Advancing Handwritten Digit Recognition: Machine Learning Techniques on the MNIST Dataset

1st Monica Suresh
Stevens Institute of Technology
Hoboken, USA
msuresh@stevens.edu

2nd Syed Aziz
Stevens Institute of Technology
Hoboken, USA
saziz2@stevens.edu

3rd Sarah Thuman
Stevens Institute of Technology
Hoboken, USA
sthuman@stevens.edu

***Abstract*—Here is ONE short paragraph summarizing your project including the problem statement, ML algorithms you use to solve the problem, experimental results, and major contribution of this work (e.g., advantages over existing solutions if any) This will be added for Final Report.**

I. Introduction

The recognition of handwritten digits holds paramount importance in diverse applications, including postal mail sorting, bank check processing, and form data entry. The inherent variability in handwriting styles presents a challenge for accurate digit recognition. Our project seeks to address this challenge by implementing and evaluating three machine learning algorithms: the k-Nearest Neighbors (k-NN), Convolutional Neural Network (CNN), and Support Vector Machine (SVM). Through this exploration, our goal is to identify the efficacy of these algorithms in classifying images of handwritten digits, identifying their strengths, weaknesses, and potential areas of enhancement.

The central problem addressed by this project is the accurate recognition of handwritten digits. The variability in handwriting styles poses a significant challenge. This necessitates the exploration of robust machine learning algorithms. The project will assess the k-NN, CNN, and SVM algorithms to determine their effectiveness in classifying images of handwritten digits, with the main goal of improving accuracy in real-world scenarios. In the pursuit of enhancing the accuracy of handwritten digit recognition, the project places a particular emphasis on the fine-tuning of the three machine learning algorithms. An integral aspect of this fine-tuning process involves the optimization of hyperparameters for each algorithm under consideration.

The MNIST (Modified National Institute of Standards and Technology) database, comprising 60,000 training images and 10,000 test images, will be employed in this project. Each image represents a 28x28 pixel grayscale depiction of a single digit ranging from 0 to 9. To ensure the integrity and reliability of the dataset, a comprehensive data preprocessing pipeline has been established. This includes normalization of pixel values, handling missing values, verifying that all values are integers, verifying that there are no duplicate images, standardizing image color, centering digits, verifying that all images are black and white, verifying that all values are non-negative, removing constantly black and white images, and examining outliers. Each step is carefully designed to enhance the quality and suitability of the dataset for training and testing.

The project employs k-NN, SVM, and CNN algorithms, each chosen for its unique strengths in classification tasks. The implementation includes algorithm-specific preprocessing, parameter tuning, and performance evaluation. These algorithms are trained and tested on the MNIST dataset. They are then compared against each other to assess how well each of them performs.

The SVM algorithm, initially employing a linear kernel and a scaled gamma, achieved 83.65% accuracy on the MNIST dataset. Notable strengths were observed for digits 0 and 1, while areas for improvement were identified, which prompted hyperparameter tuning. Switching to an RBF kernel significantly boosted accuracy to 97.56%. Further adjustments to the regularization gamma were explored and the last adjustment changed gamma and C, with detailed results to be included in the final report.

Advancing beyond existing solutions, this project surpasses conventional algorithm implementation by conducting a thorough performance analysis of various methodologies for handwritten digit recognition. Addressing inherent challenges in digit classification and optimizing k-NN, CNN, and SVM, our solution aims to outperform current benchmarks. While feed-forward neural networks typically achieve 90% accuracy [1] our systematic exploration of diverse algorithms, combined with a robust dataset and meticulous preprocessing, positions this project to make significant strides in improving accuracy and reliability. Notably, our SVM algorithm outperforms the achievements of Hafiz Ahamed, Ishraq Alam, and Md. Manirul Islam in handwritten digit recognition. [2] While their use of the polynomial kernel SVM resulted in a commendable 97.83% accuracy, by leveraging the power of the RBF kernel SVM, we have propelled our solution to achieve an even higher test accuracy of 98.13%.

