e f g

Figure 3. Dataset with noise of various ratios added. A. normal image, b. 5% noise added, c. 10% noise added, d. 15% noise added, e. 20% noise added, f. 25% noise added, g. 30% noise added.

Figure **3**. a. shows a sample of the original data, adding noise with the ratio of 5%, 10%, 15%, 20%, 25%, and 30% produce a dataset as shown in Figure **3**. B., Figure **3**. C., Figure **3**. D., Figure **3**. E., Figure **3**. F., and Figure **3**. G. respectively.

4.2 Network building

The designed network is shown in Figure **4**. The network is divided into two main parts, feature-extraction and classification. The feature-extraction layers are basically convolutional layers. Features obtained from these layers are then processed in the classification layers, which is fully connected layers, activated by softmax, and classification layer.

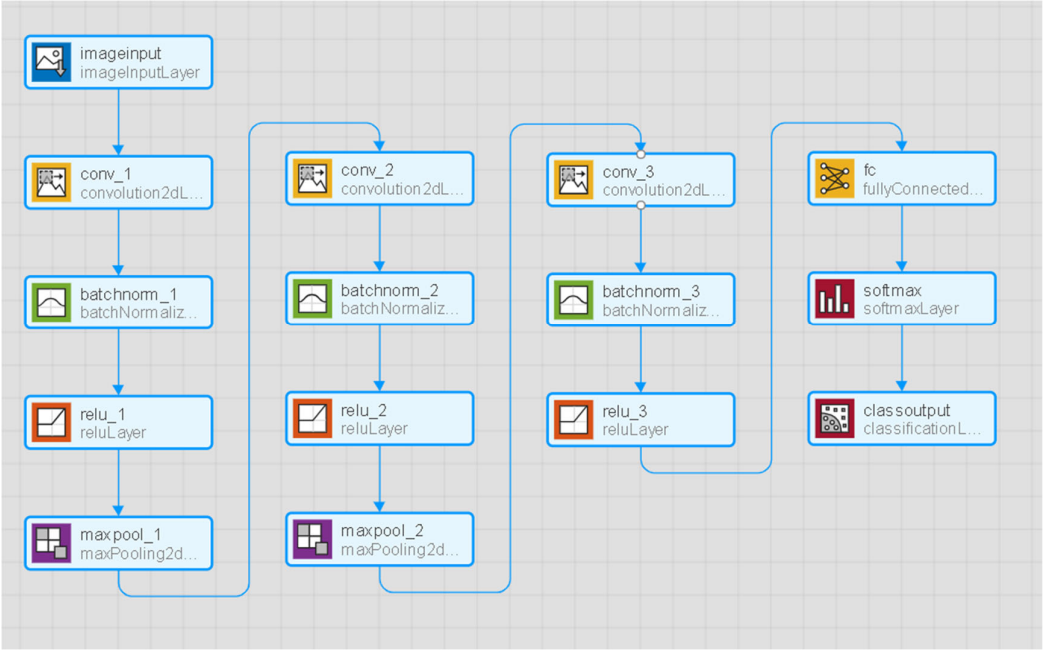

Figure 4. Network result

This deep network design follows a typical Convolutional Neural Network (CNN) architecture commonly employed for image classification tasks. The architecture exhibits a sequential arrangement of layers, starting with an input layer (`imageinput`) that receives the image. The core of the network comprises three convolutional blocks (`conv_1`, `conv_2`, `conv_3`), each consisting of a convolutional layer (`convolution2dL`) for feature extraction, batch normalization (`batchNormalization`) for improved training stability, a ReLU activation (`reluLayer`) for introducing non-linearity, and max pooling (`maxPooling2d`) for spatial dimension reduction.

Following the convolutional blocks, a fully connected layer (`fullyConnected`) integrates the learned features, and a softmax layer (`softmaxLayer`) generates probabilities for each class. Finally, the classification output layer (`classificationL`) provides the predicted class label. While the diagram provides a basic overview, several details remain unclear, including the specific kernel sizes, strides, padding, and number of filters in each layer. These parameters significantly influence the network's capacity and performance.

