

Boosting Feature Learning in Autoencoders with Convolutional Layers

Muhammad Usama Yousaf^{1*}, Rabia Khan², Aiman Sohaib³, Ayesha Saddiqua⁴, Mahnoor Izzat⁵, Sana Hassan⁶, Muhammad Adnan Kaim Khani⁷

Abstract: This research proposes a Convolutional Autoencoder (CAE) as an alternative to traditional Autoencoders (AEs) for improving both image reconstruction quality and feature representation. While conventional AEs are effective for dimensionality reduction and unsupervised feature learning, they often struggle to accurately reconstruct compressed images, which leads to a loss of spatial information and, in turn, weaker feature representations and reduced classification performance on downstream tasks. To address this limitation, the proposed CAE integrates convolutional layers into its encoder and decoder, enabling the model to capture local spatial hierarchies and translational invariance that are critical for image data. Supplementary compression layers are further introduced to refine the latent representation, allowing for more efficient encoding with minimal information loss. Together, these design choices improve the model's ability to reconstruct images with higher fidelity while extracting more discriminative features. The proposed CAE is evaluated against three established models, D-CAE, ScatNet, and PCANet, on the MNIST dataset, where it achieves 96% accuracy, outperforming ScatNet (93%), PCANet (92%), and D-CAE (92%). It attains this result with a comparatively lightweight architecture, suggesting a favorable balance between accuracy and computational cost; although a detailed runtime, comparison is left for future work. These findings indicate that convolutional enhancements can make autoencoders more effective for image reconstruction and classification tasks that require both accuracy and efficiency; since the evaluation is limited to the MNIST benchmark, further testing on more complex, real-world datasets would be needed to confirm how broadly these benefits generalize.

Keywords: Convolutional Neural Network, Autoencoder, ScatNet, and PCANet

I. INTRODUCTION

An autoencoder is a form of artificial neural network that learns how to efficiently encode data without labels (unsupervised learning). An autoencoder performs two functions: encoding, which alters the input data, and decoding, which reconstructs the input data from its encoded representation. Autoencoders (AEs) are trained via backpropagation to convert a high-dimensional set of input features into a low-dimensional encoded representation, and then to reconstruct the original feature set from that representation. Encoding is the reduction of high-dimensional data to low-dimensional data, whereas decoding is a reconstruction of the original data from low-dimensional data [1]. In deep learning, the autoencoder is capable of extracting target features, successfully addressing the issue of inadequate feature extraction by traditional manual approaches while avoiding overfitting. Autoencoders also have certain drawbacks, including long training times for deep models trained layer by layer, limited interpretability of the learned features, and, in some cases, insufficient accuracy.

^{1,5,7} Faculty of Information and Communication Technology, Universiti Tunku Abdul Rahman

⁶ Faculty of Business and Finance, Universiti Tunku Abdul Rahman

^{2,3} Department of Management Science, The University of Haripur

⁴ Institute of Computing, Kohat University of Science and Technology
Country: Pakistan, Malaysia

Email: *its.usamayousaf@gmail.com

To address these issues, researchers have carried out extensive studies and proposed a range of improved autoencoder models. Convolutional neural networks, such as AlexNet, VGG, and ResNet, serve as the foundation for most deep learning models. Researchers employ deep convolutional neural network models to tackle real-world challenges [2]-[5]. Neural network-based techniques include convolutional neural networks and autoencoders. Autoencoders are designed to train representation functions for data sets and produce models based on the learned data. Data noise reduction is another advantage of AEs. AEs may be taught to remove noise and unnecessary details from input data, hence improving data quality. AEs promote a greater comprehension of data by creating meaningful representations [6]. However, conventional autoencoders typically rely on fully connected layers, which flatten the input and disregard the spatial structure of image data; as a result, they are less able to capture local patterns such as edges, textures, and shapes, and they require substantially more parameters to represent an image than an approach that exploits spatial locality. Convolutional layers address this limitation directly: by sharing filter weights across the spatial extent of an image and preserving local neighborhood relationships, they allow an autoencoder to learn spatially-aware, translation-invariant features with far fewer parameters than an equivalent fully connected design. This motivates the Convolutional Autoencoder (CAE) proposed in this work, which combines this spatial feature extraction with a classification branch so that reconstruction and digit recognition are learned jointly rather than as two separate stages.

The remainder of this paper is organized as follows. Section II reviews related work on autoencoder-based feature learning. Section III describes the proposed methodology, including data preparation, model architecture, and training procedure. Section IV presents the experimental results and compares the proposed CAE against established baseline models. Section V concludes the paper and outlines directions for future work.

