Article review: A Multi-ELM Model for Incomplete Data

Matias Etcheverry
Ecole Normale Supérieure, France
matias.etcheverry9@gmail.com

Oumniya Ramdi
Ecole Normale Supérieure, France
ramdioumnia@gmail.com

1 INTRODUCTION

Missing or incomplete values in datasets are quite common when dealing with real-world machine learning tasks and are usually attributed to human error during preprocessing causing the machine learning models results to be biased and less accurate.

The work proposed by Chi et al. [1] considers a new approach for dealing with missing values in datasets. This approach is based on the deployment of a new machine learning algorithm referred to as Extreme Learning Machines (ELM) over several commonly used strategies, including discarding missing values and using traditional imputation methods [4, 5]. These methods can both be effective when the amount of missing data is small. However, when this amount is significant, they become impractical. The proposed approach based on the Extreme Learning Machine (ELM) has been shown to be more effective when the proportion of missing values is large and can be directly applied without any further imputations hat could introduce additional noise and inaccuracy to the dataset [7, 9].

In this article, we review this approach as follow: first, we outline the related works in brief, second, we provide details about the proposed model of Extreme Learning Machines (ELM). In section 4, we describe the methodology used, we also look into the mathematical formulation of the model. Next, in section 5, we present the results to show the performance of this approach. Finally, concluding remarks and critiques are provided in Section 6 and 7.

2 RELATED WORKS

Missing data is a common problem encountered in many different application fields of science and has received considerable attention in the last few years. One of the easiest and most popular methods that have been deployed to deal with missing values is the mean imputation [8]. This method aims at replacing the missing value on a certain variable by the mean of the available values. It is shown to be effective but only when a few samples have missing values.

The K Nearest Neighbors (KNN) is another widely used method for imputation, which impute the missing values based on the k nearest neighbors found from the closest known values [6]. This method has also proven to be effective but not in the case where multiple components are missing for one data sample.

In order to avoid the errors introduced by imputations, a third approach suggests applying the machine learning models directly to the incomplete data. However, only a few of them are directly applicable to the dataset. Unlike the two methods we previously discussed, the method we are reviewing in this paper proposes the use of a different type of ML models based on Extreme Learning Machines. It can be applied directly to the dataset and has proven to be effective, especially when there are a lot of missing or incomplete values.

3 EXTREME LEARNING MACHINES

The algorithm relies on multiple Extreme Learning Machines (ELM). An ELM is a single feed forward network with a fixed first layer. Thus, it is lightweight, fast to train and able to generalize well [7, 9]. The single layer of L hidden neurons maps the input to a higher dimensional space using a matrix product and an activation function. That is to say the activation value of the i-th neuron for the input x is $h_i(x) = \phi(x^T w_i + b_i)$. The prediction is the weighted sum of all the hidden neurons: $\hat{y}(x) = \sum_{i=0}^{L} \beta_i h_i(x)$, as depicted by Fig. 1.

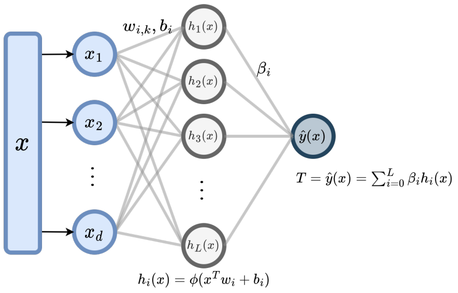

Figure 1: Extreme Learning machines with L hidden neurons. The output is a scalar here.

Supposing we have N input data composed of features and targets denoted as x_i and y_i. We use the matrix form H and β:

$$H = \begin{bmatrix} h_1(x_1) & \cdots & h_L(x_1) \\ \vdots & & \vdots \\ h_1(x_N) & \cdots & h_L(x_N) \end{bmatrix} \quad \text{and} \quad \beta = [\beta_1, \cdots, \beta_L]^T \tag{1}$$

As the hidden weights $w_{i,k}$ and b_i are fixed, the ELM aims at solving an ordinary square problem with unknown β as follow:

$$\beta^* = arg \min_{\beta} \sum_{i=1}^{N} (\hat{y}_i(x) - y_i(x))^2 = arg \min_{\beta} ||H\beta - Y||_2^2 \tag{2}$$

This minimization problem can be solved using the pseudo-inverse of $H^T H$.

4 METHODOLOGY

Supposing we have a dataset $\mathcal{X}$ of N samples composed of d features. We want to predict the set of targets $\mathcal{Y}$. A primary EML is used to predict the targets. However, as the dataset is made of samples having incomplete data, we can't compute the exact elements of H in Eq. (1). To do so, we are going to approximate the unknown values of H using multiple secondary ELMs.

4.1 Decomposition

We decompose the dataset $\mathcal{X}$ in two sub-datasets: $\mathcal{X}_C$ contains all the complete samples, ie samples having all their features known and $\mathcal{X}_M$ contains samples having mssing features.