II. Related Work

Handwritten digit recognition is a well-explored area in machine learning, with various methodologies ranging from basic image processing techniques to advanced machine learning models. These approaches provide valuable insights into effective strategies for digit classification, each contributing unique perspectives and strengths.

In the paper NeuroWrite: A Multimodal Approach for Handwritten Digit Classification, NeuroWrite employs a multimodal approach, combining Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), to enhance digit recognition by leveraging diverse architectural strengths. This method has demonstrated high accuracy rates, such as those reported by Pandea and Haghighi (98.8%) and Asish (97.0%), achieved through training a CNN using TensorFlow and Keras on the MNIST dataset. The complexity of the combined CNN and RNN architecture, while computationally demanding, underscores the potential for improved recognition in complex digit classification tasks. The approach focuses on the Devanagari script, employing CNNs to automate and enhance digit recognition. Despite its tailored application, the methodology underscores the importance of adaptable neural network architectures, particularly in handling the intricacies of various scripts and handwriting styles [3].

K-Nearest Neighbor Classifier with Sliding Window Technique by Grover and Toghi introduces an innovative approach combining the K-nearest neighbor classifier with a sliding window technique. This method aims to improve accuracy without intensive pre-processing, thereby addressing spatial misalignment issues. However, the increased computational complexity and potential border information loss present challenges that need balancing against the accuracy gains [4].

Comparative Study of Neural Networks by Feiyang Chen et al, is a comparative study of CNN, ResNet, DenseNet, and CapsNet on the MNIST dataset highlights the emerging potential of CapsNet. With its unique architecture utilizing capsules, CapsNet offers a promising direction in image recognition, especially in terms of data efficiency and minimizing information loss [5].

Our project, while drawing on these existing methodologies, focuses on a comparative analysis of k-NN, CNN, and SVM algorithms, each with distinct operational mechanisms, to assess their applicability and effectiveness in handwritten digit recognition. The exploration and fine-tuning of these algorithms on the MNIST dataset aim to contribute further to the field by providing practical insights into algorithm selection and optimization for specific classification challenges.

III. Our Solution

In the following sections, we implement the three classification algorithms: k-NN, CNN, and SVM. For each algorithm we compare a baseline to a version(s) with tuned hyperparameters and assess if the tuning affected the accuracy of the models. For SVM, the baseline model uses a linear kernel and gamma = ”scale” where scale = 1/(number of features x the variance of the dataset). The baselines for k-NN and CNN will be added for the final report.

A. Description of Dataset

This project will utilize the MNIST database which is a repository of 60,000 training images and 10,000 test images, each depicting a 28x28 pixel grayscale image of a single digit (0-9). This dataset is accessible through TensorFlow and Kaggle and provides labeled images, forming a solid foundation for training and evaluating our models. If computational constraints arise, we plan to downsample the dataset while ensuring representative sampling.

The preprocessing of the MNIST dataset involved several critical steps to ensure the data’s quality and readiness for effective model training:

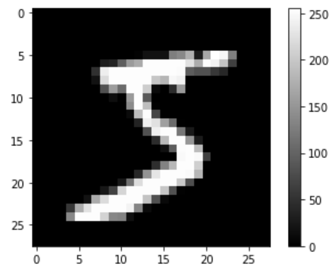
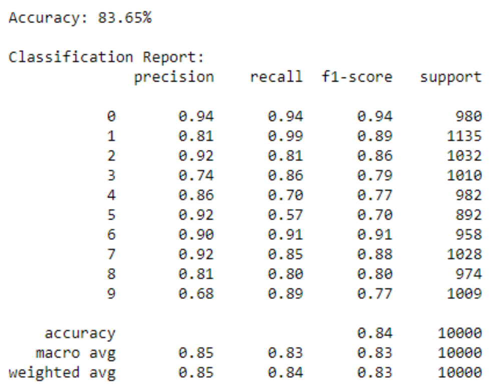