II. RELATED WORK

In the literature, many authors have applied an autoencoder for different types of image classification, e.g., Federico Pichi et al [7]-[10] present a novel structure for nonlinear model order reduction based on a Graph Convolutional Autoencoder (GCA-ROM). Recent advances in reduced order modeling (ROM) have moved beyond standard linear techniques like Proper Orthogonal Decomposition (POD) and into nonlinear methods like the GCA-ROM. This system uses autoencoders and graph neural networks (GNNs) to simulate parametric PDEs on unstructured meshes in an efficient and generalizable manner. The GCA-ROM performs and scales well in complicated and data-limited settings across several models.

Jia K et al [11]-[13] used a regularized function framework in feature learning to improve the locality-preserving of input observations on the manifold during encoding and dubbed it LAE. They employed the benchmark datasets MNIST and CIFAR to recognize handwritten numbers and objects,

respectively. MNIST utilized 50,000 photos for training, 10,000 for validation, and 10,000 for testing purposes. CIFAR10 utilized 50,000 photos to train an object identification model and 10,000 to test the model. The model received 3468 input nodes and produced ten output classes.

Michel Barlaud et al [14]-[16] propose a supervised generative non-parametric autoencoder to overcome the constraints of traditional variational autoencoders (VAEs). Unlike VAEs, which require a parametric latent area distribution, their technique trains a non-parametric distribution and shapes its latent space using labels. This approach, which utilizes ℓ_1 regularization for better sparsity and clustering, is successful on the MNIST and biological datasets.

Janjua et al [17] implemented three deep learning denoising autoencoder image retrieval models trained and tested on Fashion-MNIST and MNIST datasets. When the Label Ranking Average Precision (LRAP) values of the three suggested models were compared, it got clear that the third model outscored the others, proving its superior picture retrieval capabilities. These models were additionally evaluated and trained on the MNIST dataset, which has greater inter-class separability. Surprisingly, the second and third models outperformed comparably to models examined on the Fashion-MNIST dataset when tested on the MNIST dataset, with higher LRAP values.

Muhammad Zulqarnain et al. [18] enhanced and suggested a gated recurrent text classification unit based on an auto-encoder. This study produced three new findings. Identifying important learning features and effectively reducing the dimensionality of those characteristics is the main goal of our proposed research using ES-GRU. Second, we removed the reset gate in the candidate state with an update gate and dropped the reset gate from the standard GRU. Thirdly, the final Softmax output layer for sentence categorization uses this research's Rectified Linear Unit (ReLU) activation in place of the conventional hyperbolic tangent activation function (tanh), and five benchmark datasets (20NG, IMDB, AG news, Yahoo, and Yelp), the suggested model outperformed.

Aamir M et al [19] presented the three most widely used versions of the traditional AE model: SAE, DAE, and CAE. The AE is a three-layered neural network model that is notable for having one input layer with n units coupled to m units in the hidden layer, where $m < n < 0$. After that, the concealed layer is joined to the last layer, which has n_c units in total, such that $n = n_c$. Because each of the three AE versions has unique qualities, researchers are drawn to using them to solve issues in the field of contemporary machine learning. To restrict the activation value at the hidden-layer neuron average, a sparse penalty component was added to the SAE cost function. Developing the capacity to reconstruct noisy data from the original data was the main goal of DAE. Training data with tiny changes and noise could not affect the feature representations learned by the CAE-based training model. Based on the MNIST dataset, the classification results of the comparative AE variations demonstrated that all of the AE models operated satisfactorily. The execution times of SAE were notably faster than those of DAE and CAE. In terms of training and testing errors, CAE fared better than SAE and DAE, according to the overall experimental data.

Fazli Wahid et al [20] proposed an improved convolutional neural network using L1 regularization, was used to automatically segment coronary arteries. Incorporating L1 regularization with varied coefficient values improves the performance of the CNN architecture. The suggested approach uses Gaussian Matched Filters (GMF) to improve the quality of X-Ray angiograms. Segmentation was done using eCNN. Dual L1 regularization

(coefficient: 0.001 for convolutional and 0.01 for dense layers) enhances model interpretability and generalization, allowing for accurate artery segmentation. The suggested model's performance was evaluated using standard parameters and compared to current methodologies.

Taken together, this body of work shows that autoencoder-based feature learning has been applied successfully across a wide range of data types and objectives, from graph-structured simulations [5] and manifold-preserving encodings [4] to text classification [8] and medical image segmentation [10]. However, most of the autoencoder variants discussed above, including the SAE, DAE, and CAE variants compared in [9], are evaluated primarily on reconstruction quality or use relatively shallow, non-convolutional architectures when applied to image data, and comparatively few studies jointly optimize a single convolutional architecture for both reconstruction and classification. The present work addresses this gap by proposing a hybrid CAE that shares a convolutional encoder between an image-reconstruction branch and a digit-classification branch, and by benchmarking it directly against ScatNet, PCANet, and D-CAE on a common dataset.