Moreover, we can decompose a single sample $x \in \mathcal{X}$ in 2: $x = x^c + x^m$, where x^c is the set of complete features of x and x^m is the set of missing features of x. For instance, Fig. 2 introduces 5 different samples $x_1, \cdots, x_5$. x_2 can be decomposed as $x_2 = x_2^c + x_2^m$ where $x_2^c = [x_{2,1}, x_{2,3}]$ and $x_2^m = [x_{2,2}, x_{2,4}]$. Similarly, we can decompose the weights $w_{i,k}$ and then the matrix H, using $h(x) = h^c(x^c) + h^m(x^m)$[1]:

$$
\begin{aligned}
H &= H^c + H^m \\
\text{where} \quad H^c &= \begin{bmatrix} h_1^c(x_1^c) & \cdots & h_L^c(x_1^c) \\ \vdots & & \vdots \\ h_1^c(x_N^c) & \cdots & h_L^c(x_N^c) \end{bmatrix} \\
H^m &= \begin{bmatrix} h_1^m(x_1^m) & \cdots & h_L^m(x_1^m) \\ \vdots & & \vdots \\ h_1^m(x_N^m) & \cdots & h_L^m(x_N^m) \end{bmatrix}
\end{aligned} \tag{3}
$$

Intuitively, H^c is the matrix of activated neurons on known data, while H^m is the matrix of activated neurons on unknown data. Thus, H^m is an unknown matrix we would like to approximate.

4.2 Secondary ELMs

To approximate every element of the unknown matrix H^m, we create as many ELM as they are patterns of missing data. A pattern of missing data of a sample x denotes the set of missing features which should be predicted from known features. There are $2^d - 1$ maximum secondary ELMs. For a single EML $\mathfrak{E}$ used to predict an element of H^m, we use the whole dataset $\mathcal{X}_c$ as training dataset: the inputs are the features accessible by the EML while the targets are the missing features. Fig. 2 represents the training for a single EML. The dataset $\mathcal{X}$ is made of 5 samples. In this case, we want to predict the first activation neuron h_1^m on the samples x_3 and x_4 which have missing data. We can generalize this method so that the EML predict the whole vector of hidden neurons h^m on the samples x_3 and x_4.

With this algorithm, we can compute the whole matrix H^m for every pattern of missing data. Finally, we can make predictions on the target value y, using the primary ELM, applied to $H^c + H^m$.

5 EXPERIMENTS AND RESULTS

The authors tested their algorithms on 2 datasets. The first dataset called *Abalone* [3] is made of 2784 samples of 7 features each. The second dataset is a wine dataset [2] composed of 4898 samples and 11 features. The final measurement of the performances is done with the Mean Squared Error (MSE), against the K-nearest neighbor method. It is important to test the 2 algorithms with different ratios of missing values. Each dataset is split into training and validation set, and they are normalized using the training data. They ran a few experiments using different number of neurons in the primary EML (between 27 and 36 neurons) and the secondary EML (between 214 and 443 neurons). The validation MSE are listed in Table 1.

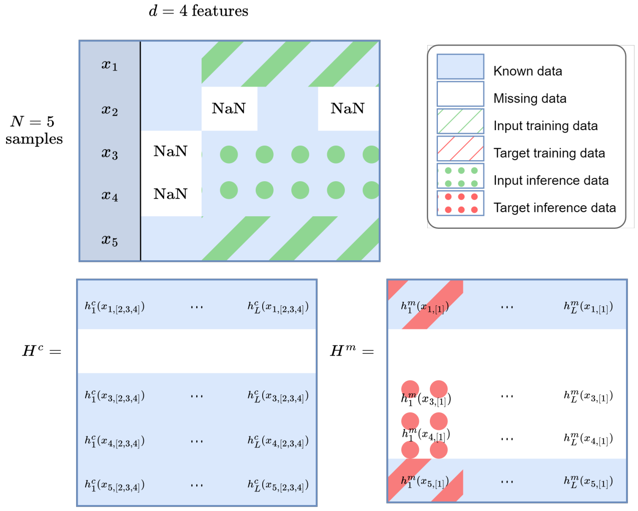
$d = 4$ features
$N = 5$ samples
x_1
x_2
x_3
x_4
x_5
NaN
NaN
NaN
NaN
Known data
Missing data
Input training data
Target training data
Input inference data
Target inference data
$H^c =$
$h_1^c(x_{1,[2,3,4]})$ $\cdots$ $h_L^c(x_{1,[2,3,4]})$
$h_1^c(x_{3,[2,3,4]})$ $\cdots$ $h_L^c(x_{3,[2,3,4]})$
$h_1^c(x_{4,[2,3,4]})$ $\cdots$ $h_L^c(x_{4,[2,3,4]})$
$h_1^c(x_{5,[2,3,4]})$ $\cdots$ $h_L^c(x_{5,[2,3,4]})$
$H^m =$
$h_1^m(x_{1,[1]})$ $\cdots$ $h_L^m(x_{1,[1]})$
$h_1^m(x_{3,[1]})$ $\cdots$ $h_L^m(x_{3,[1]})$
$h_1^m(x_{4,[1]})$ $\cdots$ $h_L^m(x_{4,[1]})$
$h_1^m(x_{5,[1]})$ $\cdots$ $h_L^m(x_{5,[1]})$

Figure 2: Training and inference data (in H^m) for the secondary ELM. This ELM predicts the first activation neuron, knowing the 3 last features. Dash lines represent the training data while the dots represent inference data and predictions.