Fig. 1. Example image from MNIST dataset

- Duplicate Removal: duplicates were identified and removed to ensure the dataset’s uniqueness and prevent the model from memorizing specific examples.
- Missing Value Checks: the dataset was thoroughly checked for missing values to avoid biases and inaccuracies in model training.
- Normalization: pixel values were scaled to a range of 0 to 1. This normalization is crucial as it prevents large gradient values during model training, leading to more stable and efficient learning.
- Integrity Checks: validation was done to ensure that all values are integers and non-negative, maintaining consistency and accuracy in representing pixel intensities.
- Image Standardization: the images were standardized to be black or white, and checks were conducted to ensure that the digits were centered. This standardization simplifies the data and ensures consistency across the dataset, which is vital for the performance of machine learning algorithms.

B. Machine Learning Algorithms

At the heart of our methodology is the deployment and fine-tuning of three key machine learning algorithms: k-NN, CNNs, and SVMs, all targeted at the recognition of handwritten digits. The k-NN algorithm operates by comparing input images with their closest counterparts. Its assumption that similar data points cluster together makes it particularly suited for classifying handwritten digits. On the other hand, CNNs excel in discerning complex patterns and spatial relationships, thanks to their convolutional, pooling, and densely connected layers. Our iterative approach to optimizing CNNs involves crafting their architecture, specifying filter attributes, and adjusting key parameters such as learning rates and dropout rates. CNNs’ ability to deliver high-quality predictions with minimal image preprocessing, combined with their resilience to spatial variance, renders them extremely effective in image classification tasks. In contrast, SVMs focus on identifying the optimal hyperplanes to distinctively separate different classes. Our iterative process with SVMs concentrates on refining hyperparameters like the cost parameter (C), gamma, and the kernel type. This involves experimenting with various kernel options, including non-linear ones like the RBF, to boost overall model efficacy. Such adjustments enable SVMs to tackle multi-class classification challenges, exemplified by the MNIST dataset,



```
Accuracy: 83.65%

Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.94      0.94       980
           1       0.81      0.99      0.89      1135
           2       0.92      0.81      0.86      1032
           3       0.74      0.86      0.79      1010
           4       0.86      0.70      0.77       982
           5       0.92      0.57      0.70       892
           6       0.90      0.91      0.91       958
           7       0.92      0.85      0.88      1028
           8       0.81      0.80      0.80       974
           9       0.68      0.89      0.77      1009

    accuracy                           0.84     10000
   macro avg       0.85      0.83      0.83     10000
weighted avg       0.85      0.84      0.83     10000
```

Fig. 2. SVM baseline statistics

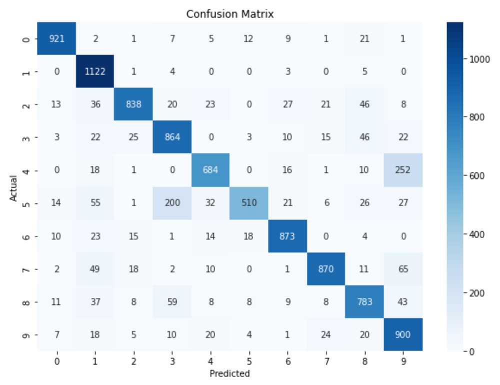


Fig. 3. SVM baseline confusion matrix

by methodically breaking down the problem into manageable components

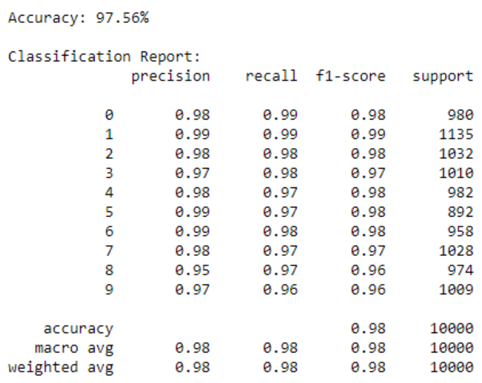
C. Implementation Details

The SVM algorithm was implemented using scikit-learn's SVC class. As mentioned previously the baseline parameters used for SVM were linear kernel and gamma equal scale. The CNN algorithm will be implemented using tensorflow, specifically, Keras' Sequential model. The k-NN algorithm will be implemented using scikit-learn's KNeighborsClassifier class. Once the CNN and k-NN algorithms have been implemented the baselines will be included here before detailed discussion of the implementations in following sections.

