A Comparative Analysis of Feature Extraction Techniques on MNIST Digit Classification

Theo Siemens-Rhodes
Department of Electrical and Computer Engineering
University of British Columbia
`theo.siemensrhodes@gmail.com`

Abstract

We perform a thorough comparison of different feature extraction methods applied to the MNIST handwritten-digit classification problem. By isolating the feature extraction stage and utilizing a consistent Random Forest classifier, we compare raw pixel intensities, principal-component projections (PCA), fixed convolution filters, learned convolutional encodings, and autoencoder-derived embeddings. Results demonstrate the superior efficacy of learned convolutional features, achieving near-perfect accuracy with a low-dimensional embedding. This analysis provides actionable insights for practitioners considering feature extraction approaches under various resource constraints.

1 Introduction

Feature extraction serves as a crucial component in image classification pipelines, converting raw image data into meaningful representations that significantly influence the performance of downstream classifiers. Although modern machine learning methods can learn complex decision boundaries, the choice of feature extraction approach often determines the efficacy and computational efficiency of classification systems. A thorough comparative analysis of feature extraction methods is therefore indispensable, as it guides practitioners toward selecting the most suitable representation for their specific application and resource constraints.

In this study, we systematically compare multiple feature extraction techniques applied to the MNIST handwritten digit dataset. The methods explored range from straightforward approaches, such as raw pixel intensities and linear projections (PCA), to more sophisticated and non-linear approaches, including fixed convolutional filters, convolutional neural networks (CNNs), and autoencoder-derived features. Each technique captures unique aspects of image data, such as structural edges, stroke variations, and global shape characteristics, thus affecting classifier performance differently.

To ensure an unbiased comparison, we employ a consistent Random Forest classifier with default hyperparameters and no hyperparameter tuning across all experiments. By keeping the classifier configuration fixed, differences observed in classification accuracy can be attributed directly to the effectiveness of the respective feature extraction techniques. This experimental design enables a clear interpretation of each method's relative performance, isolating the role of representation quality from that of classifier complexity.

The findings of this comparative analysis provide actionable insights, clarifying how different feature extraction strategies balance accuracy, representation compactness, and computational demands. Practitioners can leverage these insights to make informed decisions when deploying image recognition models under varying conditions and constraints.

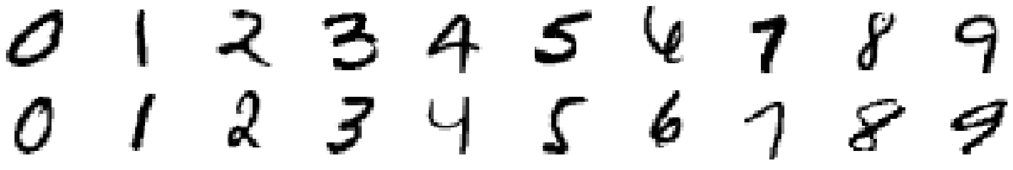
39th Conference on Neural Information Processing Systems (NeurIPS 2025).

2 Dataset and Experimental Setup

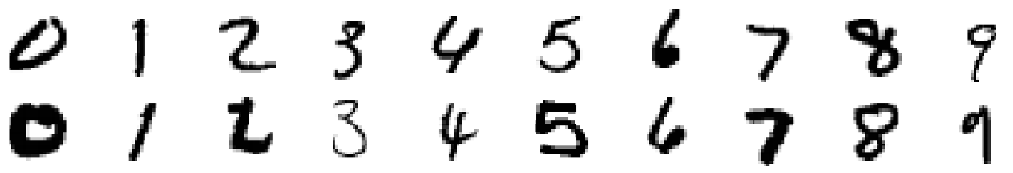
The MNIST dataset is a widely-used benchmark for evaluating digit classification methods. It contains 70,000 grayscale images of handwritten digits (0 to 9), each with dimensions of 28×28 pixels. The dataset was partitioned into training and test subsets using a randomized, stratified 80/20 split, resulting in 56,000 training images and 14,000 test images. This stratified approach ensures balanced digit class distributions across both subsets, providing a robust and representative framework for comparative feature extraction experiments.

