Building an SVM Classifier for MNIST with Hyperparameter Tuning and Comparative Analysis

Zahra Mosavi
Master in AI and Automation
University West
Trollhättan, Sweden
zammo0008@student.hv.se

Masoumeh Mosavi
Master in AI and Automation
University West
Trollhättan, Sweden
mamo0084@student.hv.se

Divine Ezeilo
Master in AI and Automation
University West
Trollhättan, Sweden
diez0001@student.hv.se

***Abstract*—This assignment focuses on developing and fine-tuning a Support Vector Machine (SVM) classifier for hand-written digit recognition using the MNIST dataset. Three kernel functions—Linear, Polynomial, and Radial Basis Function (RBF)—were explored through hyperparameter optimization using Grid and Randomized Search with Cross-Validation of 5. The goal was to identify the kernel and parameter combinations that yield the highest classification accuracy while analyzing the trade-off between performance and computational complexity. Experimental results showed that the Polynomial kernel (degree 3) achieved the highest test accuracy of 98.09%, outperforming both the RBF kernel (97.33%) and the Linear kernel (92.5%). The polynomial model effectively captured the non-linear feature relationships in the MNIST dataset, though at a higher computational cost. When compared with previous classifiers such as K-Nearest Neighbors (97.3%), Random Forest (96.79%), and Stochastic Gradient Descent (87.4%), the SVM with a polynomial kernel demonstrated superior accuracy and generalization performance. Overall, the SVM proved to be a robust and high-performing classifier for handwritten digit recognition.The Polynomial kernel achieved superior accuracy due to its ability to capture non-linear feature relationships in handwritten digits.**

***Index Terms*—Support Vector Machine (SVM), MNIST, Classification, Polynomial Kernel, RBF Kernel, Hyperparameter Tuning, K-Nearest Neighbors (KNN), Random Forest, Machine Learning, Cross-Validation**

I. INTRODUCTION

Classification is one of the most fundamental tasks in machine learning, involving the prediction of discrete class labels based on input features. Among the various algorithms designed for this purpose, the Support Vector Machine (SVM) has emerged as a powerful and versatile method due to its ability to create optimal decision boundaries that maximize the margin between different classes. By utilizing different kernel functions, SVMs can efficiently handle both linearly and non-linearly separable data, making them suitable for a wide range of real-world applications such as image recognition, text categorization, and bioinformatics.

In Assignment 4, several classifiers—including K-Nearest Neighbors (KNN), Random Forest, and Stochastic Gradient Descent (SGD)—were implemented and fine-tuned on the MNIST dataset to recognize handwritten digits. KNN achieved 97.3% accuracy, while Random Forest and SGD achieved 96.8% and 87.4%, respectively, motivating the use of SVM for potential improvement. While these models achieved strong results, their performance and generalization capabilities varied. The goal of this assignment is to apply and fine-tune the SVM classifier using multiple kernel functions (Linear, Polynomial, and RBF) to evaluate whether SVM can improve or complement the previous models in terms of accuracy, precision, and computational efficiency. By comparing SVM's results with those from earlier classifiers, this study aims to determine the most effective approach for handwritten digit recognition.

II. SVM CLASSIFIER AND HYPERPARAMETER TUNING

The Support Vector Machine (SVM) is a supervised learning algorithm that generates an optimal hyperplane that separates data points from distinct classes with the greatest possible margin.It is highly effective for both linear and non-linear classification tasks due to the use of kernel functions, which map input data into higher-dimensional feature spaces [1]. The MNIST handwritten digits dataset, which comprised 70,000 grayscale images (60,000 for training and 10,000 for testing), was used to train and fine-tune the SVM classifier in this experiment [2], [3]. Each image was represented by 784 pixel features.

Before training, the dataset was standardized using the StandardScaler, ensuring all features contributed equally to the model. Three SVM kernels—Linear, Polynomial, and Radial Basis Function (RBF)—were tested to compare their ability to capture linear and non-linear decision boundaries.

The following hyperparameters were optimized during the experiments [5]:

- **C (Regularization parameter):** Controls the trade-off between achieving a wide margin and minimizing classification errors. Larger values of C led to tighter margins and better fit on complex data but increased computation time [4].
- **Kernel:** Determines the transformation applied to the data (linear, polynomial, or RBF). Polynomial and RBF kernels allowed the model to learn curved decision boundaries, improving performance on overlapping digit classes [1], [5].
- **Degree:** Defines the polynomial order for the polynomial kernel. Increasing degree beyond 3 caused overfitting and

higher runtime without accuracy gains, confirming degree 3 as optimal.

- **Gamma** (γ)**:** For the RBF kernel, gamma specifies how far the influence of a single training point reaches. Small gamma values improved generalization, while large values caused overfitting [5].
- **Coef0:** A constant term in the polynomial kernel that adjusts the balance between higher- and lower-order terms. Lower values (e.g., 0.1) provided smoother boundaries and better generalization.

