CSE251A Programming Project 1 — Prototype Selection for Nearest Neighbor

Thaddaeus Sandidge [1]

Abstract

This project submission describes the implementation of Mini-Batch K-Means as described by Sculley in the paper *Web-Scale K-Means Clustering*. Implementing this strategy for prototype selection was found to show significant improvement over random sampling with minimal computation time (Sculley, 2010).

1. High Level Description

The Mini-Batch K-Means algorithm is a variation of the classic K-Means algorithm (Lloyd, 1982; MacQueen, 1967) designed to be faster and more scalable. The K-Means algorithm groups data into k clusters by minimizing the distance between points and their assigned cluster centers. It does this by repeatedly assigning points to the nearest centroid and updating those centroids as averages. These cluster centers will serve as representative prototypes that summarize groups of similar samples. Mini-Batch K-Means is a faster, scalable version of K-Means that updates cluster centers using small random subsets (mini-batches) of the data instead of the full dataset each iteration.

2. Algorithm Pseudocode

To efficiently classify all 10 possible digits, we first take in the train images and labels and our prototype limit M. We then divide M by 10 to obtain our k value. k will be the number of cluster centers per digit class and the same among all digits to ensure equal representation among each class. Then we filter the train images by digit, and run Mini Batch K-Means on each digit, producing k cluster centers per digit which will be joined to create our final prototype subset, M_s.

Our Mini Batch K-Means algorithm takes as input a dataset X of images of identical digit class, k number of clusters, a batch size B, and a max iterations value T. The algorithm first randomly selects k points from X to serve as initial cluster centers. For each cluster center, we initialize a count of 1.

Then we perform T iterations. For each iteration, we randomly sample a mini batch B form X. For each point x in the mini-batch, we compute the distance d from each cluster center and assign x to the nearest cluster center c_j. Then for each assigned point, $x \to c_j$, we increment the count of that cluster center, compute the learning rate as $\eta = 1/count_j$ and update the cluster center using an incremental mean update as $c_j \leftarrow (1-\eta)\, c_j + \eta\, x$. After completing T iterations, we output the k cluster centers.

The k cluster centers of each digit are combined to return our final prototype selection M_s.

Algorithm 1 Class-Wise Mini-Batch K-Means Prototype Selection

```
Input: Training images $X$, labels $Y$, prototype limit $M$, batch size $B$, iterations $T$
Output: Prototype set $P$, prototype labels $L$
$k \leftarrow M/10$ {prototypes per digit class}
Initialize empty prototype list $P$
Initialize empty label list $L$
for each digit class $d \in \{0, \dots, 9\}$ do
    $X_d \leftarrow \{x \in X \mid Y(x) = d\}$ {filter class data}
    Randomly sample $k$ points from $X_d$ as initial centers $\{c_1, \dots, c_k\}$
    Initialize counts: $count_j \leftarrow 1$ for all $j \in \{1, \dots, k\}$
    for $t = 1$ to $T$ do
        Sample mini-batch $B_t \subset X_d$ with $|B_t| = B$
        for each $x \in B_t$ do
            Assign $x$ to nearest center $c_j$
            $count_j \leftarrow count_j + 1$
            $\eta \leftarrow 1/count_j$
            $c_j \leftarrow (1-\eta)\, c_j + \eta\, x$
        end for
    end for
    Add $\{c_1, \dots, c_k\}$ to $P$
    Add label $d$ repeated $k$ times to $L$
end for
$P, L$
```

[*]Equal contribution [1]. Correspondence to: Thaddaeus Sandidge <>.

3. Experimental Results

All recommended values of M (1000, 5000, 10,000) were tested in three trials per M value. The Mini-Batch K-Means algorithm saw significant improvement over random selection, particularly for lower values of M.