Beyond the studies discussed above, several more recent works speak more directly to convolutional autoencoders, hybrid reconstruction-classification designs, and representation learning. Chen and Guo review the development of autoencoder variants, including convolutional and adversarial architectures, and discuss their expanding role in representation learning across computer-vision tasks [21]. Dong and Kotenko combine a convolutional variational autoencoder with a weighted reconstruction loss and a classification objective for network-intrusion data, showing that coupling reconstruction and classification in a single model improves performance on imbalanced datasets [22]. Laakom et al. jointly train paired convolutional encoder-decoder branches to reconstruct multimodal inputs while keeping their shared latent representation compact, and use that representation for downstream one-class classification [23]. Together with the CAE variants reviewed above, these studies support the growing use of convolutional, jointly trained reconstruction-classification architectures for representation learning, which is the direction pursued by the hybrid CAE proposed in this work.

III. PROPOSED METHODOLOGY

A. Data Acquisition

This study uses the MNIST dataset, a widely used benchmark for image classification tasks. It contains 60,000 training and 10,000 testing images of handwritten digits ranging from 0 to 9, with each image measuring 28×28 pixels. The dataset is loaded using TensorFlow's Keras module, which converts the training and testing sets into a format suitable for model development and evaluation. MNIST is widely used in the autoencoder literature reviewed in Section II [13], [9], which makes it a suitable choice for benchmarking the proposed CAE directly against established models such as ScatNet, PCANet, and D-CAE under comparable conditions. Each image is grayscale, with pixel intensities ranging from 0 to 255, and the dataset is reasonably class-balanced across the ten digit categories, which limits the risk of the model being biased toward any single class during training.

B. Data Preprocessing

Preprocessing is critical in preparing the data for the neural network model. First, pixel values are normalized to the range [0, 1] by dividing by 255 and converting to the float32 data type; this normalization speeds up the model's convergence. The images are then reshaped to add a channel dimension, transforming them

from (28, 28) to (28, 28, 1), as required for compatibility with the network's Conv2D layers. To enable multi-class classification, the class labels are one-hot encoded using `to_categorical()`. In addition, Gaussian noise is injected into the dataset to generate noisy copies of the images for autoencoder training. This step improves the model's ability to learn robust features by training it to denoise images. The noisy images are used only as the autoencoder's input during training, while the corresponding clean images remain the reconstruction target, so the network learns to map a corrupted view of a digit back to its noise-free form. The same underlying images are later partitioned into the training, validation, and test subsets described in Section III.D, and this partitioning is applied consistently so that no image contributes to more than one subset. Together, these preprocessing steps ensure that the network receives normalized, appropriately shaped, and suitably augmented input, which supports stable gradient-based training and helps the autoencoder generalize beyond the exact pixel patterns seen during training.

C. Methodology Design

The model architecture consists of both an autoencoder and a classifier, which share a common encoder. This hybrid design takes advantage of both unsupervised learning (via the autoencoder) and supervised learning (via the classifier). The model's input layer is configured to accept images of shape (28, 28, 1). The encoder consists of two convolutional layers, each followed by a max-pooling layer that reduces spatial dimensions while extracting feature representations. The decoder mirrors the encoder's structure and uses Conv2D and UpSampling2D layers to progressively restore the image's resolution from the encoded representation. The classification branch takes the encoded representation, flattens it, and passes it through dense layers, ending in a softmax layer that outputs the predicted digit class. This hybrid model is then built, using the image as input and producing both the reconstructed image and the class prediction as outputs. Sharing a single encoder between the reconstruction and classification branches means that the features learned to reconstruct an image are the same features used to classify it; this encourages the encoder to retain information that genuinely represents the digit's structure, rather than encoding only whatever narrow set of features happens to minimize classification error in isolation. Compared to training a separate autoencoder and classifier independently, this joint design is also more parameter-efficient, since the convolutional feature-extraction layers are learned once and reused for both tasks.

D. Model Compilation

The hybrid model is compiled using the Adam optimizer, which is well-suited for training deep neural networks. The model uses two loss functions: binary cross-entropy for the autoencoder's image-reconstruction task, and categorical cross-entropy for the classification task. This dual-loss approach allows the model to simultaneously reduce reconstruction error and improve classification accuracy. Unless stated otherwise, the two loss terms are combined with equal weighting into a single scalar objective that is minimized during training, so neither task is prioritized over the other by default. Adam was selected over simpler alternatives such as plain stochastic gradient descent because it adapts the learning rate for each parameter individually using running estimates of the gradient's first and second moments, which typically leads to faster and more stable convergence for deep convolutional architectures. Because the reconstruction and classification losses are both computed from the same shared encoder output, gradients from both objectives jointly shape the learned representation during backpropagation, reinforcing features that are useful for both tasks at once.