Further analysis could delve into regularization techniques, activation functions, loss functions, and optimizers used during training. The suitability of this architecture depends on factors such as dataset complexity, size, and available computational resources. Potential improvements could involve hyperparameter tuning, data augmentation, transfer learning, and exploration of more advanced architectures. Visualizing feature maps and conducting ablation studies can provide valuable insights into the network's learning process and the impact of individual components. Overall, this CNN architecture represents a solid foundation for image classification, with opportunities for optimization and refinement based on specific task requirements and data characteristics.

4.3 Network training

Training progress of normal dataset is shown in Table **3**.

Table 3. training results using normal datasets.

Epoch	Iteration	Time Elapsed (s)	Mini-batch Accuracy (%)	Mini-batch Loss
1	1	2	7.03	2.8750
1	50	11	88.28	0.4952
2	100	20	94.53	0.2032
3	150	29	99.22	0.0726
4	200	37	99.22	0.0697
4	232	43	100.00	0.0563

Table **3** shows the training progress of a deep neural network, showcasing a clear trend of improvement across epochs. Initially, at epoch 1, iteration 1, the model exhibits low accuracy (7.03%) and high loss (2.8750), indicating a nascent stage of learning. However, as training progresses, we observe a significant increase in accuracy and a corresponding decrease in loss. By epoch 4, iteration 232, the model achieves perfect accuracy (100.00%) with a minimal loss (0.0563), suggesting successful convergence and effective learning from the clean image inputs. This demonstrates the network's ability to extract relevant features and patterns from the data, leading to a highly accurate predictive model.

Table 4. training result using normal dataset 5%.

Epoch	Iteration	Time Elapsed (s)	Mini-batch Accuracy (%)	Mini-batch Loss
1	1	3	7.81	2.6049
1	50	13	75.78	0.7522
2	100	23	85.16	0.4463
3	150	32	92.19	0.2958
3	200	41	93.75	0.2382
4	250	47	94.53	0.1849
4	296	53	96.88	0.1605

Table **4** shows the training process of a deep neural network on noisy image inputs with a Signal-to-Noise Ratio (SNR) of 0.05. The initial epoch shows a low accuracy of 7.81% and a high loss of 2.6049, indicating the challenge posed by the noisy data. However, as the training progresses through iterations and epochs, a consistent improvement is observed. The accuracy steadily increases, reaching 96.88% by epoch 4, iteration 296, while the loss correspondingly decreases to 0.1605. This demonstrates the network's ability to learn and extract meaningful features even from slightly noisy images, highlighting its robustness to noise and the effectiveness of the training process in mitigating the impact of low SNR.

Table 5. training results using a noisy dataset with SNR 10%.

Epoch	Iteration	Time Elapsed (s)	Mini-batch Accuracy (%)	Mini-batch Loss
1	1	1	3.91	2.8793
1	50	8	57.81	1.1273
2	100	14	82.81	0.5344
3	150	21	82.03	0.4360
3	200	30	89.84	0.4168
4	250	38	92.97	0.2902
4	296	46	92.97	0.2524

Table **5** shows the training results of a machine learning model on a noisy dataset with an SNR of 10%, showing the progression of mini-batch accuracy, loss, and time elapsed across iterations and epochs. Initially, the model starts with a low accuracy of 3.91% and a high loss of 2.8793, but it quickly improves, reaching 57.81% accuracy by iteration 50 and 82.81% by iteration 100. As training progresses, the accuracy continues to rise, stabilizing around 92.97% by iteration 296, while the loss steadily decreases to 0.2524, indicating effective learning. The elapsed time increases linearly, reaching 46 seconds at the final recorded iteration, suggesting efficient computation. The model exhibits rapid learning in the early stages, followed by gradual improvement, with diminishing returns beyond Epoch 3, implying potential convergence.

Table 6. training results using a noisy dataset with SNR 15%.