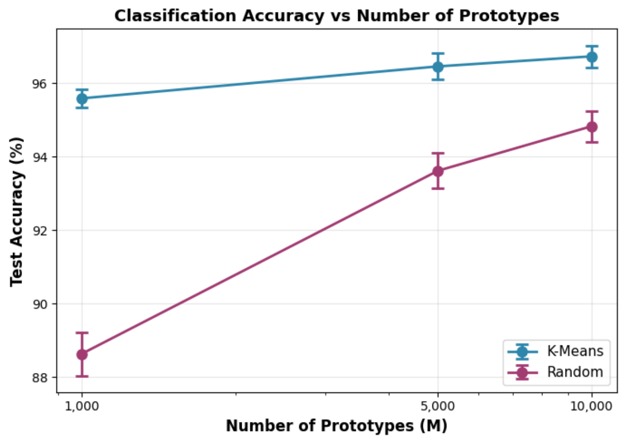


Figure 1. The classification accuracy of Mini-Batch K-Means and random prototype selection by the number of prototypes (M)

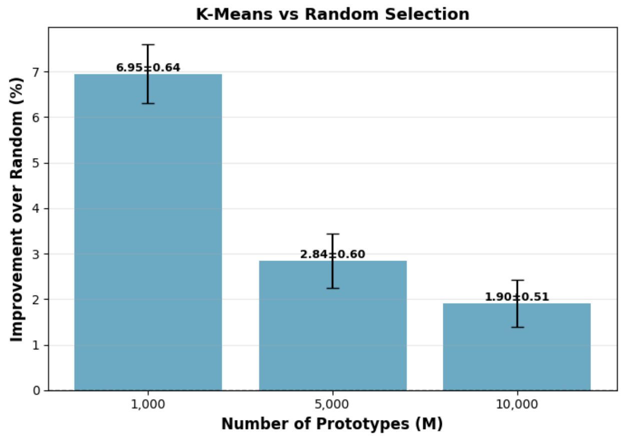
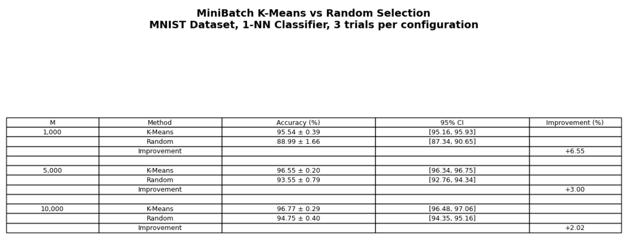


Figure 2. The improvement over random selection of Mini-Batch K-Means for each tested value of M

The figures visualize that the effectiveness of the Mini-Batch K-Means algorithm compared to random selection decreases as the size of M increases. The confidence intervals in the trials were also calculated.

4. Critical Evaluation

Overall, Mini-Batch K-Means appears to be an excellent strategy for prototype selection for the MNIST dataset and



MiniBatch K-Means vs Random Selection
MNIST Dataset, 1-NN Classifier, 3 trials per configuration

M	Method	Accuracy (%)	95% CI	Improvement (%)
1,000	K-Means	95.54 ± 0.39	[95.16, 95.93]	
	Random	88.99 ± 1.66	[87.34, 90.65]	
	Improvement			+6.55
5,000	K-Means	96.55 ± 0.20	[96.34, 96.75]	
	Random	93.55 ± 0.79	[92.76, 94.34]	
	Improvement			+3.00
10,000	K-Means	96.77 ± 0.29	[96.48, 97.06]	
	Random	94.75 ± 0.40	[94.35, 95.16]	
	Improvement			+2.02

Figure 3. Table showing confidence intervals for the experiments for each value of M

significantly outperforms random selection. There may be possibilities for improvement in investigating strategies to improve classification for higher values of M, as the improvement over random diminished as M increased.

In the future, I would like to compare the classification accuracy of Mini-Batch K-Means to the traditional K-Means strategy to investigate the tradeoffs in classification accuracy versus computation time. Lastly, in this approach the cluster centers serve as representative prototypes, and thus are not a real data point in the training set but rather low-variance representative approximation of the local data distribution. This proved to be an effective strategy for this dataset, but I would also like to compare the effectiveness of this approach with an algorithm that explicitly selects actual train data points and evaluate both the computation time and classification accuracy compared to the mean-based Mini-Batch K-Means approach.

References

Lloyd, S. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

MacQueen, J. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pp. 281–297. University of California Press, 1967.

Sculley, D. Web-scale k-means clustering. In *Proceedings of the 19th International Conference on World Wide Web (WWW)*, pp. 1177–1178. ACM, 2010.