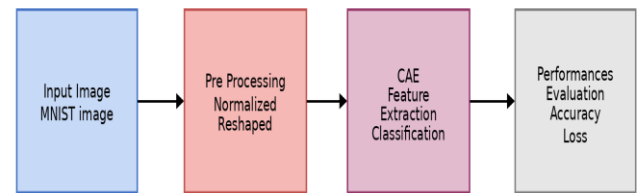


Figure 1: The Proposed Methodology

The end-to-end pipeline: the input MNIST image is pre-processed (normalized and reshaped), passed through the CAE for feature extraction and classification, and the output is evaluated for accuracy and loss.

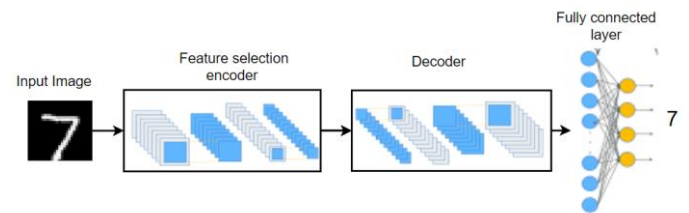


Figure 2: Architecture of CAE

The encoder-decoder structure of the convolutional autoencoder: the encoder extracts and compresses features from the input digit image, the decoder reconstructs it, and a fully connected layer performs the final digit classification (shown here for an input digit "7").

E. Proposed Algorithm

Algorithm 1 summarizes the high-level training procedure described above, from the labeled MNIST input through to the final performance evaluation.

Algorithm 1: CAE TRAINING PROCEDURE

Start
Stage 1: Input MNIST images, which are labeled
Stage 2: Training starts when images are provided to the hybrid model
Stage 3: Passing them to an autoencoder having convolutional layers for feature extraction
Stage 4: after that, passing them to a fully connected layer for classification.
Stage 5: output
End

F. Pseudocode

Algorithm 2 details this procedure in pseudocode, clarifying the split of the data into training, validation, and test subsets and the order in which model fitting and evaluation are performed.

Algorithm 2: PSEUDOCODE

START
Training images=X, Label=Y
INPUT ← Training images, Label
CAE(X): apply the convolutional autoencoder to a training image X
X-Train, Y-Train, X-Val, Y-Val, X-Test, Y-Test
← split(X, Y) into training, validation, and test subsets
Train the model: Model.fit(X-Train, Y-Train), monitoring (X-Val, Y-Val)
After training completes, evaluate: Model.predict(X-Test) and compare against Y-Test
Output: reconstructed images and predicted digit classes for the test set
END

IV. EXPERIMENTAL RESULTS

A. System Specification

The hardware used in this suggested methodology is a 2.20 GHz HP 640 model Core i7 7th generation laptop with 12 GB RAM, 250 M2 SSD, and Windows 10 64-bit operating system. The Python language is used to execute code using Jupyter Notebook.

B. Dataset Description

As described in Section III.A, this research uses the MNIST dataset of 60,000 training and 10,000 testing images of handwritten digits (0–9), each 28×28 pixels.

C. Results and Discussions

This section presents the results of the proposed CAE on the MNIST dataset described above. To assess the consistency of the proposed CAE, its performance is reported separately for two data splits, referred to throughout this section as category 1 and category 2; the same set of training-metric, confusion-matrix, and ROC-curve figures is reported for each split below.

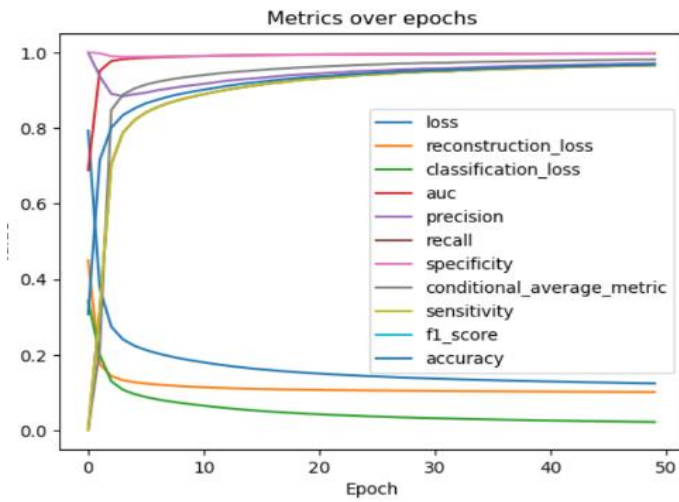


Figure 3: Training Metrics over Epochs -Category 1

Loss, accuracy, precision, recall, and related metrics tracked across 100 training epochs for the category 1 data split, showing how the model converges as training progresses.