Prior to feature extraction, the dataset underwent minimal preprocessing to maintain a realistic evaluation environment. Pixel intensities were normalized linearly from their original scale (0 to 255) to a standardized [0,1] range, facilitating fair comparison across different extraction methods. Beyond this basic normalization step, no further data augmentation, enhancement, or distortion was applied, thereby ensuring that observed performance differences stem solely from the effectiveness of each feature extraction approach rather than preprocessing biases.

In Fig. 1 we show some examples of digits in the training and testing sets, inverted for visual clarity.



(a) Digit samples from MNIST training set



(b) Digit samples from MNIST test set

Figure 1: MNIST database handwritten digit examples

3 Feature Extraction Techniques

3.1 Raw Pixel Features

The most straightforward approach to feature extraction involves using the raw pixel intensities of the images. Each 28×28 grayscale image is flattened into a 784-dimensional vector, with pixel values normalized to the $[0, 1]$ range. This method retains all the original information present in the image without any transformation or abstraction.

Including raw pixel features in our comparative analysis serves as a baseline to assess the effectiveness of more sophisticated feature extraction techniques. By evaluating the performance of a classifier trained on raw pixels, we can determine the extent to which additional processing or transformation of the data contributes to improved classification accuracy. This baseline is crucial for understanding the trade-offs between simplicity, model performance, and computational costs between feature extraction techniques.

3.2 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a statistical technique that transforms the original high-dimensional data into a lower-dimensional space by identifying the directions (principal components) that capture the maximum variance in the data. By projecting the data onto these components, PCA reduces dimensionality while preserving as much variability as possible.

In our analysis, we applied PCA to the normalized pixel data, evaluating representations across multiple dimensions (4, 8, 16, 32, 64, 128). This range allows us to observe how the dimensionality of the feature space impacts classification performance. PCA is particularly valuable in scenarios where computational resources are limited, as it can significantly reduce the feature space, leading to faster training and inference times. Including PCA in our analysis helps us understand the trade-offs between the computational cost saving and information loss in the context of image classification that comes with dimensionality reduction.

3.3 Fixed Convolutional Filters

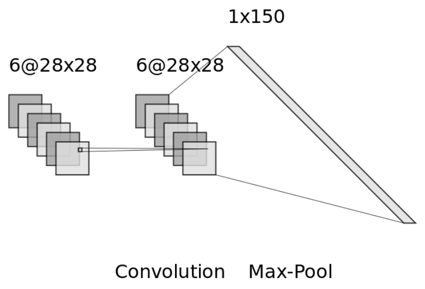


Figure 2: Fixed Filter Architecture.

Fixed convolutional filters are traditional image processing techniques that apply predefined kernels to extract specific features from images. Common filters include Sobel and Prewitt operators for edge detection, Laplacian filters for highlighting regions of rapid intensity change, and Gaussian blurs for noise reduction. These filters are applied uniformly across the image, emphasizing particular spatial patterns.

As illustrated in Fig. 2, our fixed filter pipeline involves applying these filters to the input images, followed by pooling operations to reduce dimensionality. This approach captures essential structural information, such as edges and textures, which are critical for distinguishing between different digit classes. Including fixed convolutional filters in our comparative analysis allows us to assess the effectiveness of handcrafted feature extraction methods compared to learned approaches.

