Effects of Data Division Methods on Neural Network Performance

Jimmy Woo (100823301)
April 11, 2020

Abstract

It is widely used in practice to subset the dataset randomly using predetermined ratios when splitting the dataset into training and testing sets. However, This may lead to issues, having a training set that does not optimally represent the testing set. In this report, various data division methods were explored, applying clustering algorithms in preprocessing to generate a training set that better represents the testing set than a traditional random split. It was shown that the training sets identified from these clustering algorithms helped reduce issues related to overfitting.

1 Introduction

According to the universal approximation theorem, a feed forward neural network with a single hidden layer of finite number of nodes is capable of generating approximations for any limited functions [1]. With all of the publicly available computational resources and datasets, ANNs have been becoming increasingly more popular in recent years. Often when using ANNs, cross-validation [2] is used, iteratively dividing the dataset into training and testing sets to avoid overfitting. This technique allows maximal reuse of the available data in the dataset. The current norm of subsetting involves randomly splitting the dataset into training and testing sets into predetermined proportions (e.g. 70/30 split). However, it has been shown that random partitioning may cause sample representation issues, which may result in assessing the model performances with lacking relevant examples during the training phase [4]. Therefore, the method used for division of data can have a significant impact on the performance of an ANN model.

While random division of datasets are widely used for machine learning models, it has been previously shown that the performance of an ANN model can be improved if the statistical properties of the data subsets are similar [3]. Shahin *et al.* (2000) showed that datasets can be divided into subsets for an ANN model by using self-organizing maps (SOM). In this report, the division of datasets using various types of clustering algorithms, including centroid-based, density-based, and graph-based approaches will be explored.

In section 2, the necessary background information including the overview of the clustering algorithms used, will be covered. In section 3, the methodology used to obtained the results of this report will be explained in detail. In section 4, the obtained results will be discussed.

2 Background

2.1 Related Work

Genc *et al.* (2019) proposed a histogram matching approach, where the training and testing sets were partitioned optimally to best fit the distributions of the original dataset, and has shown the application on a purely categorical dataset [5]. Shahin *et al.* (2000) proposed an approach where a topological map representation of a dataset is created using SOM, and each of the clusters formed in the map were evenly distributed for training, testing and validation sets. In this report, the application of splitting the dataset by clusters will be extended to utilize various clustering algorithms, include K-Means [6], Density-Based Spatial Clustering Applications with Noise (DBSCAN) [7], and a graph based clustering algorithm that optimizes based on modularity scores of the clusters [8].

2.2 Clustering Algorithms

K-Means

K-Means clustering algorithm iteratively attempts to partition the given dataset into K predefined clusters, where each observation can only belong to one cluster. The algorithm involves minimizing the pairwise sum of squared deviation from a data point to its centroid. The objective can be expressed as:

$$\underset{S}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|$$

where S is the set containing the entire dataset, S_i are the clusters within S, μ_i are the cluster centroids, and x represent the observations in the dataset. The algorithm randomly assigns K observations as initial centroids, assigns each observation in the dataset to the cluster with the closest centroid, and recalculates the updated centroids of the clusters. The algorithm converges when cluster assignments are no longer changed.

DBSCAN

DBSCAN is a density-based clustering algorithms, meaning that clusters are formed at dense regions in the data space. DBSCAN requires two parameters, ϵ, defining the radius of a cluster with respect to some observation, and *minPts*, the minimum number of points to form a cluster. The algorithm is executed as follows:

1. Identify an arbitraty observation that has not yet been visited.
2. Identify the ϵ-neighbourhood of this observation (i.e. all points within the ϵ radius in the data space). If the size of this ϵ-neighbourhood is $\geq$ *minPts*, a cluster is formed. Otherwise, label this observation as noise.
3. If the observation is found within a cluster, add this observation and its ϵ-neighbourhood to the cluster.
4. Repeat the above steps until all observations are processed as either clusters or noise.

Graph-Based

The particular graph-based clustering algorithm of interest (Clauset *et al.* (2004) finds the optimal clusters of a graph by optimizing the modularity scores of each subgraph forming the clusters. Modularity of is a property of a network that measures the strength of the divisions [9], where networks with high modularity have densely connected modules, and have sparse inter-module connections. This algorithm performs greedy optimization of modularity score of the graph by merging observations into clusters. The algorithm converges when no further increase in the modularity score can be obtained. Refer to [8] for details.