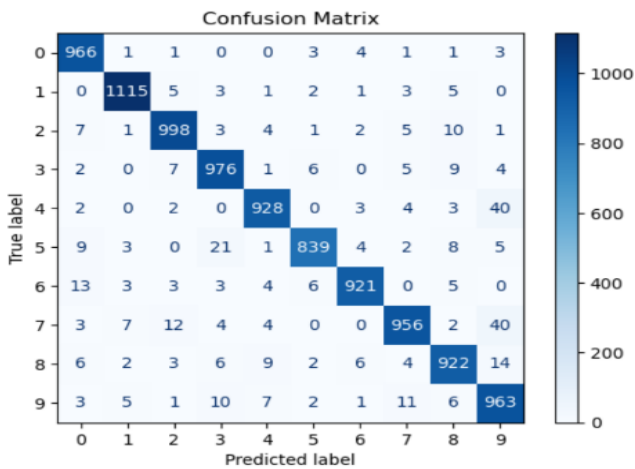


Figure 4: Confusion Matrix-Category 1

Predicted versus true digit labels (0–9) for the category 1 test set. The strong diagonal shows most digits are classified correctly, while the off-diagonal cells show where the model occasionally confuses one digit for another.

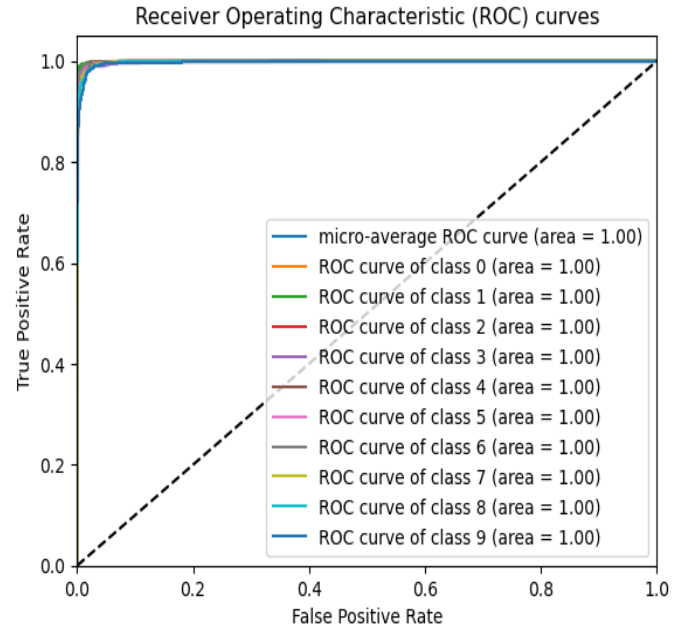


Figure 5: ROC Curves-Category 1

Receiver Operating Characteristic curves for each digit class (0–9) plus the micro-average, for the category 1 split.

Figures 3, 4, and 5 show the training metrics, confusion matrix, and ROC curves for the category 1 data split. The algorithm has been run for 100 epochs. The AUC and accuracy graphs show that the model gives efficient results both for training and testing samples revealing the best performance for this division of data. The training accuracy is 95%, and the model validation accuracy is 96%. As revealed by the loss graph, the error nearly reduces to 0.3 at the end of the last epoch, showing high performance of the model. A closer look at the confusion matrix in Figure 4 shows that the model's errors are concentrated among a small number of digit pairs rather than spread evenly across all classes: the two largest confusions are 40 images of a true 7 predicted as 9, and 40 images of a true 4 predicted as 9. Every digit class is still classified correctly in at least 93% of its instances, with digit 7 having the lowest per-class accuracy (93.0%), consistent with the strong overall accuracy and precision reported above.

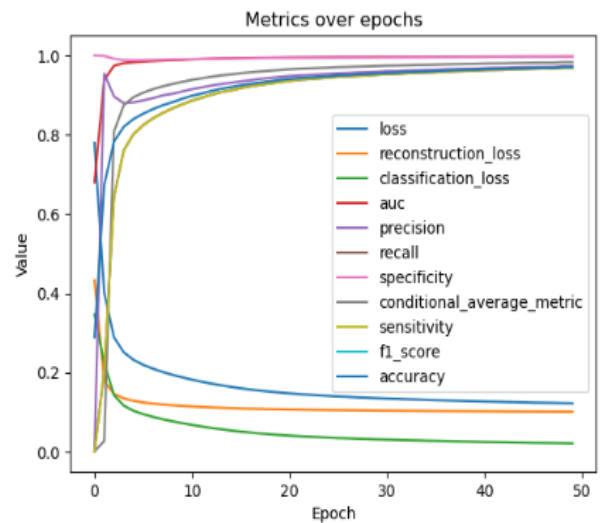


Figure 6: Training Metrics over Epochs-Category 2

The same set of training metrics as Figure 3, tracked across 100 epochs for the category 2 data split.

