Semi-Supervised Approach Using Nearest Neighbors Clustering and Deep Learning

Bruno V. A. Lima ∗ Adrião D. D. Neto ∗∗ Lúcia E. S. Silva ∗∗∗ Vinicius P. Machado ∗∗∗∗ João G. C. Costa †

∗ Department of Computer Engineering and Automation, Federal University of Rio Grande do Norte, RN, (e-mail: brunovicente@ufrn.edu.br).
∗∗ Department of Computer Engineering and Automation, Federal University of Rio Grande do Norte, RN (e-mail: adriao@dca.ufrn.br)
∗∗∗ Department of Computer, Federal University of Piauí, PI, (e-mail:luciaemilia72@gmail.com)
∗∗∗∗ Department of Computer, Federal University of Piauí, PI, (e-mail:vinicius@ufpi.edu.br)
† Institute of Mathematical and Computer Sciences, Universidade of São Paulo, SP, (e-mail:Joaoacavalcanti@usp.br)

Abstract: The field of Deep Learning is in constant evolution, with new techniques and applications being developed by the day. Of those techniques, semi-supervised deep learning have promising results, especially in combination with the standard Convolutional Neural Network (CNN) architectures. CNNs attain state-of-the-art performance on various classification tasks assuming a sufficiently large number of labeled training examples. Unfortunately, labeling sufficiently large training data sets requires human involvement, which is expensive and time consuming. In semi-supervised learning there is not only a set of labeled samples (L), but also a set of unlabeled samples (U), which is generally greater than the first (U > L). This paper presents a semi-supervised model using a CNN supported by a Multilayer Perceptron (MLP) network, and a clustering process by k Nearest Labeled Neighbors. The results showed that the proposed model solves the semi-supervised learning problem over different scenarios.

Keywords: Semi-supervised; Deep Learning; Neighbors; Labeling.

1. INTRODUCTION

Recent developments in technology increased the possibilities on how to obtain data. For example, one can use the Internet to search any information on economics, industrial designs, health problems, scientific results and so on. The technological advance has also contributed to an accumulation of information, by increasing the capacity of storage devices with higher speed of access and lower cost.

Some of this information can be analyzed in order to generate knowledge that assists in decision-making tasks related to the data. Usually, machine learning techniques are used to make such analyzes. One of those techniques is semi-supervised learning, which handles classification problems. In these problems, a small amount of samples have a label which specifies the class of the sample. In contrast, most of the samples are unlabeled, meaning they are not previously classified. This group is usually larger because it is easier to acquire such type of information, as it does not require a prior processing.

In the context of semi-supervised learning, techniques are needed to process labeled and unlabeled data simultaneously. In this sense, this paper presents a model for working with semi-supervised learning using a Deep Learning technique known as Convolutional Neural Network (CNN), and supported by a Multilayer Perceptron network (MLP).

Experimental tests have shown the efficiency of the model in solving semi-supervised learning problems, such as processes clustering, data labeling, and classifying data. Outperforming the classical methods Co-training and SEKED K-means.

2. RELATED WORKS

In Sajjadi et al. (2016) the problem of semi-supervised learning with deep Convolutional Neural Networks is introduced. The authors propose an unsupervised regularization that explicitly forces the classifier’s prediction for multiple classes to be mutually exclusive and effectively guides the decision boundary to be in the low-density space between the manifolds corresponding to different classes of data. The method can be applied to images.

In Chamberlain et al. (2016) it is presented the development of a semi-supervised deep learning algorithm for automatically classify lung sounds using a Convolutional Neural Network, this paper solve a specificy problem with Deep Semi-supervised.

In Noroozi et al. (2017), it is proposed a deep semisupervised model named SEmi-supervised VERification Network
The model consists of two complementary components: the generator component, which addresses the lack of supervision within each category by learning general salient structures from a large amount of data across categories, and the discriminative component, which explores the general features to mitigate the lack of supervision within categories, and further devote the generator component to find more informative structures of the whole data manifold. The two components are interconnected at SEVEN allowing end-to-end training for both.

In Baur et al. (2017), apply concept of auxiliary manifold embedding for semi-supervised learning to Fully Convolutional Networks with the help of Random Feature Embedding. This paper uses imagens in the Semi-Supervised Problem.

A new generative model of the crowdsourcing labeling process is presented in Kingma et al. (2014). It leverages unlabeled data effectively by introducing latent resources and data distribution. As the distribution of the data can be complicated, a deep neural network was used to do it. Therefore, the model can be considered as a kind of deep generative model.