Epoch	Iteration	Time Elapsed (s)	Mini-batch Accuracy (%)	Mini-batch Loss
1	1	1	11.72	2.7565
1	50	9	54.69	1.3975
2	100	18	70.31	0.7903
3	150	25	85.16	0.5392
3	200	33	88.28	0.3715
4	250	42	90.63	0.3224
4	296	49	91.47	0.2839

Table **6** shows the training results of a machine learning model on a noisy dataset with an SNR of 15%, tracking the mini-batch accuracy, loss, and time elapsed over multiple iterations and epochs. Initially, the model starts with an accuracy of 11.72% and a high loss of 2.7565 at iteration 1, but it rapidly improves to 54.69% accuracy and a reduced loss of 1.3975 by iteration 50. As training progresses, accuracy continues to rise, reaching 70.31% at iteration 100 and 85.16% at iteration 150, while the loss steadily decreases. By iteration 296 (Epoch 4), accuracy peaks at 91.47% with a final mini-batch loss of 0.2839, indicating strong learning performance. The increasing time elapsed follows a stable trend, reaching 49 seconds at the last iteration. Compared to the SNR 10% scenario, the model achieves higher initial accuracy and slightly faster convergence, suggesting that a cleaner dataset (higher SNR) enhances learning efficiency and performance.

Table 7. Training results using a noisy dataset with SNR 20%.

Epoch	Iteration	Time Elapsed (s)	Mini-batch Accuracy (%)	Mini-batch Loss
1	1	1	11.72	2.8399
1	50	8	46.09	1.5321
2	100	16	75.78	0.8322
3	150	26	81.25	0.6613
3	200	34	82.03	0.6252
4	250	42	85.16	0.4701
4	296	50	85.16	0.4198

Table **7** shows the training results of a machine learning model on a noisy dataset with an SNR of 20%, tracking mini-batch accuracy, loss, and elapsed time across iterations and epochs. The model starts with an accuracy of 11.72% and a high loss of 2.8399 at iteration 1 but quickly improves, reaching 46.09% accuracy and a reduced loss of 1.5321 by iteration 50. As training progresses, accuracy continues to increase, reaching 75.78% at iteration 100 and 81.25% at iteration 150, with a corresponding decline in loss. By iteration 296 (Epoch 4), accuracy stabilizes at 85.16%, while the loss drops to 0.4198, indicating learning convergence. Compared to SNR 10% and 15%, the model initially learns more slowly but achieves a stable performance around Epoch 4. The slightly lower final accuracy suggests that while a cleaner dataset (higher SNR) aids learning, performance gains diminish beyond a certain noise threshold, indicating that other factors may limit further improvement.

Table 8. Training results using a noisy dataset with SNR 25%.

Epoch	Iteration	Time Elapsed (s)	Mini-batch Accuracy (%)	Mini-batch Loss
1	1	0	7.03	2.8124
1	50	6	50.00	1.4524
2	100	14	71.09	0.8211
3	150	22	64.84	0.9639
3	200	31	73.44	0.7780
4	250	40	76.56	0.6469
4	296	47	78.91	0.5817

Table **8** shows the training results of a machine learning model on a noisy dataset with an SNR of 25%, tracking mini-batch accuracy, loss, and elapsed time over iterations and epochs. The model starts with a low accuracy of 7.03% and a high loss of 2.8124 at iteration 1 but quickly improves to 50.00% accuracy and a reduced loss of 1.4524 by iteration 50. As training progresses, accuracy increases to 71.09% at iteration 100 and fluctuates slightly before reaching 78.91% at iteration 296, with loss steadily decreasing to 0.5817. Compared to lower SNR datasets, the model's final accuracy is lower, suggesting

that higher noise levels hinder learning efficiency. While the model still improves over time, the slower convergence and lower final accuracy indicate that excessive noise in the dataset negatively impacts performance despite continued training.