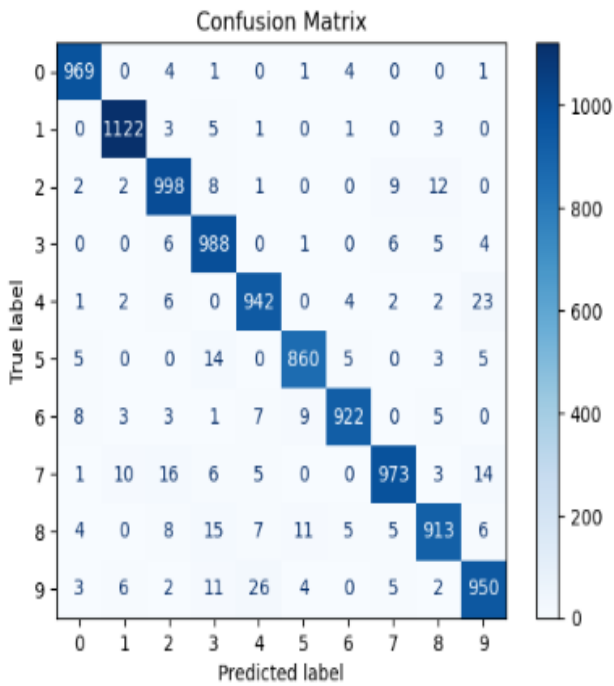


Figure 7: Confusion Matrix-Category 2

Predicted versus true digit labels for the category 2 test set, in the same format as Figure 4.

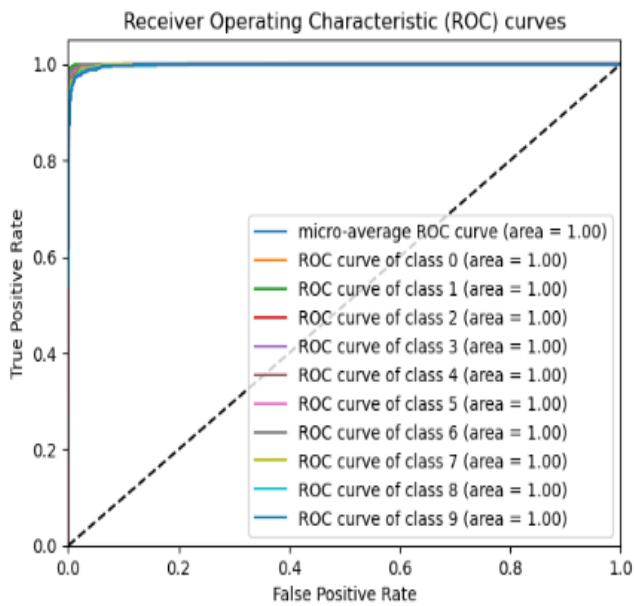


Figure 8: ROC Curves-Category 2

Receiver Operating Characteristic curves for each digit class for the category 2 split, in the same format as Figure 5.

Figures 6, 7, and 8 show the training metrics, confusion matrix, and ROC curves for the category 2 data split. The algorithm has been run for 100 epochs. The AUC and accuracy graphs show that the model gives efficient results both for training and testing samples revealing the best performance for this division of data. The training accuracy is 95%, and the model validation accuracy is 96%. As revealed by the loss graph, the error nearly reduces to 0.3 at the end of the last epoch, showing high performance of the model. As with category 1, the largest confusion in Figure 7 again involves the digits 9 and 4: 26 images of a true 9 were predicted as 4, and 23 images of a true 4 were predicted as 9. Every digit class is classified correctly in at least 93% of its instances, with digit 8 having the lowest per-class accuracy (93.7%), showing a similar error pattern to category 1 despite the two splits being evaluated separately.

D. Comparison chart

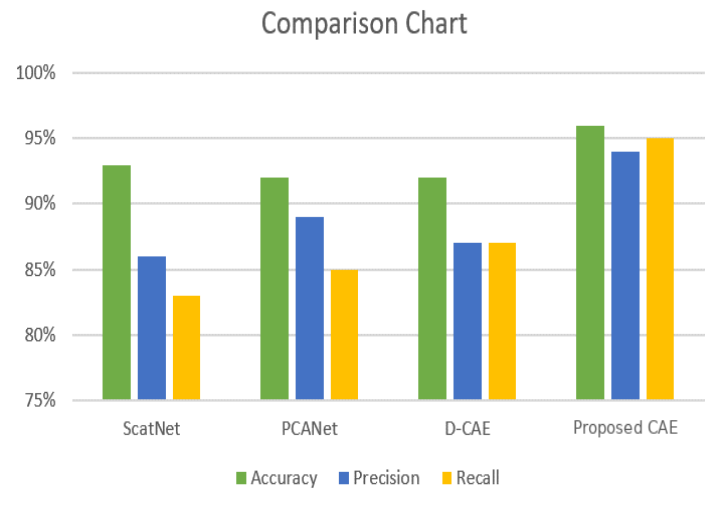


Figure 9: Comparison Chart

Accuracy, precision, and recall of the proposed CAE against three baseline models—ScatNet, PCANet, and D-CAE — on the MNIST dataset.