3.4 Learned Convolutional Features

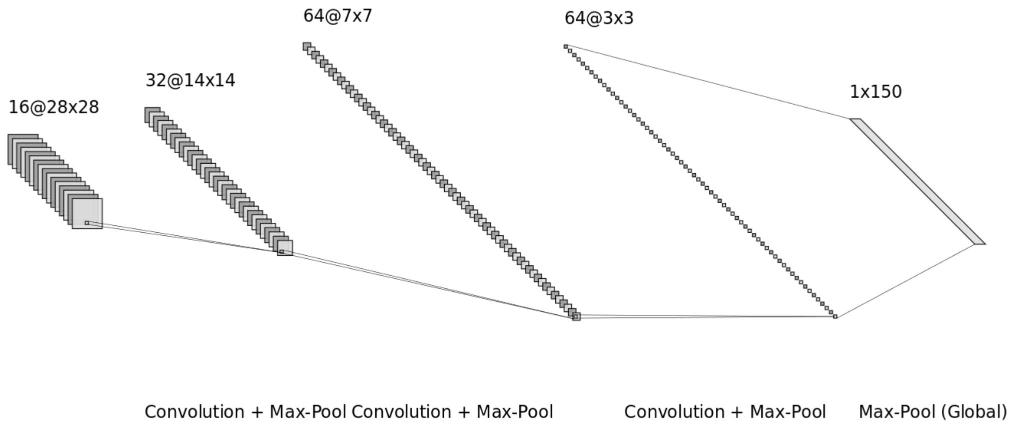


Figure 3: Learned CNN Architecture.

Convolutional Neural Networks (CNNs) are powerful deep learning models that learn hierarchical feature representations directly from labeled data. Unlike unsupervised methods, CNNs utilize the provided class labels during training to identify and emphasize the features most relevant for distinguishing between different classes. This supervised learning approach enables CNNs to focus on patterns that are directly tied to the classification task, such as specific strokes or shapes that differentiate one digit from another.

Our CNN architecture comprises three convolutional layers, each followed by max-pooling, concluding with global average pooling to produce compact feature embeddings, as depicted in Fig. 3. By training on labeled digit images, the CNN learns filters that capture increasingly abstract and task-specific features, from simple edges to complex digit structures, resulting in low-dimensional, highly discriminative representations. This supervised approach contrasts with unsupervised methods like PCA or autoencoders, which focus on variance or input reconstruction without class labels. Including CNN-based features in our comparative analysis enables us to evaluate the advantages of supervised learning for handwritten digit recognition.

3.5 Autoencoder-Based Features

Autoencoders are unsupervised neural network models designed to learn efficient codings of input data. They consist of an encoder that compresses the input into a lower-dimensional latent representation (bottleneck) and a decoder that reconstructs the original input from this representation. Training the autoencoder to minimize reconstruction error encourages the model to capture the most salient features of the data.

In our experiments, we trained autoencoders on the MNIST dataset and used the encoder's output as feature representations for classification. This approach allows us to evaluate the effectiveness of unsupervised feature learning in capturing meaningful patterns in the data. Including autoencoder-based features in our comparative analysis helps us understand the potential of unsupervised methods in scenarios where labeled data is scarce or expensive to obtain.

4 Results

We systematically evaluate the comparative effectiveness of the five feature extraction methods introduced previously. To provide a comprehensive analysis, we first present the overall accuracy achieved by each method across various dimensionalities. Subsequently, we explore the impact of dimensionality reduction in greater detail, emphasizing efficiency and accuracy trade-offs among the considered feature extraction techniques.

4.1 Global Performance Overview

To summarize the comparative performance across all pipelines clearly and precisely, Table 1 presents the exact test accuracy obtained by each feature extraction method at different dimensionalities. Learned convolutional features (CNN encoded) achieved the highest test accuracy, reaching up to 98.5% at 128 dimensions. Autoencoder-derived embeddings also demonstrated robust performance, albeit lower than CNN-derived features, highlighting the benefits of supervised feature learning. Fixed convolutional and PCA-based methods delivered respectable performance, especially considering their unsupervised and computationally simpler nature, with accuracies around 95% and 93%, respectively, at their best-performing dimensions. Raw pixel intensities exhibited competitive accuracy at full dimensionality (784 dimensions), but their performance degraded sharply with dimension reduction, underscoring the redundancy present in unprocessed pixel data.

