Prototype selection for nearest neighbor using K-Means on dimension reduced data

Prasad Madhava Kamath
pkamath@ucsd.edu

Abstract

In this paper we describe a method for prototype selection for a 1- Nearest Neighbor classifier based on a K-Means clustering scheme on a dimensionality reduced feature set. The curse of dimensionality is a well know phenomenon which states that the amount of data need to maintain density increases as the number of dimensions increase. Since we aim to select a small set of prototypes for the nearest neighbor classification on high dimensional dataset, we propose a UMAP based dimensionality reduction scheme, followed by K-Means clustering for identifying prototype points as an effective prototype selection strategy. Using this scheme we were able to achieve an accuracy of 96.6% using 1000 prototype points from the MNIST dataset for the 1-Nearest Neighbor classification task.

1 Introduction

Nearest Neighbor is a simple non-parametric learning algorithm that can be used to classify data. There is no explicit training step involved in this learning scheme, the classifier simply takes a sample data point from the test set and finds its distance with respect each data point in the training set and assigns the label of the the training sample which has the smallest distance, thus the nearest neighbor. We can thus see, given a large corpus of training data the Nearest Neighbor algorithm would take considerable time to scan through the data set and find the nearest neighbor. This could be remedied by choosing a smaller corpora of data called prototypes that is representative of data in the training set.

The challenge with selection of such prototype points lies in the nature or dimensionality of data.The nearest neighbor algorithm find the nearest neighbor by comparing distances between two samples and requires that this distance be minimum in every single dimension. However, it can be inferred from the curse of dimensionality that as the dimensionality increases we require more data to ensure a dense distribution.With increasing dimension and sparse data the nearsest neighbor the ratio between the closest and average distances between data points tends to one, making it a difficult task. Therefore in the following section we propose a prototype selection scheme which mitigates the curse of dimensionality using a dimensinality reduction techniques followed by a K means based prototype selection.

2 Methodology

1. *Uniform Manifold Approximation and Projection (UMAP) :*
   UMAP is a dimensionality reduction algorithm which uses manifold learning and ideas from topological data analysis to obtain low dimensional data representations (McInnes et al., 2020). We choose UMAP over other approaches such as t-SNE, as UMAP has faster runtimes and tends to preserve more of the global structure in data[https://pair-code.github.io/understanding-umap/].The UMAP model is trained on the training data and fit to obtain a low dimensional representation of the training data. Here each 784 dimension input sample (28x28 image from MNIST training set) is transformed into a 32 dimension input sample vector using UMAP with the cosine distance metric. The trained UMAP model is then used to transform the test data from the MNIST dataset to 32 dimensional vectors.

2. *K-Means Clustering :*
   The K-Means clustering scheme uses the low dimension training data to cluster data into 'K' clusters. We then consider 'N' the points

around the centroid of each cluster to be the prototype points representative of each cluster, such that the total number of prototypes needed is M= K x N. The closeness from the centroid is determined using the euclidean distance between points.These M points are used as prototypes for the Nearest Neighbor algorithm (Ougiaroglou and Evangelidis, 2012).

Pseudo code:

Learning

1. *Train the UMAP model on the MNIST training data and obtain the low dimensional embedding of the same*
2. *Using UMAP model fit on training data obtain the low dimension embedding of the MNIST test data*
3. *Use K-Means clustering algorithm to cluster the embedded training data into K clusters*
4. *To pick prototype points consider (M/K) points closest to cluster centroid of each cluster, where M is the total number of prototype points - [1000, 5000, 10000]*

Inference

5. *Use these M prototypes to perform 1- Nearest Neighbor classification using the embedded MNIST test data*

3 Experimental results

As a baseline reference for accuracy at various values of prototype points 'M', we choose a scheme in which we use the random module to generate random indices in the range of the MNIST test data size and use these indices to pick M random prototype points. Using the 1-Nearest Neighbor algorithm from sklearn (Pedregosa et al., 2011) we then compute the accuracy metric over the MNIST test dataset. Here we operate on the original data which consists of 28x28 size images flattened into a 784 dimension vector.
The results of 1-Nearest Neighbor- accuracy at various values of M - 1000,5000,10000 are plotted on a graph as shown in Figure 1.

