Prototype Selection for Nearest Neighbor

Bella Jeong

Abstract

Nearest Neighbor (1-NN) classification has the disadvantage of large computational volume and slow speed because it stores and compares training data. To solve this problem, we will look into a method to reduce the computational volume and maintain high classification accuracy by selecting a prototype that is part of the training dataset. The prototype used in this study is described below.

1. Random Selection: M random samples are selected from the training data and used as a baseline.

2. Centroid-Based Selection: The center of each class is calculated and M samples close to the center are selected.

In this study, we compare the two methods and measure the accuracy at M=10000, M=5000, and M1000, and analyze them with confidence intervals. Finally, based on the experimental results, we suggest directions for further improvement of the prototype.

1 Prototype Selection

1.1 Random Selection

The random prototype selection method is a method of selecting M random data sets from the training data set. This has the advantage of not needing to know the data distribution in advance and being computationally simple. However, since it randomly collects samples, there are cases where data with important information is not selected. Although this method is simple, it has excellent classification performance under special conditions. For example, it performs well when the data distribution between classes is uniform or the representativeness of the data (representativeness of the sample) is not important. However, since it does not consider various patterns among the data within the class, there are frequent cases where boundary data is not selected. As a result, the function may be degraded in data sets where specific samples play an important role.

1.2 Centroid-Based Selection

Centroid-based selection, chosen by prototype selection, is a method that emphasizes the representativeness of each class. The center vector is obtained by averaging all samples of all classes. Then, M samples close to the center are selected.

1.2.1 Mathematical Definition

Let's say given class C_k's sample is :

$$X_k = \{x_{k1}, x_{k2}, \ldots, x_{kN_k}\}$$

Here, $x_{ki}$is kth class's ith sample and total N_k sample exists. The class middle point(centroid) $\mu_k$is calculated by the following:

$$\mu_k = \frac{1}{N_k}\sum_{i=1}^{N_k} x_{ki}$$

Now, calculate **Euclidean distance** for each sample $x_{ki}$and centroid μ_k and select most close Msample:

$$d(x_{ki}, \mu_k) = \|x_{ki} - \mu_k\|_2$$

Allign the samples according to the sample's distance and select the most closest M.

1.2.2 Limitation

This method has some limitation even though the sample centroid in the class shows representativeness. 1. Doesn't reflect class boundary data: Classification performance often determined at the decision boundary but this method is unlikely to reflect this edge cases. 2. When class distribution is asymmetric, the effectiveness can decrease: Some class can be distributed asymmetrically, but centroid may not represent the important data well.

Therefore, it is necessary to consider other method too such as: clustering-based selection to compensate these limitations.

1.3 Pseudocode

1.3.1 Random Prototype Selection

Algorithm 1 Random Prototype Selection

```
Require: Training set $(X_{\text{train}}, y_{\text{train}})$, Number of prototypes $M$
Ensure: Selected prototype set $(X_{\text{proto}}, y_{\text{proto}})$
1: Select $M$ random indices from range $[0, \text{len}(X_{\text{train}}))$ without replacement
2: Return the corresponding samples and labels
```

1.3.2 Centroid-Based Prototype Selection

Algorithm 2 Centroid-Based Prototype Selection

```
Require: Training set $(X_{train}, y_{train})$, Number of prototypes $M$
Ensure: Selected prototype set $(X_{proto}, y_{proto})$
1: Group samples by class labels
2: Compute the centroid $\mu_k$ for each class $C_k$
3: Determine number of samples per class: $M/$num of classes
4: for each class $k$ do
5:     Compute distance of each sample to the class centroid $\mu_k$
6:     Select the $M_k$ closest samples to the centroid
7: end for
8: Return the selected samples and labels
```

1.3.3 1-NN Classification

Algorithm 3 1-NN Classification

```
Require: Prototype set $(X_{proto}, y_{proto})$, Test set $X_{test}$
Ensure: Predicted labels $y_{pred}$
1: for each test sample $x$ in $X_{test}$ do
2:     Compute distance to all prototype samples
3:     Identify the nearest neighbor
4:     Assign its label as the predicted label
5: end for
6: Return predicted labels
```

These pseudocode provide step-by-step breakdown of the processes.