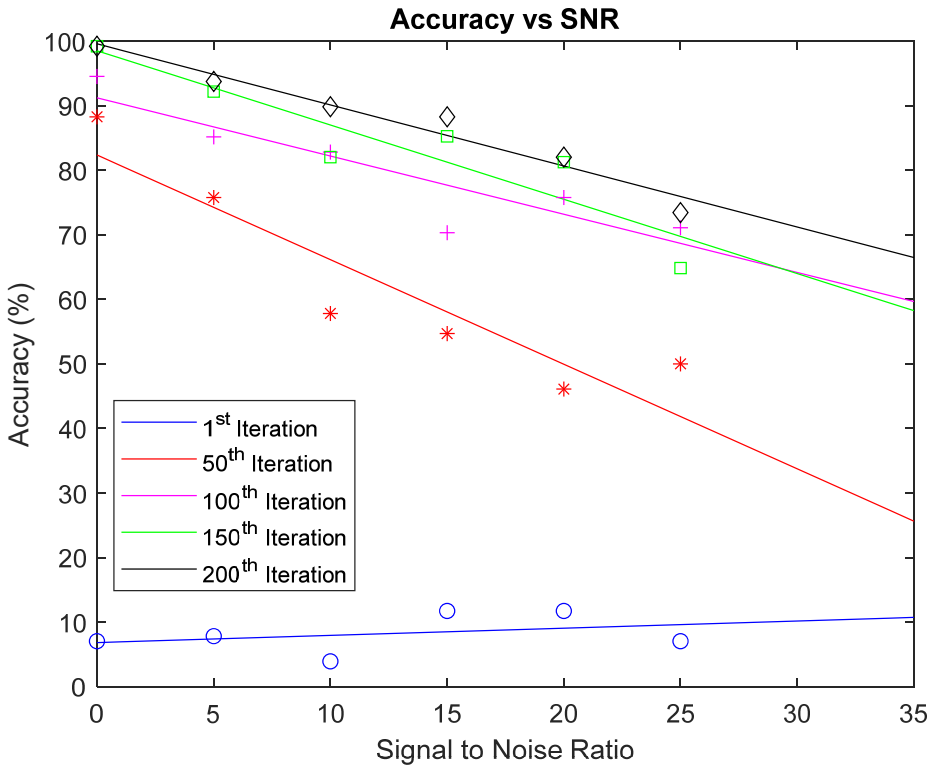

Figure 5. Accuracy vs SNR for every iteration

Figure **5** illustrates the relationship between the Signal-to-Noise Ratio (SNR) and model accuracy at different training iterations (1st, 50th, 100th, 150th, and 200th). As SNR increases, accuracy consistently improves across all iterations, indicating that lower noise levels facilitate better learning. The trend lines for later iterations (100th, 150th, and 200th) show higher accuracy compared to earlier ones, demonstrating that longer training helps the model achieve better performance. However, the accuracy drop at lower SNRs is more pronounced, especially in the 50th iteration, where performance declines sharply, suggesting that noisy data significantly hinders early learning. The 1st iteration remains consistently low across all SNR values, emphasizing that the model starts with minimal knowledge. The overall trend suggests that while training can mitigate the impact of noise to some extent, excessive noise still negatively affects final accuracy, highlighting the importance of dataset quality in model performance.