The clear dominance of learned convolutional features underscores the benefits of supervised training for capturing discriminative, task-specific image features, whereas unsupervised approaches remain valuable when computational constraints are significant. In particular, methods such as PCA and autoencoders provide viable alternatives when labeled data is limited or when rapid deployment with minimal training overhead is essential.

Table 1: Test Accuracy by Feature Extraction Pipeline

Feature Extraction Method	**Dimensions**	**Test Accuracy (%)**
CNN Encoded	128	98.5
CNN Encoded, PCA projected	64	98.5
CNN Encoded, PCA projected	32	98.4
CNN Encoded, PCA projected	16	98.4
CNN Encoded, PCA projected	8	98.2
Raw Pixels	784	96.6
Fixed Convolution Encoded	150	96.1
Fixed Convolution, PCA projected	32	96.0
Fixed Convolution, PCA projected	64	95.5
Autoencoded, PCA projected	32	95.3
Autoencoded, PCA projected	64	95.1
Raw Pixels, PCA projected	128	94.4
Raw Pixels, PCA projected	64	94.3
Autoencoded, PCA projected	128	94.3
Autoencoded, PCA projected	16	94.3
Raw Pixels, PCA projected	32	93.6
Autoencoded, PCA projected	8	93.1
Fixed Convolution, PCA projected	16	92.9
Raw Pixels, PCA projected	16	92.2
Fixed Convolution, PCA projected	8	91.5
CNN Encoded, PCA projected	4	87.4
Fixed Convolution, PCA projected	4	86.8
Autoencoded, PCA projected	4	85.8
Raw Pixels, PCA projected	8	68.3
Autoencoded, PCA projected	4	67.7
Fixed Convolution, PCA projected	4	56.5

4.2 Impact of Dimensionality

Next, we examine the impact of dimensionality reduction more closely, specifically assessing how the classification accuracy of each method varies with decreasing embedding dimensionality. Fig. 4 shows the accuracy curves as the embedding dimension is reduced, clearly illustrating the efficiency and effectiveness of each feature extraction approach.

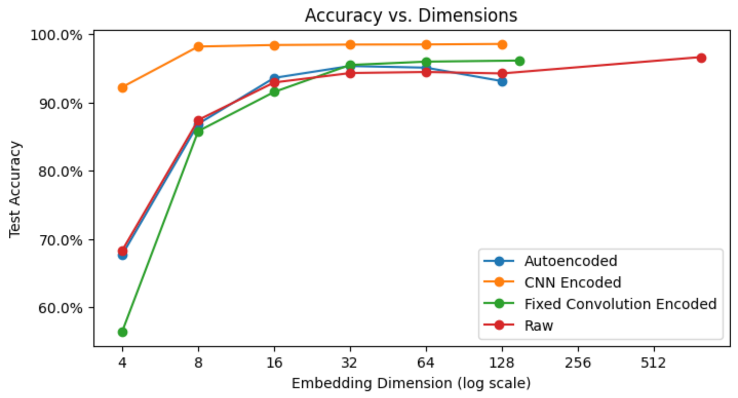


Figure 4: Accuracy vs. embedding dimensions (log scale).

CNN-derived features quickly achieve high accuracy even at extremely low dimensionalities, surpassing 98% at dimensions as low as 8. This rapid saturation indicates that CNN features efficiently capture essential digit characteristics with minimal redundancy. Autoencoder-based embeddings also exhibit rapid improvement with increasing dimensionality, though they do not match CNN performance due to their unsupervised nature. Fixed convolutional features require moderate dimensionalities to reach high accuracy, demonstrating strong baseline effectiveness without learning complexity. Raw pixel data initially suffer significantly with dimensionality reduction, showing substantial redundancy and inefficiency in the original representation, although PCA projections partially mitigate this inefficiency at intermediate dimensions.

Overall, these findings clearly illustrate the benefits of learned and carefully engineered representations over raw data or simplistic dimensionality reductions, providing insights into selecting feature extraction techniques depending on available computational resources and accuracy requirements.

