Variance-Weighted Centroid Selection for Prototype-Based 1-Nearest Neighbor Classification

Wayne Wang [1]

Abstract

We address the problem of selecting a small set of representative prototypes from a large training set for 1-nearest neighbor (1-NN) classification. We propose a variance-weighted centroid selection algorithm that allocates prototypes proportionally based on within-class variance and uses K-Means clustering to identify representative samples. Experiments on MNIST demonstrate that our method significantly outperforms random selection, with advantages becoming more pronounced at higher compression ratios. At 60× compression (1,000 prototypes from 60,000 training samples), our method achieves 92.47% accuracy compared to 88.56% for random selection. At extreme compression (6,000×, only 10 prototypes), our method maintains 67.01% accuracy while random selection degrades to 39.37%. We also report negative results from two alternative approaches—boundary-first selection and condensed nearest neighbor—providing insights into why typical samples make better prototypes than boundary samples.

1. Introduction

The k-nearest neighbor (k-NN) algorithm is a fundamental non-parametric classifier that predicts the label of a test sample based on the labels of its k closest training samples (Cover & Hart, 1967). Despite its simplicity and strong theoretical guarantees, k-NN suffers from high computational cost at inference time, as it requires computing distances to all training samples.

Prototype selection addresses this limitation by selecting a representative subset of training samples, called prototypes, to use for classification instead of the full training set. The goal is to maintain classification accuracy while dramatically reducing storage and computational requirements.

In this work, we focus on prototype selection for 1-NN classification on MNIST handwritten digits. Our key contributions are:

1. A **variance-weighted centroid selection** algorithm that allocates prototypes proportionally based on within-class variance and uses K-Means to identify representative samples.
2. Comprehensive experiments showing our method's advantage increases with compression ratio, achieving +27.64% improvement over random selection at 6,000× compression.
3. **Negative results** demonstrating that boundary points and misclassified points are poor prototype candidates, contrary to intuition.

2. Problem Formulation

Given a training set $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{0, 1, \ldots, C-1\}$, the prototype selection problem is to find a subset $\mathcal{P} \subset \mathcal{D}$ with $|\mathcal{P}| = M \ll N$ that maximizes classification accuracy on a held-out test set when using 1-NN with $\mathcal{P}$ as the reference set.

The 1-NN classifier predicts:

$$\hat{y} = y_{j^*}, \quad \text{where } j^* = \underset{(\mathbf{x}_j, y_j) \in \mathcal{P}}{\arg\min} \|\mathbf{x} - \mathbf{x}_j\|_2 \tag{1}$$

The compression ratio is defined as N/M. For MNIST with $N = 60,000$ training samples, selecting $M = 1,000$ prototypes yields 60× compression.

3. Proposed Method

3.1. Key Insight

Our method is based on the observation that **good prototypes are typical, representative samples**—not boundary samples. While decision boundaries are defined by samples near class interfaces, the best prototypes are cluster

[1]Department of Computer Science and Engineering, University of California San Diego, La Jolla, CA, USA. Correspondence to: Wayne Wang <waw009@ucsd.edu>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

Algorithm 1 Variance-Weighted Centroid Selection

```
**Input:** Training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$, budget $M$
**Output:** Prototype indices $\mathcal{I}$

// Step 1: Compute within-class variance
**for** each class $c = 0, \ldots, C-1$ **do**
    $\sigma_c^2 \leftarrow \frac{1}{|D_c|} \sum_{\mathbf{x} \in D_c} \|\mathbf{x} - \boldsymbol{\mu}_c\|^2$
**end for**

// Step 2: Allocate prototypes proportionally
**for** each class $c$ **do**
    $M_c \leftarrow \max\left(1, \text{round}\left(M \cdot \frac{\sigma_c^2}{\sum_{c'} \sigma_{c'}^2}\right)\right)$
**end for**
Adjust allocations to sum to exactly $M$

// Step 3: Select prototypes via K-Means
$\mathcal{I} \leftarrow \emptyset$
**for** each class $c$ **do**
    Run MiniBatchKMeans on $D_c$ with $M_c$ clusters
    For each centroid, add index of nearest point to $\mathcal{I}$
**end for**
**return** $\mathcal{I}$
```

centroids that represent the "typical" appearance of each class.

Different classes may have different amounts of within-class variation. For example, in MNIST, the digit "1" has relatively consistent appearance, while "2" and "4" exhibit more variation. Classes with higher variance require more prototypes to adequately cover their data distribution.

3.2. Algorithm

Our variance-weighted centroid selection algorithm proceeds as follows:

Complexity: The algorithm runs in $O(N \cdot M \cdot t)$ time where t is the number of K-Means iterations. Using MiniBatchK-Means with batch size 256 and $t = 100$ iterations, selection takes 2–680 seconds depending on M.

4. Experiments

4.1. Dataset and Setup

We evaluate on the MNIST handwritten digit dataset (LeCun et al., 1998) containing 60,000 training and 10,000 test images of size 28×28 pixels. Images are flattened to 784-dimensional vectors and normalized to $[0, 1]$.

We compare our method against random selection (ensuring at least one sample per class). All experiments use 5 trials with different random seeds, reporting mean ± standard deviation.

Table 1. Test accuracy (%) for different prototype budgets M. "Ours" refers to variance-weighted centroid selection.