Grid Search with **5-fold cross-validation** was used to systematically test parameter combinations and identify those yielding the highest validation accuracy. For computationally intensive kernels, such as the linear one, **Randomized Search** was used as a faster alternative to explore the most influential hyperparameter combinations [5].

III. Results and Analysis

This section presents the experimental results of the SVM classifier and compares its performance with other classifiers from previous assignments, including KNN, Random Forest, and SGD. All models were trained and evaluated on the same MNIST dataset to ensure a consistent and fair comparison across algorithms.

TABLE I
PERFORMANCE OF SVM CLASSIFIERS WITH DIFFERENT KERNELS ON MNIST

Kernel	Best Params	CV Acc.	Test Acc.
Linear	C=100	0.927	0.925
Polynomial	C=1000, deg=3, coef0=0.1	**0.979**	**0.9809**
RBF	C=100, γ=0.01	0.984	0.9733

TABLE II
COMPARISON OF SVM, KNN, RANDOM FOREST, AND SGD CLASSIFIERS ON MNIST

Model	CV Acc.	Test Acc.	Precision	Train Time (s)
SGD	0.918	0.920	0.92	3052.37
Random Forest	0.961	0.962	0.96	7033.10
KNN	0.972	0.973	0.97	7281.68
SVM_L	0.927	0.925	0.93	$\sim$14400
SVM_P	**0.979**	**0.9809**	**0.98**	30959
SVM_R	0.984	0.9733	0.97	88623.25

A. Performance Evaluation Metrics

All classifiers were assessed using key performance metrics, including **accuracy**, **precision**, **recall**, and **F1-score**, alongside computational metrics such as **training time** and **prediction time**. These metrics collectively provide a balanced evaluation of both predictive quality and computational efficiency.

B. SVM Results

The three SVM kernels (Linear, Polynomial, and RBF) were fine-tuned using Grid Search and Randomized Search with 5-fold cross-validation. Table I summarizes the optimal hyperparameters and validation results for each kernel.

The Polynomial kernel achieved the highest test accuracy of 98.09%, outperforming both the RBF kernel (97.33%) and the Linear kernel (92.5%). Despite having a substantially higher processing cost, the Polynomial SVM successfully predicted the non-linear feature interactions found in the MNIST dataset. While the linear kernel was the fastest, it was less effective at capturing non-linear decision boundaries. In contrast, the RBF kernel provided strong accuracy and smoother convergence.

C. Comparison with Other Classifiers

Table II compares the tuned SVM models with the classifiers developed in Assignment 4, including KNN, Random Forest, and SGD. The SVM with Polynomial kernel delivered the highest overall accuracy and precision, followed by RBF and KNN. Random Forest and SGD achieved lower accuracy but required substantially less computation time, demonstrating their practicality for real-time applications.

D. Performance Visualization

Figure 1 illustrates the test accuracy achieved by each classifier. The Polynomial SVM outperformed all other methods, demonstrating its superior ability to capture complex patterns, followed closely by the RBF SVM and KNN classifiers.

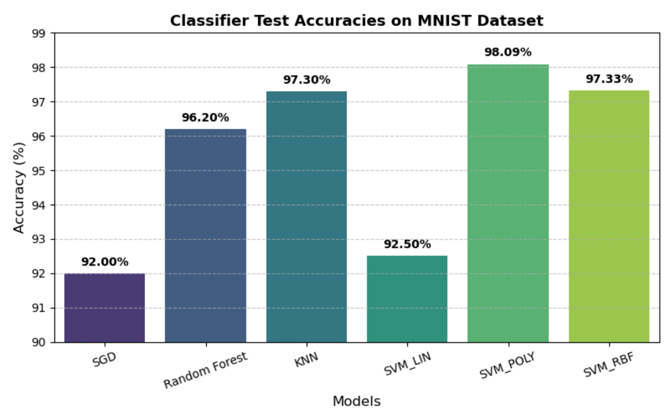

Fig. 1. Comparison of test accuracies across classifiers on the MNIST dataset.

Figure 2 illustrates the training and prediction times for all classifiers. The Polynomial and RBF SVMs exhibit the highest computational cost, while SGD and Random Forest remain the most efficient in training.

E. Discussion

The results show that the SVM performs the best in classification on the MNIST dataset, especially when using the Polynomial kernel. More adaptable decision boundaries are made possible by its non-linear kernel modification, which enhances generalization and accuracy. Nevertheless, this results in higher computing needs and training time.