3.1 Experiment1 : Prototype selection using K-Means

We first propose to use the K-Means clustering algorithm from sklearn (Pedregosa et al., 2011) to obtain the cluster centers and then pick prototype points close to the centroid of each cluster. For this we need to first pick the number of clusters that we need. There are a few standard metrics to find the optimal value of clusters - Inertia. We know that as we increase the number of clusters the inertia would decrease, however we need to use the elbow method to find the optimal 'K' cluster value, since beyond this point the decrease in inertia would become asymptotic. Figure 2 shows the plot of Inertia vs Cluster Size. We can see that the optimal value is obtained closed to 100, so we choose the number of clusters as 100 for our K-Means algorithm. Once we fix on the number of clusters we follow the below procedure:

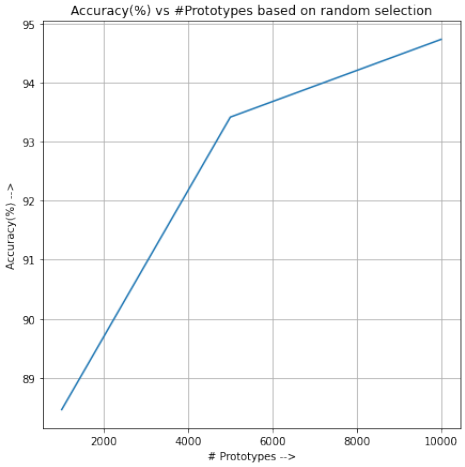

Figure 1: Accuracy vs Number of Prototypes plot for 1-Nearest Neighbor classifier for the MNSIT dataset

1. Train the K-Means model on MNIST training data using the optimal cluster value
2. Find the centers of each cluster
3. Find (M/K) number of points in each cluster that is closest to the center using euclidean distance
4. use these M points as prototypes for 1-NN classification and compute the accuracy for different values of M= [1000,5000,10000]

We now measure the accuracy of the 1-NN classifier with prototypes selected based on k-means clustering for M= [1000,5000,10000] Figure 3 shows the plots of accuracy of this approach vs the classification accuracy using random prototype selection.

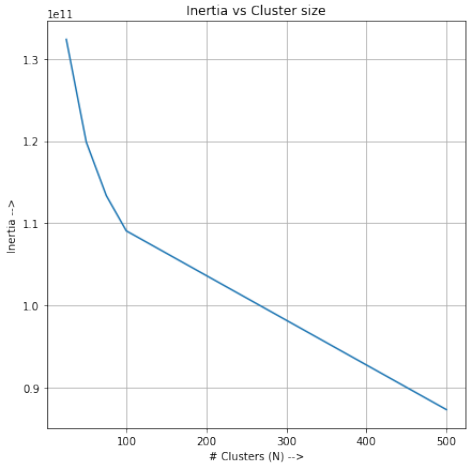

Figure 2: Inertia vs Number of clusters for K-Means algorithm

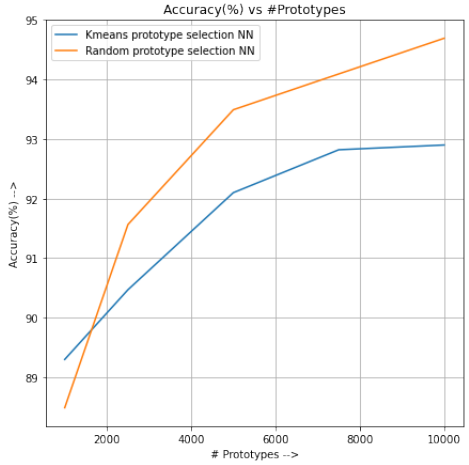

Figure 3: Accuracy vs Prototype points, a comparison between K-Means based prototype selction and random prototype selection

3.2 Experiment2 : Prototype selection using UMAP and K-Means

We see in experiment 1, K-means tends to be incrementally worse in the accuracy of the 1-NN classifier on the MNIST dataset. We can infer that due to the high dimensionlity and sparse dataset, the euclidean distance measure used suffers from the curse of dimensionality. With this in mind we try reducing the dimensionality of each data point from 784 to a smaller dimensional vector ensuring good data density. From experiments we find that the cosine distance metric is a better measure compared to the euclidean measure when trying to reduce the dimensionality using UMAP. Once the dimensionality is reduced we could use the euclidean distance measure since UMAP tends to project the data points into a euclidean space.