5 Discussion

Our results reveal a clear performance gradient across the evaluated feature extraction methods, shaped by the extent to which each technique incorporates supervision and adaptivity. Learned convolutional encodings, derived from supervised CNNs, consistently outperformed all other representations. By using class labels during training, these networks explicitly learn to emphasize features that separate digit classes, e.g. the presence of loops, junctions, or stroke curvature, rather than relying on general-purpose statistical structure. This results in compact embeddings that not only reduce the computational burden of high-dimensional data but also align tightly with the task-specific decision boundaries of digit classification.

In contrast, autoencoders, while similarly capable of learning compact representations, operate without label supervision. They are optimized to reconstruct inputs rather than to discriminate between classes, which often results in embeddings that capture global data structure but miss finer class-distinctive nuances. Despite this limitation, autoencoders performed remarkably well, especially when paired with PCA for dimensionality control, suggesting their strong potential in unsupervised or semi-supervised learning pipelines where labeled data may be scarce.

Fixed convolutional filters, although simplistic and devoid of learning, showed surprising efficacy. By embedding known edge and gradient detectors like Sobel and Prewitt, they exploit well-established priors about the visual structure of digits. This highlights that even without training, domain knowledge can provide a powerful inductive bias. Their robustness and zero training cost make them particularly appealing in constrained environments, e.g. embedded systems or real-time applications, where resources for learning are limited or unavailable.

PCA, while the most computationally efficient and conceptually straightforward method, showed limitations in its ability to preserve discriminative information at lower dimensions. Its performance degradation at small embedding sizes reflects the fact that variance preservation does not equate to class separation. Nevertheless, its low overhead and tunability make it an attractive preprocessing step, especially in situations where simplicity, speed, or interpretability are critical.

6 Conclusion

This study provided a comprehensive comparative analysis of five feature extraction pipelines for MNIST digit classification, spanning unsupervised, handcrafted, and supervised approaches. By isolating the feature extraction stage and standardizing the downstream classifier, we were able to directly assess the informational and discriminative power of each representation.

Learned convolutional features emerged as the most effective, achieving near-perfect accuracy at compact embedding sizes and setting a clear benchmark for task-specific, supervised representations. Autoencoder-based features, though unsupervised, demonstrated that learned transformations can rival more sophisticated methods when trained on sufficient data. Fixed convolutional filters reaffirmed the enduring value of classical techniques, offering a strong, training-free baseline that competes surprisingly well. PCA, while limited in its ability to preserve class-separating information, remains a fast and reliable dimensionality reduction tool, particularly for linear models.

Ultimately, the results of this analysis highlight that no single method is universally optimal. The choice of feature extraction must balance accuracy, interpretability, compute constraints, and the availability of labeled data. Our findings offer practical guidance for selecting the most suitable representation under a variety of real-world constraints, and underscore the continued relevance of feature design in the deep learning era.

References

[1] A. Nish. MNIST Original Dataset. *Kaggle Datasets*, 2017. `https://www.kaggle.com/datasets/avnishnish/mnist-original`.

[2] T. S. Gunawan, H. Mansor, A. Suryani, and A. M. Zainuddin. Performance evaluation of feature extraction methods for handwritten digit recognition. In *2016 IEEE Int. Conf. on Control System, Computing and Engineering (ICCSCE)*, pages 130–134, IEEE, 2016. `https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7829626`.

[3] W. AlKendi, F. Gechter, L. Heyberger, and C. Guyeux. Advancements and Challenges in Handwritten Text Recognition: A Comprehensive Survey. *Journal of Imaging*, 10(1):18, 2024. `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10817575/`.

AI Acknowledgement

AI assistance, specifically from OpenAI's ChatGPT, was used throughout the preparation of this report to clarify documentation for various machine learning libraries and provide iterative editing suggestions. The model also contributed to refining phrasing and improving clarity and technical precision in the final text. All conceptual design, experimentation, and analysis were conducted by the author.