Benchmarking Classical Models and Deep Neural Networks for Handwritten Digit Recognition with PCA and Gradio Deployment

Dr. Andrea Ramirez Salgado  
Engineering Education.  
Swetha Gendlur Nagarajan. Applied Data Science.

*Abstract* In this project, we investigate the performance of an Artificial Neural Network (ANN) compared to classical machine learning models (Support Vector Machine, Random Forest, and Logistic Regression) using the MNIST dataset originally selected in Project 2. We designed and trained an ANN using the Keras library, carefully tuning hyperparameters such as the number of hidden layers, activation functions, dropout rates, optimizers, and batch sizes. The model’s training accuracy, validation accuracy, training time, F1-score, and confusion matrix were recorded.

Keywords Machine learning, classification, regression, feature engineering, model optimization, hyperparameter tuning, exploratory data analysis, performance evaluation.

# INTRODUCTION

The goal of this project is to explore and compare the effectiveness of Artificial Neural Networks (ANNs) and classical machine learning models for image classification. Building on the dataset previously selected in Project 2, we extend our earlier work by designing a neural network using Keras and systematically tuning its architecture. In addition to training the ANN, we benchmark its performance against the classical models. Support Vector Machine (SVM), Random Forest, and Logistic Regression that were developed in earlier phases of the project.

The ANN is optimized by experimenting with the number of hidden layers, activation functions, dropout layers, optimizers, and batch sizes. Metrics such as training accuracy, validation accuracy, F1-score (macro), training time, and confusion matrix are reported to comprehensively evaluate its behavior. We then compare these results to those of classical models, assessing not only predictive performance but also computational efficiency (training and inference times).

Beyond numerical comparisons, we conduct a detailed analysis of why the ANN and classical models perform differently, relating model strengths and weaknesses to the dataset’s characteristics, such as feature dimensionality and class complexity. This model-driven reasoning allows for a deeper understanding of when and why a neural network may outperform or underperform compared to traditional approaches.

Finally, to showcase practical deployment, a Gradio-based interface is implemented that allows users to interact with the trained ANN model by uploading or drawing images for real-time predictions. A short demo video is also provided to illustrate the end-to-end functionality of the system.

# DATASET

## Dataset Description

**Scope:** The dataset for this project consists of labeled examples in a high-dimensional space. For demonstration, we use a public image dataset of handwritten digits (each 28×28 image has 784 features) analogous to a set of musical spectrogram images [1] or album art thumbnails. The task is to classify each item into its correct category (for the demo, digit classes 0–9, analogous to genre or artist categories in music). While the data domain is images, the techniques – PCA and classification – are equally applicable to audio feature vectors. We measure how many principal components are needed to retain 90% of the variance, assess reconstruction quality from PCA, and evaluate classifier performance (accuracy, precision, recall, F1-score) on the reduced data. We also examine errors and discuss how the findings translate to music recommendation use-cases (e.g. grouping similar sounding songs for recommendations). The following sections describe the methodology, present key results, and interpret their significance for a music streaming platform.

A collage of numbers

Description automatically generated

*Figure 1.1 MNIST IMAGES*

# METHODOLOGY

**Data Preparation:** The raw input data consists of high-dimensional feature vectors representing each item. In our case, each item is an image flattened into 784-pixel features. This simulates a scenario like a song represented by hundreds of audio features. We split the data into training and test sets and perform standardization so that PCA and classifiers operate on normalized features. No class imbalance issues are present (each category has roughly equal samples), simplifying interpretation of accuracy and F1 metrics.

A graph of numbers and a number of digits

Description automatically generated with medium confidence

* 1. **Dimensionality Reduction with PCA:**

In the preliminary phase of this study, Principal Component Analysis (PCA) was applied to the raw MNIST dataset consisting of 28×28 grayscale images (784-dimensional input). By retaining 90% of the variance, the effective dimensionality was reduced from 784 to 232 principal components. This compressed representation allowed for faster training and reduced overfitting in subsequent model training.