2 Experimental Results

2.1 Experimental Setup

We conducted the experiment using the MNIST dataset and applied a 1-NN classifier without using sklearn. We set the M value to 10000, 5000, and 1000, and compared the random selection value and the centroid selection value. To provide a confidence interval, we repeated the experiment several times and calculated the average accuracy and standard deviation.

2.2 Experiment Result Chart

M (Prototypes)	Random Selection Accuracy ± CI	Centroid Selection Accuracy ± CI
10000	0.9670 ± 0.0135	0.8450 ± 0.0000
5000	0.9300 ± 0.0190	0.8300 ± 0.0000
1000	0.8790 ± 0.0145	0.7700 ± 0.0000

Table 1: Accuracy comparison of Random Selection and Centroid Selection

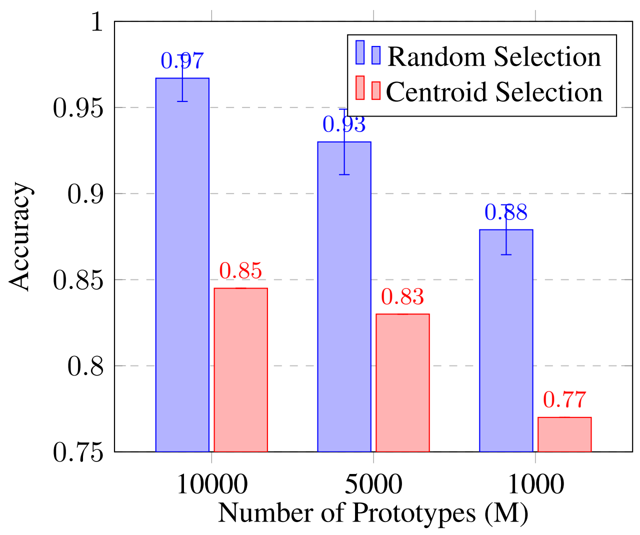

Figure 1: Comparison of Random vs. Centroid Selection Accuracy

2.3 Confidence Interval Calculation Formula

After measuring the accuracy of the data multiple times, the 95% confidence interval (CI) is calculated as follows:

$$CI = 1.96 \times \frac{\sigma}{\sqrt{n}}$$

where:

- σ is the standard deviation. - n is the number of repeated experiments.

In the case of **random selection**, the accuracy decreases **gradually** as M decreases. In contrast, **centroid selection** shows a **sharp decline** in accuracy. This suggests that centroid selection fails to sufficiently capture **decision boundary information**.

3 Critical Evaluation

3.1 Performance Comparison and Analysis

Random selection showed higher accuracy compared to centroid-based selection.

Since centroid-based selection only chooses samples that is near the class center, I think it failed to capture decision boundary samples, which gave lower accuracy than random selection.

In the case of $M = 1000$, centroid-based selection achieved 77% accuracy, which is significantly lower than random selection. This result shows that it failed to have the diversity of samples within a class.

3.2 Limitations and Possible Improvements

Need to Include Decision Boundary Samples Instead of only considering centroids, the selection process should include data points near class boundaries. To achieve above, techniques such as **K-Means clustering** or **SVM-based decision boundary sampling** can be considered.

Lack of Intra-Class Diversity Because this experiment didn't fully reflect various patterns within a class, it may fail to represent certain categories accurately. So hybrid approach including high-variance samples with centroid-based selection can be considered in the future.

Possible Integration of KNN-Based Prototype Selection To optimize the selected prototypes for KNN classification, the data selection needs improvement.

3.3 Future Research Directions

- Experimenting with **Boundary-Based Prototype Selection** techniques.

- Exploring and comparing existing prototype selection methods such as **Condensed Nearest Neighbor (CNN)** and **Edited Nearest Neighbor**.

- Investigating a **hybrid approach** that combines existing methods.

Even though the selection that I chose wasn't exceeding random selection accuracy, in this experiment I learned how to solve this critical error.