Level of Incompleteness	5%		15%	
Name of the dataset	Abalone	Wine	Abalone	Wine
Mean Imputation	0.57	/	0.35	0.85
K-NN Imputation	0.07	/	0.09	0.77
Multi-ELM Model	**0.06**	/	**0.08**	**0.74**

Table 1: MSE comparison on the 2 datasets with different values of incompleteness.

6 CRITIQUE

This paper proposes a new method for dealing with missing values in a dataset. The authors propose to predict a target feature using a primary ELM, and estimate the parameters of this ELM using multiple secondary ELMs. It would be interesting to apply this second estimation to a more robust deep learning method. Indeed, the general idea is to estimate the activation function of the neurons on which the data is missing. We could therefore replace the primary ELM with any deep learning method featuring neurons.

Moreover, the algorithm poses some problems not solved by the authors. First of all, the authors decompose the matrix H into two sub-matrices H^c and H^m. However, this decomposition is only possible if the activation function is linear, which is never the case. The activation function can be quasi-linear by using a ReLU. However, this decomposition is valid when done before the activation function. Thus, Eq. (3) needs to be slightly changed.

Furthermore, it would be interesting to compare the training times of the algorithm and to compare them with the usual methods.

[1] This decomposition on h proposed by the article is not exactly correct. The activation function ϕ isn't linear. Thus, the decomposition needs to be done before the activation function.

Indeed, training $2^d - 1$ EMLs can be extremely time consuming. The datasets considered here are small: we would like to know how this training time evolves with respect to the dataset size. Finally, although the multi-EML model is more efficient than the K-Nearest Neighbor implementation, the difference in performance seems very small. Thus, it would have been interesting to see if the proposed method is indeed statistically better.

7 CONCLUSION

One of the most common problems we face in data is missing values. We have come across different techniques for dealing with missing or incomplet values including imputation and discarding data. However, the outcomes are not always satisfactory, particularly when there are a significant number of missing variables. The work introduced by Chi et al [1], gave us the opportunity to discover a novel method to deal with this problem in an efficient way. This method is based on a different type of machine learning algorithms known as Extreme Learning Machines (ELM), It can be applied directly to data without requiring any further imputation and produces better results, particularly with datasets with a variety of missing patterns.

REFERENCES

[1] Baichuan Chi and Amaury Lendasse. [n. d.]. A Multi-ELM Model for Incomplete Data. In *Summer Undergraduate Research Fellowship*. https://uh-ir.tdl.org/handle/10657/11675

[2] Paulo Cortez, António Cerdeira, Fernando Almeida, Telmo Matos, and José Reis. 2009. Modeling wine preferences by data mining from physicochemical properties. *Decision Support Systems* 47, 4 (2009), 547–553. https://doi.org/10.1016/j.dss.2009.05.016 Smart Business Networks: Concepts and Empirical Evidence.

[3] Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

[4] Alireza Farhangfar, Lukasz Kurgan, and Jennifer Dy. 2008. Impact of imputation of missing values on classification error for discrete data. *Pattern Recognition* 41, 12 (2008), 3692–3705. https://doi.org/10.1016/j.patcog.2008.05.019

[5] Jerzy W. Grzymala-Busse and Witold J. Grzymala-Busse. 2005. *Handling Missing Attribute Values*. Springer US, Boston, MA, 37–57. https://doi.org/10.1007/0-387-25465-X_3

[6] Eduardo Hruschka, Estevam Hruschka, and Nelson Ebecken. 2003. Evaluating a Nearest-Neighbor Method to Substitute Continuous Missing Values, Vol. 2903. 723–734. https://doi.org/10.1007/978-3-540-24581-0_62

[7] Guang-Bin Huang, Qin-Yu Zhu, and Chee-Kheong Siew. 2006. Extreme learning machine: Theory and applications. *Neurocomputing* 70, 1 (2006), 489–501. https://doi.org/10.1016/j.neucom.2005.12.126 Neural Networks.

[8] Mortaza Jamshidian and Peter M. Bentler. 1999. ML Estimation of Mean and Covariance Structures with Missing Data Using Complete Data Routines. *Journal of Educational and Behavioral Statistics* 24, 1 (1999), 21–41. http://www.jstor.org/stable/1165260

[9] Yoan Miche, Antti Sorjamaa, Patrick Bas, Olli Simula, Christian Jutten, and Amaury Lendasse. 2010. OP-ELM: Optimally Pruned Extreme Learning Machine. *IEEE Transactions on Neural Networks* 21, 1 (2010), 158–162. https://doi.org/10.1109/TNN.2009.2036259