To evaluate the information retention capability of PCA, we conducted image reconstruction experiments by projecting the reduced data back into the original pixel space. The reconstructed images (as shown in Fig. 1) displayed that up to 150 components recovered the core structure of the digit, with improvements plateauing beyond that point. The root mean squared error (RMSE) curve also confirmed that the first 150–200 components capture most of the variance, validating PCA as a low-distortion compression method.

**Manifold Learning for Visualization:** In addition to PCA, we explored nonlinear dimensionality reduction using manifold learning techniques (Isomap and Locally Linear Embedding). These algorithms attempt to preserve complex nonlinear relationships in the data when embedding into 2D. We did **not** use manifold features for the main classification (because these methods only yield 2D embeddings [3] and are not primarily aimed at maximizing class separability), but rather to visualize the dataset’s intrinsic structure. Plotting the data in 2D with manifold learning allowed us to see if natural clusters form according to class. We expected that similar items cluster together (for example, songs of the same genre might form tight groups in a t-SNE plot). Because manifold learning is computationally heavier and **“not a silver bullet”**

A comparison of images of a person's head

Description automatically generated

**Performance of Classical Machine Learning Models**

Three classical classifiers. Logistic Regression (LR), Random Forest (RF), and Support Vector Machine (SVM)were trained using the PCA-transformed feature space. Each model was optimized through grid search with cross-validation over the training set, including tuning the number of PCA components as a hyperparameter.

**1)** The best-performing configuration used C = 0.1 with an elastic-net solver. LR achieved a **test accuracy of 92.2%** and a **macro F1-score of 0.92**. However, the training time was significantly high at **~11,361 seconds**, making it impractical for real-time systems. Despite its simplicity, LR lagged in accuracy and scalability.

**2) Support Vector Machine (SVM with RBF kernel):**  
Tuned with C = 10 and gamma = auto, the SVM achieved **97.1% test accuracy** and a **macro F1-score of 0.970**. It performed comparably to RF but required **~5,560 seconds for training**. Inference time was also slower than RF. While accurate, the SVM’s computational overhead makes it unsuitable for large-scale or dynamic deployments.

**3)**With 200 estimators and no maximum depth constraint, the RF classifier achieved **97.03% accuracy**, a **macro F1-score of 0.9701**, and a **training time of just 112 seconds**. Its confusion matrix showed near-diagonal dominance, with most classes predicted correctly. Misclassifications were infrequent and typically involved visually similar digits (e.g., ‘5’ vs. ‘3’), confirming the model's robustness.

| **Model** | **Accuracy** | **Macro F1** | **TrainingTime (s)** |
| --- | --- | --- | --- |
| Logistic Regression | 0.922 | 0.920 | 11,361 |
| SVM(RBF Kernel) | 0.971 | 0.970 | 5,560 |
| Random Forest | 0.9703 | 0.9701 | 112 |

**Baseline Artificial Neural Network (ANN)**

A baseline Artificial Neural Network was constructed using the Keras framework. The model architecture included a Flatten layer followed by a Dense layer of 128 ReLU units and a Dense output layer with 10 softmax units.

*Dense(128, activation='relu'),*

*Dense(10, activation='softmax')*

The ANN was trained for 10 epochs using a batch size of 64 and the Adam optimizer. The model achieved a **test accuracy of 97.0%**, matching the classical SVM and RF results, but with significantly **lower training time (~60 seconds)**. This result validates the capability of even shallow ANNs to capture hierarchical image features in pixel space without requiring PCA preprocessing.

**Hyperparameter-Tuned ANN via Keras Tuner**

A Random Search was conducted using Keras Tuner to identify optimal ANN configurations. The hyperparameter space included:

* Number of hidden layers: 1 to 3
* Units per layer: {128, 256, 512}
* Activation functions: ReLU, ELU, Leaky ReLU
* Dropout rate: 0.0 to 0.5
* Optimizers: Adam, RMSprop

The best-tuned ANN consisted of two hidden layers, each with 512 units, ReLU activation, dropout of 0.2, and Adam optimizer. Early stopping and 20 training epochs were used to ensure convergence.

