Machine Learning Project

Yussuf Mohammad Taha – 20220406

Youssef Mohammad Nabeh – 20220410

Omar Osama Hassan - 20220221

1. Feature Extraction: ResNet50

The feature extraction pipeline uses a pretrained ResNet50 backbone (weights="imagenet") with the network's top removed and a GlobalAveragePooling2D layer to produce compact descriptors.

Descriptor properties

- Dimensionality: the pooled output yields a fixed-length dense vector per image (depends on ResNet50's final conv depth). - Representation: activations encode texture, shape and semantic content. - Advantages: no hand-crafted features required; good generalization when using pretrained weights. - Considerations: features are not rotation-invariant by design and may need augmentation for domain shifts.

2. Classifier Choices

2.1 k-Nearest Neighbors (k-NN)

We used a k-NN classifier with k=3 and Euclidean distance.
Rationale for k=3: small odd k reduces tie probability.
Euclidean distance is appropriate when features are standardized by standardScaler.

2.2 Support Vector Machine (SVM)

An SVM with an RBF (radial basis function) kernel is used, with hyperparameters C=10 and gamma="scale". RBF kernels map input features into a higher-dimensional space and can model complex decision boundaries. The regularization parameter C=10 biases the solver toward fewer margin violations (harder margin), potentially improving separation at the risk of reduced margin generality.

3. Design Rationale: Kernel and k choices

Kernel choice: The RBF kernel is a strong default for complex, non-linearly separable data because it constructs a flexible decision surface while keeping the parameterization compact (C, gamma). If the dataset is high-dimensional and features are well-scaled, RBF often outperforms linear kernels when classes overlap. k selection: For k-NN, small odd values such as 3 are commonly chosen for balanced bias-variance tradeoff.

4. Comparison: Feature Extraction and Classifier Performance

This section compares the chosen feature extraction method (ResNet50 + GlobalAveragePooling) and the two classifiers (k-NN, SVM) used in the project. The comparison considers accuracy, robustness, inference cost, and ease of deployment. Where numerical scores are required, the repository's training cells print accuracy results; these can be inserted or recomputed by running the training notebook or model evaluation script.

4.1 Accuracy and generalization

- Empirical behavior: SVM with RBF often produces stronger decision boundaries when features separate classes non-linearly; k-NN relies heavily on local neighborhood structure and can be more sensitive to noisy features or class imbalance.

4.2 Inference cost and deployment

- k-NN: cheap to train (no explicit training beyond storing feature vectors) but expensive at inference for large datasets because it requires a nearest-neighbor search.- SVM: training can be slower (especially with many samples) but inference is faster and model size is compact (support vectors). SVM can be preferable when low-latency predictions are required.

5. Empirical results (how to include measured numbers)

The notebook contains training cells that compute KNN and SVM accuracy using the extracted features and a StandardScaler. Knn -> 86% accuracy, SVM -> 91% accuracy