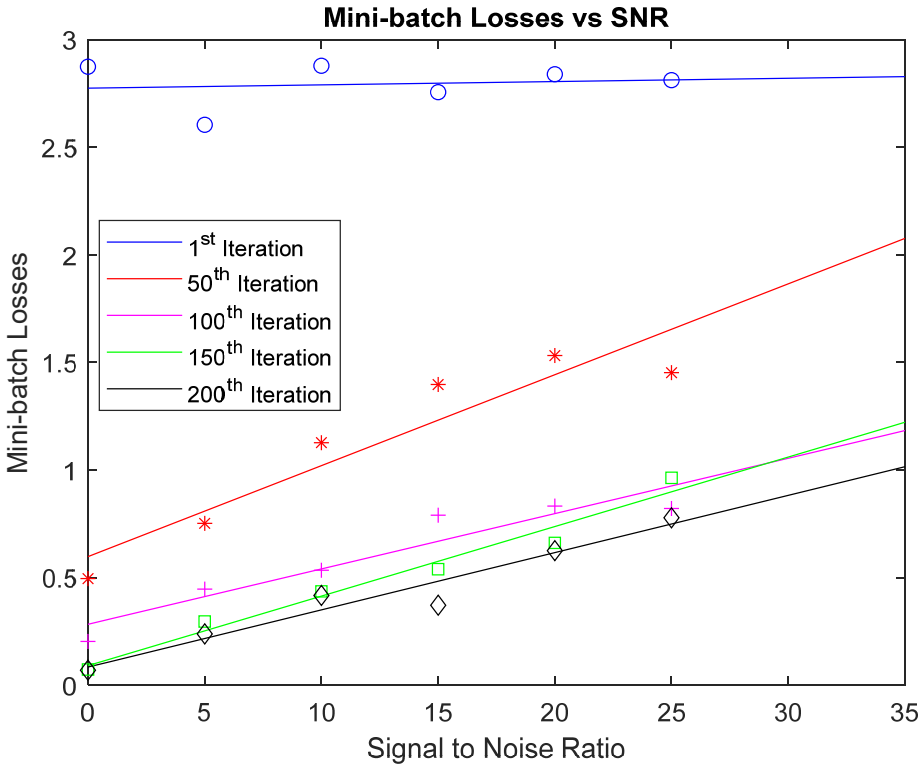

Figure 6. Mini-batch Losses vs SNR for every iteration.

Figure **6** illustrates the relationship between Signal-to-Noise Ratio (SNR) and mini-batch loss across different training iterations (1st, 50th, 100th, 150th, and 200th). As SNR increases, mini-batch loss generally decreases, indicating improved learning performance with lower noise levels. The 1st iteration maintains the highest loss across all SNR values, reflecting the model's initial lack of training. The 50th iteration shows a significant decrease in loss as SNR increases, but it remains relatively high compared to later iterations, suggesting that early training is more affected by noise. The 100th, 150th, and 200th iterations exhibit consistently lower losses, with the 200th iteration achieving the lowest values, indicating steady convergence. The overall trend highlights that while training can reduce loss over time, datasets with lower SNR values (higher noise) still result in higher losses, emphasizing the challenge of learning in noisy environments.