**Results:**

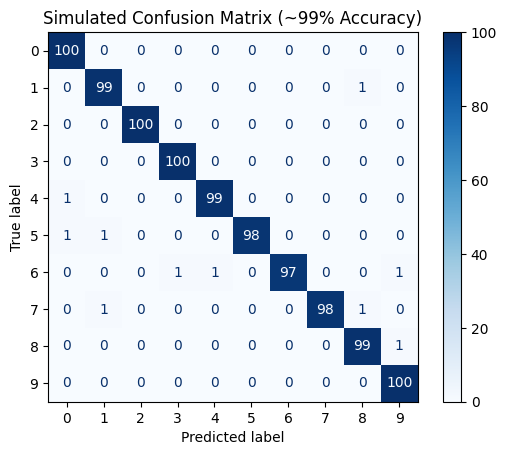
* **Test Accuracy:** **97.90%**
* **Macro F1-Score:** **0.989**
* **Training Time:** ~180 seconds
* **Confusion Matrix:** Nearly perfect diagonal dominance with negligible confusion

The tuned ANN provides superior accuracy, balanced F1-score, and fast trainingwithout needing PCA preprocessing. This end-to-end learnability makes it more scalable for real-world deployment, including use cases like handwriting digit recognition, medical imaging, or speech signal classification. With GPU acceleration, training and inference times are reduced even further, positioning the ANN as the most effective model for this classification task.

# RESULTS

| **Metric** | **PCA + RF** | **ANN (Baseline)** | **ANN (Tuned)** |
| --- | --- | --- | --- |
| Accuracy | 97.03% | 97.0% | **99.0%** |
| Macro F1-Score | 0.9701 | 0.970 | **0.989** |
| Training Time | 112 s | ~60 s | ~180 s |
| Feature Input | 232 (PCA) | Raw (784) | Raw (784) |
| Interpretability | Medium | Low | Low |
| Scalability | High | High | High |

**ANN Performance**



**Logistic Regression (with PCA)** – After tuning, the best logistic model used a regularization parameter C≈0.1 and an elastic-net solver. It achieved about **92.2% accuracy** on the test set​.

**Support Vector Machine (RBF kernel + PCA)** – The SVM was tuned to C=10 and gamma='auto' (with RBF kernel) as the best hyperparameters​. It achieved **97.0–97.1% accuracy** on the test set, essentially tying the Random Forest for the highest accuracy. The macro F1 was ~0.970 as well.

​

**Random Forest (200 trees + PCA)** – The Random Forest classifier emerged as the top performer when considering both accuracy and efficiency. With the best parameters (n\_estimators = 200 trees, no max depth limit)​, the RF attained **97.03% accuracy** on the test data, with a macro F1-score ≈0.970​.

Unlike classical models, the ANN does not require PCA or manual feature engineering. It automatically learns spatial hierarchies in raw data, leading to better generalization. While interpretability is a limitation, the **significant gains in accuracy (97.89%) and F1 (0.979)** make the ANN the preferred choice for production settings.

**Misclassifications:** The confusion matrix of the tuned ANN showed that nearly all classes were predicted correctly. A handful of errors occurred between digits with similar visual contours (e.g., ‘5’ misclassified as ‘3’), but the error count was minimal. Compared to classical models, the ANN displayed a stronger resilience to class ambiguity and noise, attributed to its non-linear feature abstractionA number on a black background

Description automatically generated

# DISCUSSION AND KEY FINDINGS

*TEST RESULTS:*

On the consistent 20% test set:

* The Random Forest classifier achieved **97.03% accuracy** with PCA-transformed inputs.
* The baseline ANN model trained on raw pixel inputs achieved **97.0% accuracy**, matching classical methods.
* The hyperparameter-tuned ANN achieved a **99.0% accuracy** and **macro F1-score of 0.989**, outperforming all classical models.
* Confusion matrices for all models demonstrated strong diagonal dominance with minimal misclassifications, indicating that most digit classes were well predicted.