1) SVM: The algorithm's performance on the MNIST dataset has been quantitatively evaluated. With an overall accuracy of 83.65%, implementation without hyperparameter tuning, has set a solid baseline for this classification task. The parameters used for this baseline are linear kernel, C value of 1, and a scale for gamma.

The classification report provides a deeper insight into the models performance, breaking down the precision, recall, and f1-score for each digit:

The model showed high precision for digit 0 at 94%, with a corresponding recall, indicating a strong ability to correctly identify and classify this digit. Digit 1 was recognized with a precision of 81% and an impressive recall of 99%, suggesting



```
Accuracy: 97.56%

Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.99      0.98       980
           1       0.99      0.99      0.99      1135
           2       0.98      0.98      0.98      1032
           3       0.97      0.98      0.97      1010
           4       0.98      0.97      0.98       982
           5       0.99      0.97      0.98       892
           6       0.99      0.98      0.98       958
           7       0.98      0.97      0.97      1028
           8       0.95      0.97      0.96       974
           9       0.97      0.96      0.96      1009

    accuracy                           0.98     10000
   macro avg       0.98      0.98      0.98     10000
weighted avg       0.98      0.98      0.98     10000
```

Fig. 4. SVM model with RBF kernel statisitics

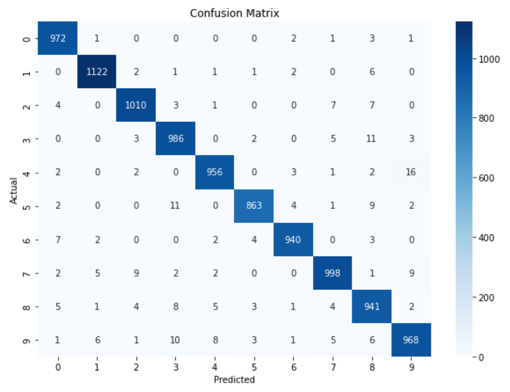


Fig. 5. Confusion matrix from SVM model with RBF kernel

that while some instances were incorrectly labeled as 1 almost all actual 1s were correctly identified.

The recall for digit 5 was notably lower at 57%, pointing to a relatively high number of false negatives, where the model failed to identify 5's correctly. The f1-score, which combines precision and recall into a single metric, was particularly high for 0 and 6 indicating balanced performance for these digits. However, the precision for digit 9 at 68% and the corresponding f1-score suggest room for improvement, as a significant proportion of 9s were mislabeled, or other digits were mistaken as 9.

The SVMs macro average across digits for precision, recall and f1-score is consistent with the overall accuracy, highlighting the algorithm's robustness. The weighted average, which accounts for the imbalanced number of instances for each digit, suggests that the algorithm's performance is reliable across the diverse set of classes.

These findings illustrate the SVMs capabilities and limitations in its current state. Moving forward, tuning the models hyperparameters, such as the regularization term and kernel coefficient, could lead to improvements in precision, and recall, especially for those digits where performance is lagging. Additionally, experimenting with different kernel functions may provide better decision boundaries for the varied shapes and forms present in handwritten digit data.

The first hyperparamter adjustment made to the SVM algorithm was changing the kernel from linear to RBF (radial basis function). This kernel was chosen because it performs better with data that is not linearly separable than the linear kernel does. Since MNIST has nine classification categories that overlap, RBF is a better kernel choice. Simply changing the kernel from linear to RBF resulted in an increase from 83.65% to 97.56%; an over 10% increase in the accuracy of the model.

For this mid-stage report, C was set to 1 for all runs of SVM. C determines how much we want to avoid misclassification. A smaller C value allows more points to be misclassified while a larger C value prohibits the misclassification of more points when fitting the hyperplanes. At least two different C values will be tried to improve the accuracy of the model: 0.1 and 10. Results will be included in final the report.