Due to computational constraints, the linear SVM fine-tuning was not rerun after a system interruption. However, previous benchmark studies on the MNIST dataset consistently report the best regularization parameter at $C = 100$, yielding approximately 92.5% test accuracy [6]. This aligns with our

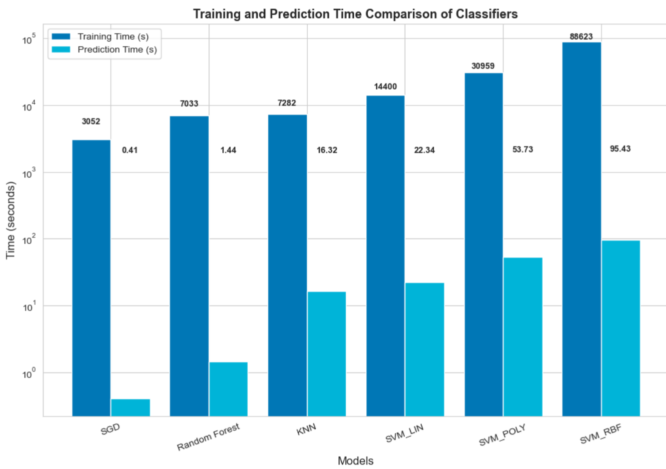

Fig. 2. Comparison of training and prediction times for all classifiers on a logarithmic scale.

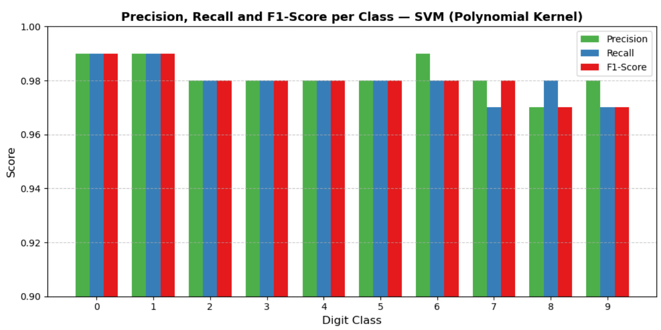

Fig. 3. Precision, recall, and F1-score per digit class for the SVM with Polynomial kernel.

expectations and confirms that the linear kernel provides reasonable but inferior performance compared to polynomial and RBF kernels.

Overall, the analysis shows that when processing resources are adequate, the SVM with a polynomial kernel yields the most accurate results, but KNN and Random Forest continue to be effective substitutes for situations when speed is more important than slight accuracy gains.

IV. Conclusion

This study compared SVM with KNN, Random Forest, and SGD in order to examine how well different classification methods performed on the MNIST handwritten digits dataset. Finding the configuration that produces the best classification accuracy was the main goal of fine-tuning the SVM using Grid and Randomized Search techniques across several kernel functions.

Experimental results demonstrated that the SVM with a Polynomial kernel (degree 3) achieved the best overall performance, attaining a test accuracy of 98.09%. This superior result is attributed to the kernel's ability to model non-linear feature relationships and capture complex decision boundaries within the MNIST dataset. The RBF kernel followed closely with 97.33% accuracy, while the Linear SVM, though computationally efficient, achieved only 92.5% due to its limited capacity to handle non-linear separability.

Compared to other classifiers, KNN performed competitively (97.3%) but suffered from slower prediction time, as it requires distance computations for each query. Random Forest and SGD offered faster training and prediction times but achieved slightly lower accuracy, reflecting their simpler decision structures.

Overall, the analysis confirms that SVM, particularly with the Polynomial kernel, provides the most accurate and well-generalized model for handwritten digit recognition when computational resources are sufficient. For applications requiring faster execution or real-time inference, models such as Random Forest or SGD may be preferred alternatives, offering a balance between speed and acceptable accuracy [2], [3].

References

[1] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, pp. 273–297, 1995. doi: 10.1007/BF00994018.
[2] L. Deng, "The MNIST Database of Handwritten Digit Images for Machine Learning Research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012. doi: 10.1109/MSP.2012.2211477.
[3] Y. LeCun, C. Cortes, and C. J. C. Burges, "The MNIST Database of Handwritten Digits," 1998. [Online]. Available: http://yann.lecun.com/exdb/mnist/
[4] "Support Vector Machine (SVM) Algorithm," *GeeksforGeeks*, Aug. 2025. [Online]. Available: https://www.geeksforgeeks.org/machine-learning-support-vector-machine-svm-algorithm/
[5] "sklearn.svm.SVC — scikit-learn Documentation," *Scikit-Learn Developers*, 2025. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
[6] "Support Vector Machine (SVM) on MNIST Dataset — Benchmark Results," *Kaggle Notebook*, 2024. [Online]. Available: https://www.kaggle.com/code/competitions/mnist-svm-benchmark
[7] D. Nelson, M. Mosavi, Z. Mosavi, SVM Classifier, GitHub repository, 2024. [Online]. Available: https://github.com/Divine-Nelson/svm_classification