### 3. SEMI-SUPERVISED LEARNING - THE PROBLEM

In a ML learning process, if there are labeled samples, it is possible to use a supervised learning technique to induce classifiers. Otherwise, if the samples are not labeled, the unsupervised learning techniques are used to find clusters. However, in many learning processes, to create labeled data is, generally, expensive and time consuming Amini and Gallinari (2003) Basu et al. (2002). On the other hand, to use unsupervised algorithms can be undesirable or not applicable to the problem.

Semi-supervised learning is the middle ground between the supervised and unsupervised learning Bruce (2001). In this method, there are two kinds of data used on the learning process: unlabeled and labeled data; the later are used to classify the first.

The semi-supervised learning process can be described as a set of labeled data defined by $L = \{x_i, y_i\}_{i=1}^n$ and a set of unlabeled data defined by $U = \{x_i\}_{i=m+1}^n$, where $m >> n$. The purpose is to find a data set $L' = \{x_i, y_i\}_{i=1}^m$, where $L'$ is the new labeled set obtained from the set $U$. Thus, for every element $x_i$ there is an associated label $y_i$, which results in a fully labeled set.

#### 3.1 SEEDED K-means

The SEEDED K-means Algorithm is a semi-supervised K-means variant that also partitions the data set into K clusters. The difference is in the construction of the constraints: while SEEDED K-MEANS uses the set of labeled examples to calculate the initial K centroids of the clusters (SEEED), K-means chooses them randomly Basu et al. (2002).

#### 3.2 Co-Training

Co-training is a semi-supervised algorithm based on the cooperation between two supervised algorithms. Its goal is to label a data set $U$ from a previously labeled set $L$ Blum and Michell (1998).

For this, two descriptions are created where the set of attributes $X$ of $L$ are subdivided into $X_{D1}$ and $X_{D2}$, where $X_{D1} = \{x_1, x_2, ..., x_j\}$, $X_{D2} = \{x_{j+1}, x_{j+2}, ..., x_F\}$ and $F$ is the number of features. Each classifier is trained with the subsets of $X$ ($X_{D1}$ and $X_{D2}$), cooperating to label the elements of the set $U$.

### 4. PROPOSED METHOD

The proposed method labels a set of unlabeled data $U = \{x_i\}_{i=1}^n$ by considering the data of a labeled set $L = \{x_i, y_i\}_{i=1}^n$ where $m >> n$. The objective is to learn a parameterized nonlinear function capable of predicting the relationship between $U$ and a set $y'$, where $y'$ is formed by the labels of $L$. The Figure 1 presents the graphic representation of the proposed method.

As shown in the Figure 1, the model has as input the set $L$ (labeled) and $U$ (unlabeled) and is divided into four phases: Normalization, Pre-Labeling, Evaluation and Labeling.

#### 4.1 Normalization

Before processing the input, the model normalizes the data using the Min-Max normalization, which sets all the characteristics to the same range, between 0 and 1. The Equation 1 describes the normalization applied in the proposed method.

$$v' = \frac{v - \min A}{\max_A - \min_A}$$

The original value of the data is represented by $v$, and the normalized value by $v'$. $\min_A$ and $\max_A$ describe, respectively, the minimum and the maximum bounds of the distribution to be normalized.

#### 4.2 Pre-Labeling

The Pre-Labeling phase classifies the set $U$ by using the set $L$. Here, the $k$ closest labeled neighbors are used to obtain the initial probability of a given element $x' \in U$ to belong to one of the $c$ classes of the problem. This method results in a vector of probabilities $P$ with size $c$ for each element.

The choice of the $k$ closest labeled neighbors for a given element $x'$ is done by calculating the Euclidean distance to each labeled sample $x'^2$ (Equation 2), selecting the $k$ closest.

$$d(x, x') = \sqrt{\sum_{i=1}^{q} (x_i - x'_i)^2}$$

The probability of an element $x$ to belong to a particular class $A$ is calculated by $P(x|A) = \frac{q}{k}$, where $q$ is the number of times that the class $A$ appears between the $k$ nearest neighbors.
The Figure 2 illustrates the analysis process for an unlabelled element (marked as X) by considering three classes: red, blue, and green. Assuming $k = 5$, the algorithm selects the 5 nearest labeled neighbors (the colored circles inside the greater one).

For these $k$ selected elements, there is the following distribution: $q_{\text{Red}} = 3$, $q_{\text{Blue}} = 2$, $q_{\text{Green}} = 0$. Thus, the vector of probabilities $P^n$ for the element $X$ to be labeled as one of the three classes is:

$$P(X|\text{Red}) = \frac{3}{5} = 0.6$$

$$P(X|\text{Blue}) = \frac{2}{5} = 0.4$$

$$P(X|\text{Green}) = \frac{0}{5} = 0$$

In this example, the element $X$ is labelled as belonging to the class Red.