The proposed CAE is compared with three established baselines—ScatNet, PCANet, and D-CAE on the MNIST dataset (Figure 9). The proposed CAE achieves 96% accuracy, 94% precision, and 95% recall, outperforming ScatNet (93% accuracy, 86% precision, 83% recall), PCANet (92% accuracy, 89% precision, 85% recall), and D-CAE (92% accuracy, 87% precision, 87% recall) across all three metrics. This advantage is most pronounced in recall, where the proposed CAE improves on the next-best baseline, D-CAE, by 8 percentage points, suggesting that the convolutional feature extraction and shared-encoder design help the model identify a larger share of true positive instances than the baseline approaches.

The improved performance of the proposed CAE relative to ScatNet, PCANet, and D-CAE can be attributed largely to its architecture. Unlike ScatNet and PCANet, which rely on fixed, hand-designed filter banks, the proposed CAE learns its convolutional filters directly from the data, allowing the feature representations to adapt to the specific structure of the MNIST digits rather than to a generic, task-independent basis. Compared with D-CAE, the proposed model's shared encoder is optimized jointly for reconstruction and classification, so gradients from both tasks continually reinforce features that are useful for recognizing digit identity, not only for minimizing pixel-level reconstruction error. This joint objective appears to be the main source of the recall improvement noted above, since features that generalize well across classes are more likely to separate visually similar digits, such as 4, 7, and 9, that account for most of the observed misclassifications.

V. CONCLUSION AND FUTURE DIRECTION

The Convolutional Autoencoder (CAE) is a reliable solution for image reconstruction and feature extraction, outperforming traditional autoencoders and related models (ScatNet, PCANet, and D-CAE) in terms of classification accuracy on the MNIST dataset. By integrating convolutional layers and additional compression stages, the CAE improves the model's ability to reconstruct images and extract relevant features, resulting in improved performance in digit classification. The shared encoder within the hybrid model successfully integrates the strengths of unsupervised and supervised learning, resulting in a significant improvement in model accuracy and performance. The evaluation metrics affirm the CAE's potential as a strong alternative for tasks requiring both image reconstruction and classification, making it a valuable contribution to the field of deep learning and image processing. These conclusions are drawn specifically from performance on the MNIST benchmark, however, and claims of

general real-world applicability should be made cautiously until the CAE has been evaluated on more complex, real-world image datasets.

These results should be interpreted within the scope of the present study. All experiments were conducted on the MNIST dataset, a well-controlled benchmark of small grayscale digit images, and evaluation on more complex or noisier image datasets would help establish how well the proposed architecture generalizes beyond this setting. Likewise, the comparison against ScatNet, PCANet, and D-CAE is based on classification accuracy, precision, and recall; a fuller comparison would additionally examine training and inference time, memory footprint, and sensitivity to hyperparameter choices. Future work will therefore apply this model to additional classification problems and datasets, explore further architectural optimization, and include a detailed runtime comparison against the baseline methods.