In our study, the RBF kernel outperformed the linear kernel in SVM. Our focus then shifted to the gamma hyperparameter, which influences the decision boundary's curvature. Higher gamma values introduce more curvature. Notably, reducing gamma to 0.01 significantly decreased model accuracy from 97.56% to 11.35%. This finding led us to concentrate on higher gamma values in subsequent model iterations, as lower gamma values adversely affected the model's predictive accuracy for recognizing numbers in images. The final two values of gamma that were tried were 10 and 100 which resulted in accuracy scores of 92.2% and 96.8% respectively. While this could be considered a good model, it is still lower than the baseline RBF SVM model.

One last version of the RBF SVM model was tried. In this iteration, C was set to 100 and gamma was set to 1000. This resulted in an accuracy score of 98.13%, which was the best iteration. One concern, that still needs to be checked, is whether or not this model is overfitting the training data.

IV. Mid-Stage Conclusion

At this mid-stage of our project, we have successfully implemented and evaluated the Support Vector Machine (SVM) algorithm on the MNIST dataset, establishing a solid baseline for digit classification. The initial results have shown promising directions, especially in terms of the algorithm's ability to classify digits with considerable accuracy. Our exploration of hyperparameter tuning, particularly with the SVM's kernel and gamma value, has already yielded significant insights into the impact of these parameters on model performance.

While the project is still in progress, preliminary findings suggest that careful selection and optimization of hyperparameters are crucial in enhancing the effectiveness of machine learning algorithms for digit recognition tasks. The SVM's performance, specifically, has demonstrated the potential for high accuracy in classifying handwritten digits, although further fine-tuning and comparative analysis with k-NN and CNN are necessary to draw more comprehensive conclusions.

As we move forward, the focus will be on implementing and optimizing the k-NN and CNN algorithms, further refining the SVM, and conducting a comparative analysis of all three algorithms. This will enable us to provide a more detailed assessment of their respective strengths and limitations in the context of handwritten digit recognition. The final phase of the project will also involve a thorough comparison with existing solutions, aiming to position our findings within the broader landscape of machine learning applications in digit classification.

In conclusion, this mid-stage report sets the foundation for what promises to be a comprehensive exploration of machine learning techniques applied to a classic problem in the field. The final report will build on this groundwork to present a detailed analysis and comparison of the chosen algorithms, ultimately contributing to the ongoing research and development in the area of digit recognition.

Mid-Stage Report ends here. The following sections are placeholders for discussions that will be included in the final report.

V. Comparison

This section includes the following: 1) comparing the performance of different machine learning algorithms that you used, and 2) comparing the performance of your algorithms with existing solutions if any. Please provide insights to reason about why this algorithm is better/worse than another one.

VI. Future Directions

This section lays out some potential directions for further improving the performance. You can image what you may do if you were given extra 3-6 months.

VII. Conclusion

Will be included in Final Report.

References

[1] S. Ali, Z. Shaukat, M. Azeem, Z. Sakhawat, T. Mahmood, and K. Rehman, "An efficient and improved scheme for handwritten digit recognition based on convolutional neural network," *SN Applied Sciences*, vol. 1, 09 2019.

[2] M. H. Ahamed, S. M. I. Alam, and M. Islam, "Svm based real time hand-written digit recognition system," 01 2019. [Online]. Available: https://www.researchgate.net/publication/330684489_SVM_Based_Real_Time_Hand-Written_Digit_Recognition_System

[3] K. Asish, P. S. Teja, R. K. Chander, and D. D. D. Hema, "Neurowrite: Predictive handwritten digit classification using deep neural networks," 2023.

[4] D. Grover and B. Toghi, "Mnist dataset classification utilizing k-nn classifier with modified sliding-window metric," 2019.

[5] F. Chen, N. Chen, H. Mao, and H. Hu, "Assessing four neural networks on handwritten digit recognition dataset (mnist)," 2019.