Procedia Computer Science
Volume 53, 2015, Pages 167–174
2015 INNS Conference on Big Data

On the Distributed Implementation of Unsupervised Extreme Learning Machines for Big Data

Yara Rizk[1] and Mariette Awad[2]

[1] American University of Beirut, Beirut, Lebanon
yar01@aub.edu.lb
[2] American University of Beirut, Beirut, Lebanon
mariette.awad@aub.edu.lb

Abstract

The emergence of the big data problem has pushed the machine learning research community to develop unsupervised, distributed and computationally efficient learning algorithms to benefit from this data. Extreme learning machines (ELM) have gained popularity as a neuron based architecture with fast training time and good generalization. In this work, we parallelize an ELM algorithm for unsupervised learning on a distributed framework to learn clustering models from big data based on the unsupervised ELM algorithm proposed in the literature. We propose three approaches to do so: 1) Parallel US-ELM which simply distributes the data over computing nodes, 2) Hierarchical US-ELM which hierarchically clusters the data and 3) Ensemble US-ELM which is an ensemble of weak ELM models. The algorithms achieved faster training times compared to their serial counterparts and generalized better than other clustering algorithms in the literature, when tested on multiple datasets from UCI.

Keywords: Big data, Extreme Learning Machines, Unsupervised learning, Distributed computing

1 Introduction

The surge in mobile computing and social media sites in addition to the increase in memory storage capabilities has lead to the abundance of large amounts of unstructured and unlabelled data that need analysis. The most common method to alleviate the big data problem is to distribute the computations over clusters of computing nodes. Since labelling the data is expensive, developing unsupervised machine learning (ML) algorithms for big data has become a popular research area. Neural networks (NN) are popular ML algorithms composed of connected neurons, proven to be universal approximators [3, 5], that have been trained in supervised, semi-supervised and unsupervised environments. Many generations of neural networks have been developed starting from the perceptron to the more recent deep network architectures. Various learning algorithms such as backpropagation and greedy layer based training, have also

Selection and peer-review under responsibility of the Scientific Programme Committee of INNS-BigData2015

doi:10.1016/j.procs.2015.07.291

been proposed to train these networks and produced improving results. Extreme learning machines (ELM) became popular due to their fast training times compared to other NN training algorithms [7] while preserving the universal approximators property of neurons [9, 8].

Our work improves on the clustering ELM algorithm presented in [6], which is referred to as Basic US-ELM hereafter, by proposing multiple distributed implementations. Parallel US-ELM simply distributes the data over the computing nodes which map the data to the feature space, then performs the clustering algorithm. Hierarchical US-ELM hierarchically clusters the data by finding the cluster centers of the distributed data on the computing nodes, then clusters these centers on a single node. Finally, the Ensemble US-ELM implements an ensemble US-ELM which runs weak US-ELM models on the computing nodes and aggregates the votes using a majority vote scheme. Furthermore, the k-means in the serial algorithm is replaced by other algorithms and compared to the original implementation, in addition to other algorithms in the literature on multiple datasets from the UCI repository [10].

Next, we present a literature review of related works. Section 3 presents the proposed distributed ELM based algorithms for unsupervised learning. Section 4 presents a theoretical analysis of the computational complexity of US-ELM. Section 5 reports on the experimental results on multiple datasets while Section 6 concludes the paper.

2 Literature Review

Several ELM implementations on distributed frameworks have been proposed. Chen et al. parallelized an ELM ensemble classifier on MapReduce where the mapper was one ELM network and the reducer aggregated the classifiers' votes to obtain the final classification [2]. Van Heeswijk et al. presented an ensemble ELM algorithm for big data regression problems implemented on GPUs [12]. Xin et al. implemented an ELM algorithm for classification on MapReduce by distributing the Moore-Penrose generalized inverse [15]. He et al. presented a MapReduce implementation of the ELM algorithm for regression by distributing the matrix computations [4]. Bi et al. implemented a kernel ELM for classification on MapReduce by distributing the kernel and Moore-Penrose generalized inverse [1].

Zhou et al. proposed a stacked ELM targeted to improve ELM's performance on large datasets [17]. Instead of using a single layer feedforward network with a large number of neurons, the number of layers was increased and the number of neurons per layer were decreased to reduce the amount of computations on large datasets. PCA was used to reduce the number of neurons per layer and the network was iteratively trained. Wang et al. presented a parallelized ELM decision tree that used information entropy and ambiguity measures to split the nodes and apply ELM at the leafs[14]. Parallelizing the information entropy, gain, and matrix computations, allowed its application to big data classification problems.

Since online sequential ELM (OS-ELM) suffers from an inability to halt when learning from large datasets when the data does not present new information that requires model updates, Zhai et al. introduced a halting condition by dividing the data into training, validation and testing sets where the lack of improvement on the validation set accuracy was used as a termination condition and trained an ensemble of OS-ELM on large datasets [16]. Wang et al. distributed the OS-ELM computations using MapReduce by parallelizing the matrix multiplications [13].

Huang et al. proposed an unsupervised ELM algorithm, referred to as Basic US-ELM in this text, for clustering problems [6]. The algorithm transformed the data from the input space to a feature space using an ELM network and applied K-means to the embedding matrix. Basic US-ELM outperformed other clustering algorithms but did not target big data problems and was not parallelized.