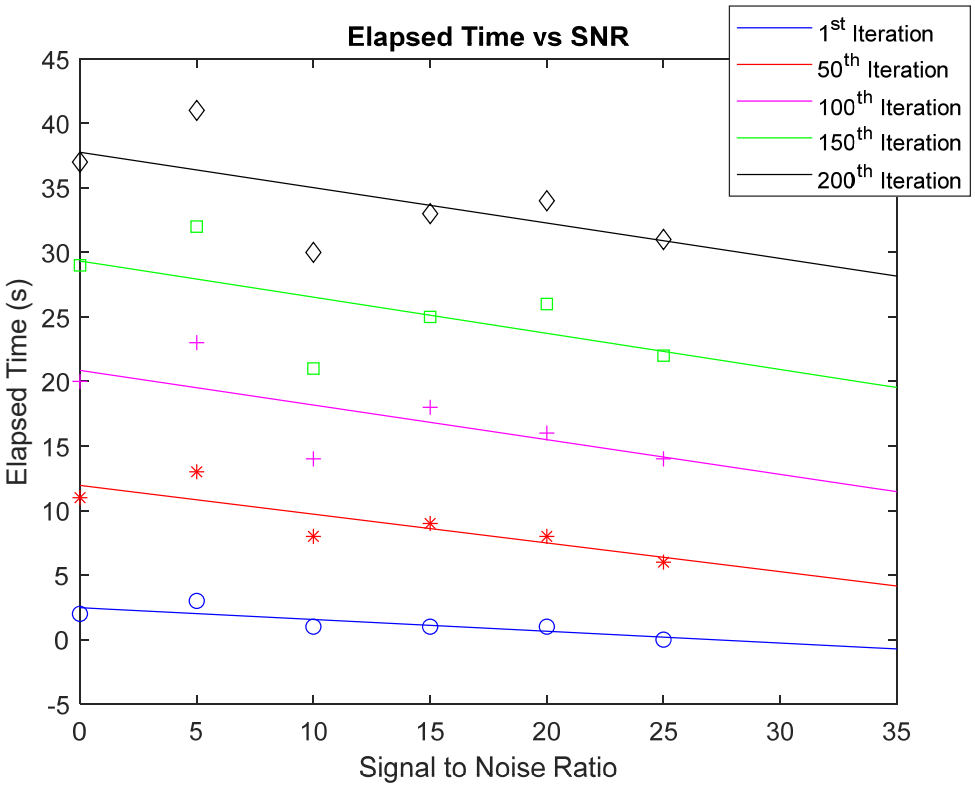

Figure 7. Elapsed Times vs SNR

Figure **7** illustrates the relationship between Signal-to-Noise Ratio (SNR) and elapsed training time across different training iterations (1st, 50th, 100th, 150th, and 200th). The general trend shows that as SNR increases, the elapsed time for each iteration decreases, indicating that training is more computationally demanding when the dataset contains more noise (lower SNR). The 1st iteration consistently exhibits the lowest elapsed time, as minimal computation occurs in the initial stage. However, as training progresses, the elapsed time increases, with the 200th iteration taking the longest. This suggests that as the model undergoes more updates, the computational burden grows due to more complex weight adjustments and gradient calculations. The trend also indicates that noisy data requires additional processing time to refine model parameters, making training more time-consuming in low-SNR environments.

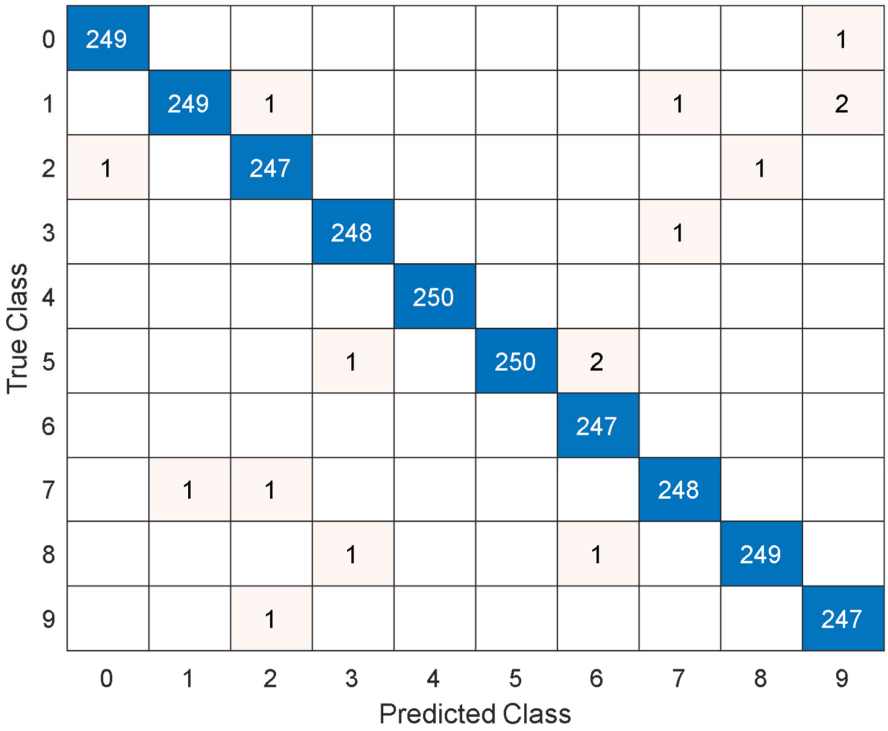

Figure 8. Confusion matrix of the normal dataset

The confusion matrix illustrates the classification performance of a model across ten classes. The diagonal elements represent correctly classified instances, with most values being close to 250, indicating high accuracy. Misclassifications are sparse, appearing as off-diagonal elements, suggesting that the model performs well but has minor errors in some classes. The most significant misclassifications occur in a few classes where predictions deviate slightly from the true class, though the overall distribution remains strong. This suggests that the model has learned meaningful features for distinguishing between classes, but there is still room for improvement in reducing misclassifications.