For this phase, the set $L$ feeds an MLP which calculates the probabilities of an element $X$ to belong to each class, outputting a second vector of probabilities $P^m$.

The Figure 3 presents the MLP’s architecture. The hidden layer neurons use the ReLU function and the output layer neurons use the softmax function to calculate the probability.

At the end of the Pre-labeling phase, there are two vectors of probabilities: $P^n$, provided by the $K$ nearest neighbors; and $P^m$, provided by the MLP trained with the set $L$.

### 4.3 Evaluation

After the Pre-Labeling step, an Evaluation is performed to define the label of each element of $U$. The label is calculated by the following Equation 6.

$$\vec{P} = \begin{bmatrix} P^m_0 + P^n_0 \\ P^m_1 + P^n_1 \\ \vdots \\ P^m_c + P^n_c \end{bmatrix}$$

$$\vec{P} = \begin{bmatrix} \frac{P^m_0 + P^n_0}{2} \\ \frac{P^m_1 + P^n_1}{2} \\ \vdots \\ \frac{P^m_c + P^n_c}{2} \end{bmatrix}$$
The average probability vector \( \vec{P} \) (Equation 6) between \( \vec{P}_m \) (Classification) and \( \vec{P}_n \) (Euclidean Distance) is calculated and then normalized to either 0 or 1, following the Equation 7.

\[
P_i = \begin{cases} 
0, & \text{if } P_i \leq 0.5 \\
1, & \text{if } P_i > 0.5 
\end{cases} 
\]  

(7)

If every probability in the vector \( \vec{P} \) relative to an element \( x \) is equal to 0, then it is not possible to label \( x \). \( x \) is then added to the set \( U' \), which contains the elements without a defined label. Otherwise, if any position of \( \vec{P} \) has the probability of 1, then \( x \) is labeled with the corresponding class and added in the set \( L' \).

4.4 Labeling

After the Evaluation stage, the sets \( L \) and \( L' \) are fed into a CNN, in order to label the set \( U' \). Finally, the set \( L^T = (x^i, y^i)_{i=1}^{m+n} \) is the output of the network, which represents the entire set of data \( (L^T = L \cup U) \), as well as all the \( y^i \) labels for the entire data.

5. EXPERIMENTS METHODOLOGY

5.1 Databases

The following databases were used to evaluate the proposed model: MNIST and Wine Quality Data set.

The MNIST data set is formed by images of numeric digits from 0 to 9 written by hand. This data set is a subset from the NIST data set and has all the elements centered on a 28x28 image. In the tests ran in this work, the 60,000 images were transformed to one dimension, thus, each sample had a total of 756 features \( (28 \times 28 = 756) \). The original data set can be found in Kaggle Repository\(^1\). The Wine Quality data set can be found in the UCI Repository\(^2\). This data set contain samples that describe wine quality in a scale from 0 to 10. It has a total of 12 features and 4898 samples.

5.2 Experiments

Two main experiments were carried out in this work. In the first one, different amount of neighbors \( k \) were tested. This experiment aims to compute the influence of this parameter to the performance of the proposed model. Thus, the proposed model was tested with the two aforementioned databases, ranging from 1 to 100.

For the tests, each database was divided into 10 subsets. For each test iteration, one of these subsets was chosen as the labeled set \( L \), and the 9 others as the unlabeled set \( U \). Thus, 10 iterations were performed for each value of \( k \). The results were compared by the average of the following metrics: Accuracy, Precision, Recall and F-Measure.

The second experiment aims to compare the performance of the proposed model to the state-of-the-art algorithms:

\(^1\) https://www.kaggle.com/

Santeed K-means Basu et al. (2002) and Co-Training Blum and Michell (1998). Random samples were selected to compose the set \( L \), using partitions of 5%, 6%, 7%, 8%, 9% and 10% of the data set. Each algorithm was tested 10 times for each partition and the averaged Accuracy, Precision, Recall, and F-Measure metrics were calculated.

5.3 Used Metrics

To evaluate the performance of the algorithms in the experiments, the following metrics were used: Accuracy, Recall, Precision and F-Measure. A Confusion Matrix was used to calculate the metrics by using TP (True Positive), FP (False Positive), TN (True Negative) and FN (False Negative) Sokolova and Lapalme (2009).