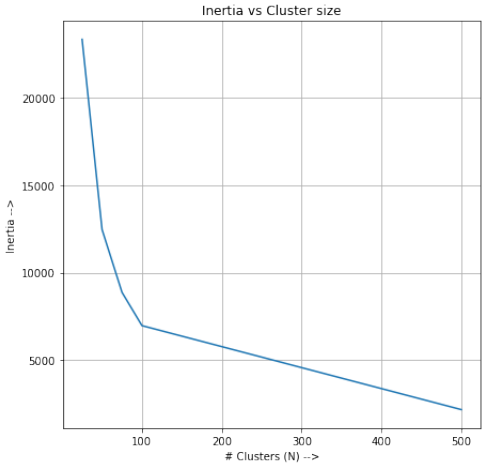

Figure 4: Inertia vs Number of clusters for K-Means algorithm on UMAP embedded data

In order to do this we need to determine the optimal value of components in the UMAP model. To do this we select different component values and generate several low dimensional embeddings.For each train data embedding we use the K-means clustering algorithm with different number of clusters and find the optimal (component, cluster number) pair for which we see an elbow in the inertia curve. From this experiment we see the optimal value of components = 16 and number of clusters=100. The inertia vs number of cluster is shown in Figure 4
Now that we have obtained the parameters of our system we need to train the UMAP model with the given number of components and obtain the train and test data embedding in lower dimensional space. the dimensionality is reduced from 784 dimension sample vector to a 16 dimension sample vector on the MNIST data set. We now have 60000 such sample points in our train set. To choose 'M' points we follow the procedure described in experiment 1 with K-Means, using 100 clusters.
The accuracy vs prototype number plot for this proposed scheme is compared with the random point selection as shown in Figure 5. Thus it can be inferred that the Kmeans prototype selection on a reduced dimensionality dataset performs better than other methods described in this paper and provides nearly the same accuracy as the 1-NN classifier operating on a training corpus of 60000 MNIST train data points. The accuracy of different schemes is compared for different number of prototypes and tablulated in table 1

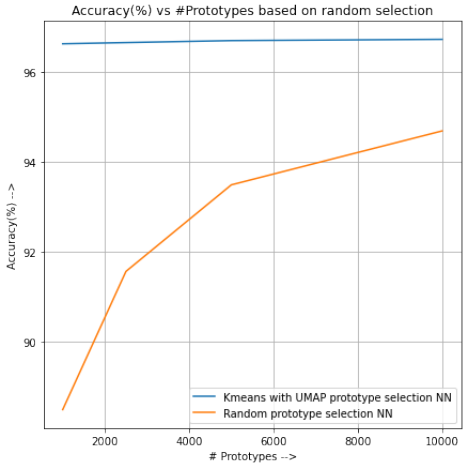

Figure 5: Accuracy vs Prototype points, a comparison between K-Means based prototype selection using UMAP projected data and random prototype selection

Algorithm	1000 prototypes		5000 prototypes		10000 prototypes	
	Mean(%)	STD	Mean(%)	STD	Mean(%)	STD
Random selection	88.65	0.33	93.42	0.19	94.71	0.18
K-Means on 784 dimensional data	89.2	0.37	92.015	0.108	93.17	0.19
K-Means on UMAP mapped 16 dimensional data	96.63	0.097	96.70	0.04	96.73	0.011

Table 1: A comparison of accuracy on the MNIST test data set for various schemes

4 Conclusion

From the experiments conducted it is evident that K-Means based prototype selection on a lower dimensional dataset yields much better results than random selection of prototype points. The curse of dimensionality is mitigated by using UMAP to map the 784 dimension data sample into a 16 dimension (using 30 neighbors) representation which packs data densly enough for K-Means to operate well and be able to cluster points around a centroid allowing better separation of points. Thus, when we use K-Means with 100 clusters over the 16 dimensional UMAP reduced data we are able to achieve an accuracy of 96.6% with just 1000 prototype points, which is comparable to the performance of 1-NN when trained on a dataset of 60000 samples. The proposed scheme clearly beats the random prototype selection scheme for every value of prototype point.

References

Leland McInnes, John Healy, and James Melville. 2020. Umap: Uniform manifold approximation and projection for dimension reduction.

Stefanos Ougiaroglou and Georgios Evangelidis. 2012. Fast and accurate k-nearest neighbor classification using prototype selection by clustering. In *2012 16th Panhellenic Conference on Informatics*, pages 168–173.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.