5. CONCLUSION AND FUTURE WORKS

The training results across different datasets, varying in noise levels (SNR), provide key insights into the deep neural network's learning behavior. When trained on the normal dataset, the model achieved rapid and high accuracy, reaching 100% with a minimal loss, demonstrating its ability to effectively learn patterns from clean images. However, as noise levels increased (lower SNR), the model's performance gradually declined, requiring more iterations to achieve convergence. At SNR 5%, the model still reached high accuracy (96.88%), indicating strong resilience to mild noise. At SNR 10% and 15%, the accuracy plateaued slightly lower (92.97% and 91.47%, respectively), suggesting that noise impacted the model's ability to extract meaningful features, as shown in Table **3** to Table **6**.

When trained on datasets with even higher noise levels (SNR 20% and 25%), the model faced more difficulties in achieving high accuracy. At SNR 25%, accuracy stagnated at 78.91%, showing that excessive noise significantly hinders learning. While loss still decreased over iterations, the diminishing accuracy gains suggest that beyond a certain noise threshold, the model struggles to distinguish relevant features from noise, leading to suboptimal performance, as shown in Table **7** and Table **8**.

The experimental results demonstrate the impact of Signal-to-Noise Ratio (SNR) on the performance of a deep neural network. As observed in Figure **5** to Figure **7**, higher SNR values lead to improved accuracy, lower mini-batch losses, and reduced training times. This confirms that cleaner datasets facilitate more efficient learning, whereas higher noise levels hinder early training and slow down convergence. The confusion matrix Figure **8** further supports these findings, showing high classification accuracy with only minor misclassifications, suggesting that the model has effectively learned distinguishing features across the ten classes.

However, the results also highlight that excessive noise negatively impacts both accuracy and training efficiency. While the model demonstrates resilience to moderate noise, performance gains diminish at higher noise levels (lower SNR), where misclassifications become more frequent. This emphasizes the importance of high-quality data and noise reduction techniques to enhance model performance. Overall, the study reinforces that while deep learning models can adapt to noisy environments, optimal results are achieved when trained on cleaner datasets with minimal noise interference.

Overall, the results indicate that while the neural network is robust to a certain degree of noise, its learning efficiency and final accuracy degrade as noise increases. This highlights the importance of data preprocessing techniques such as noise reduction and augmentation to improve model generalization in noisy environments.

Building upon the findings of this study, there are several promising directions for future research to further investigate and enhance the performance of deep neural networks when dealing with noisy image data. One important avenue is the application of noise prefiltering techniques. Before feeding the noisy images into the neural network, various denoising filters—such as Gaussian blur, median filters, or wavelet-based methods—could be applied. These preprocessing steps may reduce the impact of noise, particularly in datasets with lower signal-to-noise ratios (SNR), and improve classification accuracy. Comparing filtered versus unfiltered inputs across multiple SNR levels would help quantify the benefit of such approaches.

Another key area for exploration is the study of different types of noise. While the current work likely focuses on a single or limited set of noise models (e.g., Gaussian noise), real-world scenarios involve a diverse range of noise types including salt-and-pepper noise, speckle noise, Poisson noise, and sensor-specific artifacts. By evaluating the model's performance across these varied conditions, researchers can better understand its robustness and identify weaknesses that could be addressed through architectural or training improvements.

In addition to these extensions, a valuable direction would be to develop adaptive or noise-aware neural network models. These models could include mechanisms such as attention layers that dynamically adjust to the noise level in the input, or customized loss functions that penalize misclassifications more severely in noisier examples. Such approaches could make the model more resilient and better suited for deployment in unpredictable or low-quality imaging environments.

Furthermore, it would be beneficial to evaluate the findings across different datasets. Applying the same methodology to datasets like EMNIST, Fashion-MNIST, or CIFAR-10 would test the generalizability of the observed performance trends. This cross-dataset analysis would offer a broader understanding of how noise affects model performance in tasks of varying complexity.