The Accuracy is calculated by the number of TPs divided by the total number of samples in the test set. The Precision metric (Equation 8) reflects the ratio of TPs to all positive predictions (False and True). The Recall metric (Equation 9) is the ratio of TPs to positive predictions plus incorrect negative predictions.

\[
\text{Precision} = \frac{VP}{VP + FP} \quad (8) \\
\text{Recall} = \frac{VP}{VP + FN} \quad (9)
\]

In addition to these metrics, the F-Measure (Equation 10) is used to calculate the balance between Accuracy and Recall. The results range from 0 to 1 for all metrics. Therefore, the results indicate a better learning when their values are closer to 1.

\[
\text{F-Measure} = \frac{2 \cdot \text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}} \quad (10)
\]

These metrics were calculated using the Scikit-learn library for Python Pedregosa et al. (2011).

6. RESULTS

As mentioned in the previous section, two experiments were carried out to evaluate the model in a Semi-supervised Learning problem.

The first experiment verified the performance of the algorithm by varying the value of \( k \) (from 1 to 100), measured by running 10 iterations for each value. In this experiment the MLP had 128 neurons in the hidden layer, for both data sets.

The tests with the MNIST data set had a total of 5 convolution layers, followed by two neural hidden layers: the first one with 256 neurons and second one with 64 neurons, and there is a Dropout of 0.5 between the layers. For the Wine Quality database, one convolution layer was used. The following neural layers are the same as the one for the MNIST.

The Figure 4 shows how \( k \) affects the performance of the proposed model when labeling the databases. The reader can note that the proposed model obtained satisfactory accuracy. For the MNIST data set, its label accuracy is
above 0.9, for all values of $k$. For the Wine Quality data set, it also obtained an accuracy over 0.9, except when $k$ approached 100, which implies that the value of $k$ affects the performance of the model depending on the problem.

In addition to the Accuracy metric, the Figure 4 also shows the values of the Recall and Precision metrics, where the consistency of the classification is perceived. Consequently, the F-Measure rate shows the balance between these two rates (Recall and Accuracy).

The standard deviation for each iteration was ≤0.012 for the MNIST test, and ≤0.035 for the Wine Quality test.
For the second experiment, the proposed model was compared with the Co-training and SEEDED K-means algorithms. The Figure 5 shows the performance of the algorithms for the MNIST data set, where only a percentage of the whole data set is used as the subset L. The results show that the proposed model obtained satisfactory accuracy, above 0.9. The SEEDED K-means algorithm presented lower accuracy, assuming values between 0.8 and 0.9, while the Co-Training algorithm performed considerably lower than the others, obtaining accuracy below 0.77.

The Figure 6 shows the performance of the algorithms in labeling of the Wine Quality data set. Here, it is perceived that the proposed model obtained worst results than the SEEDED K-means algorithm when using 5% or 6% of the data as the training set. The inferior performance of the algorithm Co-training when compared to the others models can also be noticed, even with the perceivable improvement.

The Recall and Precision metrics, in Figures 5 and 6, show that the accuracy rate also reflects the labeling in term of classes, as well as the F-measure rate shows the equilibrium of these rates, meaning that there was no imbalance in labeling when comparing to individual classes.

It is observed that the model performed better when tested with a set of more complex data (the MNIST data set contains 60,000 samples and 756 features). However, the SEEDED K-means algorithm works better when the problem is of less complexity, since it obtained better performance by labeling the Wine Quality database with 5% and 6% of labeled samples.

And although the SEEDED K-means achieved better accuracy for $L = 8\%$ of the Wine Quality data set, it also had higher standard deviation ($\sigma$) than the proposed model. So it can be inferred that the results are statistically equivalents.

The Table 1 shows all the accuracy obtained in this experiment, followed by the respective standard deviation. It can be observed that for the 10 iterations ran for each size of $L$, the proposed model obtained statistically satisfactory accuracy.

7. CONCLUSION

This paper presented a Deep Semi-supervised model combining a nearest neighbors cluster approach and a multi-layer neural network to assist in the training of a Convolutional Neural Network to solve the Semi-supervised Learning problem.

Results showed that the proposed method presented similar or better performance than the semi-supervised the state-of-the-art algorithms: SEEDED K-means and Co-training. The algorithms were tested with two databases well known in the literature. By comparing the results, it is perceived that the proposed model contributes to the solving of Semi-Supervised Learning problems, achieving satisfactory results.

As future work, the authors intend to use only Deep Learning methods to execute all the steps of the model. The investigation of how to use Deep Semi-Supervised Clustering techniques to assist in the training of a Convolution Network is also considered for future work.

REFERÊNCIAS