M	Compress.	Ours	Random	Δ
10000	6×	95.36±0.13	94.76±0.09	+0.60
5000	12×	94.58±0.21	93.63±0.09	+0.95
2500	24×	93.62±0.19	91.96±0.25	+1.66
1000	60×	92.47±0.17	88.56±0.51	+3.91
500	120×	91.40±0.33	84.70±0.28	+6.70
250	240×	89.71±0.34	80.27±0.36	+9.44
100	600×	86.19±0.40	72.39±0.80	+13.80
50	1200×	82.48±0.62	61.57±4.51	+20.91
25	2400×	77.11±0.91	49.76±5.80	+27.35
10	6000×	67.01±0.33	39.37±9.21	+27.64

4.2. Baseline: Full 1-NN

Using all 60,000 training samples, 1-NN achieves **96.91%** test accuracy. This serves as the upper bound for prototype selection methods.

4.3. Main Results

Table 1 shows classification accuracy across different compression ratios. Our method consistently outperforms random selection, with the advantage growing dramatically as compression increases.

Key observations:

- **Advantage scales with compression:** At low compression (6×), improvement is modest (+0.60%). At extreme compression (6000×), improvement is dramatic (+27.64%).
- **Stability:** Random selection's variance increases sharply at low M (std=9.21% at M=10), while our method remains stable (std=0.33%).
- **Graceful degradation:** At 60× compression, we lose only 4.44% accuracy compared to full 1-NN.

4.4. Negative Results: Failed Approaches

We also experimented with two alternative approaches that performed poorly:

Boundary-First Selection: We hypothesized that points near class boundaries are most important for defining decision surfaces. We identified boundary points (those with mixed-class k-NN neighborhoods) and performed K-Means only on these boundary points. Result: **56.77%** accuracy at M=500, worse than random (84.70%).

Condensed Nearest Neighbor (CNN): Inspired by the classic CNN algorithm (Hart, 1968), we iteratively added misclassified points to the prototype set. Result: **62.97%** accuracy at M=500, also worse than random.

Analysis: Both methods fail because they select atypical samples. Boundary points are often ambiguous or noisy examples. Misclassified points are, by definition, hard to classify. Neither makes a good prototype. This confirms our key insight: **good prototypes are typical samples, not boundary samples**.

5. Related Work

Prototype selection has been studied extensively. The Condensed Nearest Neighbor (CNN) rule (Hart, 1968) iteratively builds a consistent subset. Edited Nearest Neighbor (ENN) (Wilson, 1972) removes noisy samples. More recent work includes genetic algorithms (Cano et al., 2003), particle swarm optimization (Nanni & Lumini, 2009), and deep learning approaches (Snell et al., 2017).

Our variance-weighted approach is related to stratified sampling and importance sampling, but specifically designed for the geometry of k-NN classification.

6. Conclusion

We presented a simple yet effective prototype selection method for 1-NN classification. By allocating prototypes based on within-class variance and using K-Means to identify representative samples, we achieve significant improvements over random selection, especially at high compression ratios.

Our negative results with boundary-first and CNN-inspired approaches provide a valuable lesson: for prototype selection, **typical samples are better than boundary samples**. The best prototypes are cluster centroids that represent the "normal" appearance of each class, not edge cases.

Practical Recommendation: For applications requiring high accuracy, use $M \geq 1000$ (60× compression, 92.47% accuracy). For storage-constrained applications, even $M = 50$ (1200× compression) maintains 82.48% accuracy.

Acknowledgements

This project was developed with assistance from Claude (Anthropic), a large language model. The LLM was used as an interactive coding assistant for:

- **Algorithm design:** Brainstorming and iterating on prototype selection approaches, including the variance-weighted allocation scheme.
- **Implementation:** Writing Python code for data loading, prototype selection algorithms, experiment framework, and visualization.
- **Debugging:** Identifying performance bottlenecks (e.g., switching from standard K-Means to MiniBatchK-Means).
- **Documentation:** Maintaining progress logs and generating this report.

The human contributor provided problem formulation, high-level direction, critical evaluation of results, and all final decisions. The LLM served as a "pair programming" partner, accelerating implementation while the human maintained intellectual oversight.

This collaboration exemplifies a productive human-AI workflow: the human provides domain expertise and judgment, while the AI accelerates routine tasks and suggests alternatives. All experimental results were computed on actual hardware; no results were fabricated.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. The technique of prototype selection can reduce computational costs and storage requirements for nearest-neighbor classifiers, potentially enabling deployment on resource-constrained devices. We do not foresee specific negative societal consequences from this work.

References

Cano, J. R., Herrera, F., and Lozano, M. Using evolutionary algorithms as instance selection for data reduction in KDD: an experimental study. *IEEE Transactions on Evolutionary Computation*, 7(6):561–575, 2003.

Cover, T. and Hart, P. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

Hart, P. The condensed nearest neighbor rule. *IEEE Transactions on Information Theory*, 14(3):515–516, 1968.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Nanni, L. and Lumini, A. Particle swarm optimization for prototype reduction. In *Neurocomputing*, volume 72, pp. 1092–1097, 2009.

Snell, J., Swersky, K., and Zemel, R. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Wilson, D. L. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, (3):408–421, 1972.