2.3 Dataset

The *Airfoil Self-Noise* Dataset contains the results of wind tunnel tests on various NACA 0012 airfoils with varying test parameters. These paremeters include frequency (*Hz*), angle of attack (°), chord length (*m*), free-stream velocity (*m/s* [10], and suction side displacement thickness (*m*). For each observation the scaled sound pressure level (*dB*) was recorded.

3 Methodology

The performances of a Multilayer Perceptron (MLP) model was be observed for each of the data division methods. For the random split method, the performance was observed using testing set proportion as 20 %. For the data division methods involving clustering, a grid search was performed to find the optimal hyperparameters. These optimal hyperparameters were used to cluster the dataset, and every cluster was divided evenly between the training and testing sets. The performances of the MLP model using these various splits were observed. More specifically, the Mean Sqaured Error (MSE), which is the sum of squared deviations between observed and predicted values, was used as the performance metric. This can be expressed as:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} \left(Y_i - \hat{Y}_i \right)^2$$

where *n* is the total number of observations, Y_i is the *ith* observed value, and $\hat{Y}_i$ is the *ith* predicted value.

Since the dataset is not graph-based in nature, it was required to generate an approximation for the graph representation of the dataset. The followings steps were taken to generate a graph represnetation of the dataset suitable for the graph-based clustering algorithm.

1. Treat each observation as a vertex. Apply the K-Nearest Neighbours algorithm [11] on the entire dataset, and define edge connections between all vertices in each neighbourhood.
2. generate graph based on above definitions, and execute graph-based clustering algorithm with generated graph.

The training error and the testing error were observed for all epochs during the training phase for all above mentioned methods.

4 Results

4.1 Network Architecture

The network has 5 nodes in the input layer, and 1 node in the output layer. The model uses ReLU activation everywhere, with a learning rate of 0.0025 and a decay factor of 0.95 using RMSprop optimizer The effect of the number of hidden layers and the number of nodes in each hidden layer on the performance of the network was observed using the random split method, refer to Table 1 for the results.

	Train MSE after training	Test MSE after training
1 Hidden layer, 5 nodes	40.1	44.6
1 Hidden layer, 10 nodes	36.5	42.1
1 Hidden layer, 15 nodes	36.8	42.9
2 Hidden layers, 10 - 10 nodes	33.9	46.3
2 Hidden layers, 10 - 15 nodes	31.2	45.4

Table 1: MSE After Training for various architectures

It was evident that of the explored options, the architecture with 1 hidden layer containing 10 nodes provided the optimal performance, and thus was selected to be used.

4.2 Optimal parameters for Clustering

Grid searches were performed individually for each data division method to identify the optimal set of parameters. Refer to table 2 for the optimal configuration of each technique.

	Parameter 1	Parameter 2
K Means	K = 5	N.A.
DBSCAN	ϵ = 0.05	minPts = 50
Graph-Based	K = 50 (for K-NN)	N.A

Table 2: Optimal parameters of each techinique

The above obtained parameters were used to observe and compare the model performances of each of the data division techniques.

4.3 Model Performances

Each of the data division techniques were applied to split the training and testing sets, and the model was trained using the provided training set for each method, observing the training and testing set MSE while training for 100 epochs. Refer to figure 1 for the model performances observed.

It is evident that the model trained on the randomly split data converged to the lowest training set MSE, but ended up with the highest testing set MSE, having an issue of overfitting compared to other models. Refer to figure 2 for