REFERENCES

- [1] M. Aamir, N. Mohd Nawi, and F. Wahid et al., "A deep contractive autoencoder for solving multiclass classification problems," *Evolutionary Intelligence*, vol. 14, pp. 1619–1633, 2021, doi: 10.1007/s12065-020-00424-6.
- [2] K. Berahmand, F. Daneshfar, E. S. Salehi, Y. Li, and Y. Xu, "Autoencoders and their applications in machine learning: a survey," *Artificial Intelligence Review*, vol. 57, no. 2, 2024, doi: 10.1007/s10462-023-10662-6.
- [3] A. Bachayo, Z. Ahmed, and S. Affrah, "A model design for smart home security system using (IoT) with CCTV camera," *International Journal of Computing and Related Technologies*, vol. 3, no. 2, pp. 29–42, 2022.
- [4] P. Li, Y. Pei, and J. Li, "A comprehensive survey on design and application of autoencoder in deep learning," *Applied Soft Computing*, vol. 138, p. 110176, 2023, doi: 10.1016/j.asoc.2023.110176.
- [5] M. A. Bamboat, A. A. Laghari, H. Li, A. A. Khan, and M. A. Qaimkhani, "Quality of experience assessment of over the top services," in *Proc. 2024 4th Int. Conf. Blockchain Technology and Information Security (ICBCTIS)*, 2024, pp. 145–149.
- [6] K. Jia, L. Sun, S. Gao, Z. Song, and B. E. Shi, "Laplacian auto-encoders: An explicit learning of nonlinear data manifold," *Neurocomputing*, vol. 160, pp. 250–260, 2015, doi: 10.1016/j.neucom.2015.02.023.
- [7] A. S. Shah, A. Dinesh, A. Shah, M. Farooq, A. Maqsood, and M. A. K. Khani, "Smart gas leak detection and emergency response system using IoT for homes," *Journal of ICT, Design, Engineering and Technological Science*, vol. 8, no. 1, pp. 18–23, 2024.
- [8] H. Ali, M. A. K. Khani, A. S. Shah, A. A. Laghari, and M. Minam, "Smart class analytics for assessing group understanding via facial expressions," *Journal of Information Communication Technologies and Robotic Applications*, vol. 16, no. 1, 2025.
- [9] M. A. K. Khani, A. A. Khan, A. B. Brohi, and Z. A. Shaikh, "Designing mobile learning smart education system architecture for big data management using fog computing technology," *International Journal of Imaging and Sensing Technologies and Applications (IJISTA)*, vol. 1, no. 1, pp. 1–23, 2022.
- [10] F. Pichi, B. Moya, and J. S. Hesthaven, "A graph convolutional autoencoder approach to model order reduction for parametrized PDEs," *Journal of Computational Physics*, vol. 501, p. 112762, 2024, doi: 10.1016/j.jcp.2024.112762.
- [11] A. Dashnyam, A. Peljid, B. Natsag, A. Tseden-Ish, Z. Ali, M. A. K. Khani, and S. Hassan, "Evaluating block chain based voting systems: An empirical study on public administration adoption and challenges," *Pakistan Journal of Life & Social Sciences*, vol. 23, no. 2, p. 56, 2025.
- [12] M. A. K. Khani, M. Usama, A. S. Shah, A. Shah, S. H. Abbas, A. Maqsood, and A. A. Laghari, "Intelligent vehicle number plate recognition system using YOLO for enhanced security in smart buildings," *Journal of ICT, Design, Engineering and Technological Science*, vol. 8, no. 2, pp. 11–17, 2024.
- [13] M. Barlaud and F. Guyard, "Learning a sparse generative non-parametric supervised autoencoder," in *Proc. ICASSP 2021 - 2021 IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, Toronto, ON, Canada, 2021, pp. 3315–3319, doi: 10.1109/ICASSP39728.2021.9414410.
- [14] A. A. Laghari, H. Li, A. A. Khan, Y. Shoulin, S. Karim, and M. A. K. Khani, "Internet of Things (IoT) applications security trends and challenges," *Discover Internet of Things*, vol. 4, no. 1, p. 36, 2024.
- [15] J. Janjua and A. Patankar, "Exploring the impact of denoising autoencoder architectures on image retrieval," *Procedia Computer Science*, vol. 235, pp. 2557–2566, 2024.
- [16] A. S. Shah, A. Maqsood, A. Shah, M. A. K. Khani, J. Anjum, and S. Zafar, "Enhanced airport operations: Automated baggage drop-off and boarding pass generation for travelers," *Journal of Advances in Technology and Engineering Research*, vol. 10, no. 2, pp. 1–6, 2024.
- [17] M. Zulqarnain, R. Ghazali, Y. M. M. Hassim, and M. Aamir, "An enhanced gated recurrent unit with auto-encoder for solving text classification problems," *Arabian Journal for Science and Engineering*, vol. 46, no. 9, pp. 8953–8967, 2021.
- [18] N. Malik, M. A. KaimKhani, A. A. Shaikh, A. B. Brohi, and A. Luhrani, "MCQ's evaluation using Python OCR: An algorithmic implementation and design approach," *International Journal of Artificial Intelligence & Mathematical Sciences*, vol. 2, no. 1, pp. 37–52, 2023.
- [19] M. Aamir, N. M. Nawi, H. B. Mahdin, R. Naseem, and M. Zulqarnain, "Auto-encoder variants for solving handwritten digits classification problem," *International Journal of Fuzzy Logic and Intelligent Systems*, vol. 20, no. 1, pp. 8–16, 2020, doi: 10.5391/IJFIS.2020.20.1.8.
- [20] F. Wahid, Y. Ma, R. Sheikh, and M. Aamir, "An enhanced convolutional neural network based on L1 regularization for segmentation of coronary arteries in X-ray angiograms," in *Proc. 5th Int. Conf. Advancements in Computational Sciences*, 2024.
- [21] S. Chen and W. Guo, "Auto-encoders in deep learning—A review with new perspectives," *Mathematics*, vol. 11, no. 8, p. 1777, 2023, doi: 10.3390/math11081777.
- [22] H. Dong and I. V. Kotenko, "Convolutional variational autoencoders and resampling techniques with generative adversarial network for enhancing Internet of Thing security," *Pattern Recognition and Image Analysis*, vol. 34, pp. 562–569, 2024, doi: 10.1134/S1054661824700366.

- [23] F. Laakom, F. Sohrab, J. Raitoharju, A. Iosifidis, and M. Gabbouj, "Convolutional autoencoder-based multimodal one-class classification," arXiv:2309.14090, 2023.