Finally, future work could also involve simulating real-time or streaming data conditions, where noise levels might vary dynamically. Testing model robustness in such scenarios would provide insights into its practicality for real-world applications, such as OCR systems or mobile imaging, where noise can be unpredictable and transient.

REFERENCE

[1] A. Nayan, J. Saha, K. R. Mahmud, A. K. Al Azad, and M. G. Kibria, "Detection of Objects from Noisy Images," in *2nd International Conference on Sustainable Technologies for Industry 4.0 (STI)*, Dhaka: IEEE, 2020. doi: 10.1109/STI50764.2020.9350521.

[2] H. Jang, D. McCormack, and F. Tong, "Noise-trained deep neural networks effectively predict human vision and its neural responses to challenging images," *PLoS Biol*, vol. 19, no. 12, Dec. 2021, doi: 10.1371/journal.pbio.3001418.

[3] Y. H. Wu, E. Podvalny, and B. J. He, "Spatiotemporal neural dynamics of object recognition under uncertainty in humans," *Elife*, vol. 12, May 2023, doi: 10.7554/eLife.84797.

[4] K. Guo, P. Wang, P. Shi, C. He, and C. Wei, "A New Partitioned Spatial–Temporal Graph Attention Convolution Network for Human Motion Recognition," *Applied Sciences (Switzerland)*, vol. 13, no. 3, Feb. 2023, doi: 10.3390/app13031647.

[5] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang, "Learning from Massive Noisy Labeled Data for Image Classification," in *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, [EEE], 2015.

[6] C. Liu *et al.*, "A Comprehensive Study on Robustness of Image Classification Models: Benchmarking and Rethinking," 2023. [Online]. Available: https://github.com/thu-ml/ares

[7] E. Wong, "INCOMPLETE INFORMATION I N DATABASE SYSTEMS," 1980.

[8] T. Emmanuel, T. Maupong, D. Mpoeleng, T. Semong, B. Mphago, and O. Tabona, "A survey on missing data in machine learning," *J Big Data*, vol. 8, no. 1, Dec. 2021, doi: 10.1186/s40537-021-00516-9.

[9] P. Alexandre, "Defending Networks with Incomplete Information: A Machine Learning Approach," 2013.

[10] B. Alomar, Z. Trabelsi, T. Qayyum, and M. M. Ambali Parambil, "AI and Network Security Curricula: Minding the Gap," in *2024 IEEE Global Engineering Education Conference (EDUCON)*, 2024, pp. 1–7. doi: 10.1109/EDUCON60312.2024.10578588.

[11] Z. Liu, Z. Xu, J. Jin, Z. Shen, and T. Darrell, "Dropout reduces underfitting," in *Proceedings of the 40th International Conference on Machine Learning*, in ICML'23. JMLR.org, 2023.

[12] I. Marin, A. K. Skelin, and T. Grujic, "Empirical evaluation of the effect of optimization and regularization techniques on the generalization performance of deep convolutional neural network," *Applied Sciences (Switzerland)*, vol. 10, no. 21, pp. 1–30, Nov. 2020, doi: 10.3390/app10217817.

[13] H. Al-Ghaib and R. Adhami, "On the Digital Image Additive White Gaussian Noise Estimation," in *International Conference on Industrial Automation, Information and Communications Technology (IAICT 2014)*, IEEE, Aug. 2014, p. 144.

[14] L. Deng, "The MNIST database of handwritten digit images for machine learning research," *IEEE Signal Process Mag*, vol. 29, no. 6, pp. 141–142, 2012, doi: 10.1109/MSP.2012.2211477.

[15] J. Sadaiyandi, P. Arumugam, A. K. Sangaiah, and C. Zhang, "Stratified Sampling-Based Deep Learning Approach to Increase Prediction Accuracy of Unbalanced Dataset," *Electronics (Switzerland)*, vol. 12, no. 21, Nov. 2023, doi: 10.3390/electronics12214423.

[16] M. Sivakumar, S. Parthasarathy, and T. Padmapriya, "Trade-off between training and testing in machine learning for medical image processing," 2024, doi: 10.7717/peerj-cs.2245.