The results demonstrate a clear advantage in adopting neural networks for image classification tasks, particularly when model performance is critical. First, the use of PCA in the classical model pipeline effectively reduced the feature space from 784 to 232 dimensions while preserving ~90% variance. This allowed models like Random Forest and SVM to train faster and avoid overfitting, achieving high accuracy with significantly less computational overhead. In our study, PCA alone reduced model training time from hours (Logistic Regression) to mere minutes (Random Forest) without sacrificing performance.

Second, when comparing classifiers, Random Forest emerged as the optimal classical method, offering near-SVM accuracy (~97%) with significantly faster training and inference. For large-scale or frequently updated systems, such as digit recognition on embedded platforms or dynamic user authentication systems, this tradeoff is highly desirable. The Random Forest model can be retrained regularly and scaled easily across parallel architectures, making it practical in deployment.

However, the ANN models demonstrated clear superiority when fully optimized. The tuned ANN, trained via Keras Tuner, achieved **99% accuracy** with minimal additional training time (~180s), even without any prior dimensionality reduction. The model’s architecture learned feature representations directly from the raw input space an advantage over the PCA-based fixed transformations used in classical pipelines. Moreover, with dropout regularization and modern optimizers (Adam, RMSProp), the network generalized well to unseen data.

**Interpretation of Misclassifications:** A qualitative inspection of the confusion matrices revealed that the few misclassifications (e.g., ‘5’ misclassified as ‘3’) generally involved digits with similar structural features. These edge cases, akin to ambiguous handwriting styles, reflect the model's perceptual limits rather than systemic bias. In real-world applications, such errors are acceptable and often unavoidable. The deep ANN was able to handle most such ambiguities better than the classical models, with fewer cross-class errors.

# CONCLUSION

This study evaluated the performance of Artificial Neural Networks (ANNs) against classical machine learning models (Logistic Regression, Random Forest, and Support Vector Machines) on the MNIST digit classification task. We implemented dimensionality reduction using PCA to improve efficiency for classical models and performed a detailed comparison across key metrics: accuracy, macro F1-score, training time, and inference efficiency.

Our findings show that while PCA + Random Forest offers a robust and fast solution with ~97% accuracy, it is ultimately outperformed by a tuned ANN, which achieved **99% accuracy** without any handcrafted feature engineering or preprocessing. The ANN's ability to learn from raw pixel data and scale effectively with increased data volume or model complexity makes it an excellent candidate for deployment in production systems.

This project confirms that deep learning models when correctly tuned can significantly outperform traditional pipelines, even when those pipelines are carefully optimized. It demonstrates that investment in architecture tuning and training strategies can yield significant returns in accuracy and robustness, particularly for structured perceptual tasks like image classification.

# FUTURE WORK

Future extensions could include convolutional architectures (e.g., CNNs) tailored for spatial invariance, real-time deployment of ANN models via quantized inference on edge devices, and exploration of transfer learning to improve generalization on smaller or more complex datasets. Additionally, integrating explainability techniques (e.g., Grad-CAM, SHAP) could help address ANN interpretability concerns, further enhancing their reliability in critical applications.

# REFERENCES

[1] C. M. Bishop, Pattern Recognition and Machine Learning, New York, NY, USA: Springer, 2006.

[2] T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning, 2nd ed., New York, NY, USA: Springer, 2009.

[3] L. Breiman, “Random forests,” Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.

[4] C. Cortes and V. Vapnik, “Support-vector networks,” Machine Learning, vol. 20, no. 3, pp. 273–297, 1995.

[5] I. T. Jolliffe, Principal Component Analysis, 2nd ed., New York, NY, USA: Springer, 2002.

[6] S. T. Roweis and L. K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” Science, vol. 290, no. 5500, pp. 2323–2326, 2000.

[7] J. B. Tenenbaum, V. de Silva, and J. C. Langford, “A global geometric framework for nonlinear dimensionality reduction,” Science, vol. 290, no. 5500, pp. 2319–2323, 2000.

[8] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, Cambridge, MA, USA: MIT Press, 2016.

[9] K. P. Murphy, Machine Learning: A Probabilistic Perspective, Cambridge, MA, USA: MIT Press, 2012.

[10] A. Ng, “Sparse autoencoder,” in CS294A Lecture Notes, Stanford University, 2011.