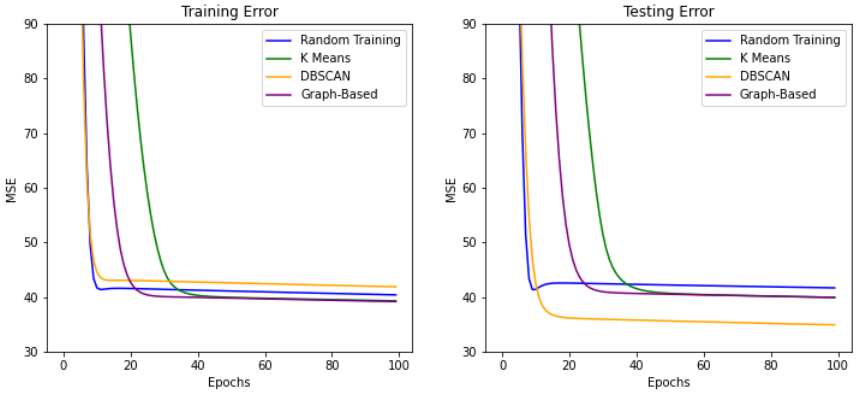

Figure 1: MSE for varying data division techniques

the breakdown of the observations used for the randomly split data in terms of memberships of each of the clustered training sets.The original breakdown of each data division method and the breakdown of the random split method in terms of clusters from other methods can be compared to see whether there is a sample representative issue present. It is evident that when comparing the K Means clustered observations to what K Means clusters the random split training observations would have been assigned to that observations of cluster 3 are lacking for the randomly split training set. Similarly, when comparing the randomly split training set with the DBSCAN clustered observations, it is evident that the randomly split training set lacks what DBSCAN identifies as noise. This may have a big impact on the performance of the network, since DBSCAN tends to label outliers as noise, meaning that the observations that are harder to predict were not present represented properly in the training set. The model trained on training set clustered by DBSCAN performed the best on this dataset.

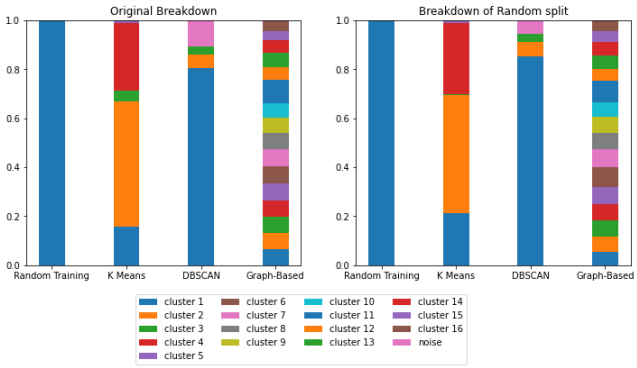

Figure 2: (left) Original breakdown of clusters for each data division method. (right) Breakdown of randomly split training set in terms of other data division methods

5 Conclusions and Future Work

Various clustering algorithms were applied in preprocessing to split the dataset and identify a training set that best represents the testing set. It was shown that these data division methods improved the model performance compared to a traditional random split. Since all of the clustering techniques used (or the preprocessing step to the clustering algorithm) in this report relies on computing euclidean distances in the data space, it may be beneficial to test these techniques with a high dimensional dataset in the future.

References

[1] Cybenko, G. (1989) "Approximations by superpositions of sigmoidal functions", Mathematics of Control, Signals, and Systems, 2(4), 303–314. doi:10.1007/BF02551274

[2] Kohavi, Ron (1995). "A study of cross-validation and bootstrap for accuracy estimation and model selection". Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence. San Mateo, CA:

[3] Shahin, M.A., Maier H.R., Jaksa M.B. (2000) "Evolutionary data division methods for developing artificial nerural network models in geotechnical engineering". Department of Civil & Environmental Engineering, The University of Adelaide, Research Report No. R 171

[4] Liu, H., Cocea, M. "Semi-random partitioning of data into training and test sets in granular computing context". Granul. Comput. 2, 357–386 (2017). https://doi.org/10.1007/s41066-017-0049-2

[5] Genc, B., Tunc H. "Optimal training and test sets design for machine learning". Turkish Journal of Electrical Engineering & Computer Sciences, 1534-1545 (2019)

[6] Lloyd, Stuart P. "Least squares quantization in PCM." Information Theory, IEEE Transactions on 28.2 (1982): 129-137.

[7] Ester, Martin; Kriegel, Hans-Peter; Sander, Jörg; Xu, Xiaowei (1996). Simoudis, Evangelos; Han, Jiawei; Fayyad, Usama M. (eds.). A density-based algorithm for discovering clusters in large spatial databases with noise. Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96). AAAI Press. pp. 226–231.

[8] A Clauset, MEJ Newman and C Moore: Finding community structure in very large networks. Phys Rev E 70, 066111 (2004).

[9] M. E. J. Newman and M. Girvan, Finding and evaluating community structure in networks. Phys. Rev. E 69, 026113 (2004).

[10] T.F. Brooks, D.S. Pope, A.M. Marcolini, Airfoil Self-Noise and Prediction, NASA RP-1218, July 1989

[11] Altman, Naomi S. (1992). "An introduction to kernel and nearest-neighbor nonparametric regression" (PDF). The American Statistician. 46 (3): 175–185. doi:10.1080/00031305.1992.10475879. hdl:1813/31637.