UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

A New Number Recognition Approach Using Typical Testors, Genetic Algorithms and Neural Networks

Eddy Alejandro Torres Constante

Ingeniería en Ciencias de la Computación

Trabajo de fin de carrera presentado como requisito
para la obtención del título de
Ingeniero en Ciencias de la Computación

Quito, 6 de mayo de 2021

UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

HOJA DE CALIFICACIÓN
DE TRABAJO DE FIN DE CARRERA

A New Number Recognition Approach Using Typical Testors, Genetic Algorithms and Neural Networks

Eddy Alejandro Torres Constante

Nombre del profesor, Título académico **Noel Pérez, Ph.D.**

Quito, 6 de mayo de 2021

© DERECHOS DE AUTOR

Por medio del presente documento certifico que he leído todas las Políticas y Manuales de la Universidad San Francisco de Quito USFQ, incluyendo la Política de Propiedad Intelectual USFQ, y estoy de acuerdo con su contenido, por lo que los derechos de propiedad intelectual del presente trabajo quedan sujetos a lo dispuesto en esas Políticas.

Asimismo, autorizo a la USFQ para que realice la digitalización y publicación de este trabajo en el repositorio virtual, de conformidad a lo dispuesto en la Ley Orgánica de Educación Superior del Ecuador.

Nombres y apellidos: Eddy Alejandro Torres Constante

Código: 00138355

Cédula de identidad: 1104732043

Lugar y fecha: Quito, 6 de mayo de 2021

ACLARACIÓN PARA PUBLICACIÓN

Nota: El presente trabajo, en su totalidad o cualquiera de sus partes, no debe ser considerado como una publicación, incluso a pesar de estar disponible sin restricciones a través de un repositorio institucional. Esta declaración se alinea con las prácticas y recomendaciones presentadas por el Committee on Publication Ethics COPE descritas por Barbour et al. (2017) Discussion document on best practice for issues around theses publishing, disponible en http://bit.ly/COPETheses.

UNPUBLISHED DOCUMENT

Note: The following capstone project is available through Universidad San Francisco de Quito USFQ institutional repository. Nonetheless, this project – in whole or in part – should not be considered a publication. This statement follows the recommendations presented by the Committee on Publication Ethics COPE described by Barbour et al. (2017) Discussion document on best practice for issues around theses publishing available on http://bit.ly/COPETheses.

RESUMEN

Los testores, y en particular los testores típicos, se han utilizado en la selección de características y problemas de clasificación supervisada. Un objetivo al elegir qué características seleccionar es preservar la precisión. Esto también permite que un modelo mantenga su capacidad para discriminar entre clases. En consecuencia, proponemos el algoritmo de construcción del discriminador como una estrategia para lograr un subconjunto de características que preserven la precisión en la clasificación. Algunos algoritmos, en la literatura, solo se enfocan en encontrar el conjunto completo de testores típicos o en construir los de longitud mínima sin considerar la precisión en la predicción sobre los clasificadores. En este artículo, proponemos un algoritmo que utiliza los conceptos de testor típico y la precisión en la predicción sobre redes neuronales como funciones objetivo para la estrategia evolutiva con la cual construir un subconjunto de características que preserven la precisión; en el mismo rango que si usáramos todo el conjunto de características para entrenar y probar el modelo. El artículo presenta algunos resultados experimentales obtenidos mediante el uso de un discriminador construido sobre el conjunto de datos NMIST con las cuales se discute sobre métricas de evaluación a un modelo entrenado con el conjunto reducido de características. También contrastamos con la tasa de error de otros clasificadores informados sobre el mismo conjunto de datos con la tasa de error del nuevo discriminador.

Palabras clave: clasificación numérica manuscrita, testores típicos, red neuronal multicapa, algoritmos genéticos, precisión, estrategia evolutiva, función objetivo.

ABSTRACT

Testors, and particularly typical testors, have been used in feature selection and supervised classification problems. An objective on choosing what features select is to preserve accuracy. This also allows a model to keep its capacity to discriminate between classes. Consequently, we propose the discriminator construction algorithm as an strategy to achieve a subset of features that preserves accuracy. Some algorithms, in literature, only focus on finding the whole set of typical testors or building minimum length ones which do not consider accuracy on prediction over classifiers. In this paper, we propose an algorithm that uses typical testors concepts and accuracy on prediction over neural networks as fitness functions for evolutionary strategy to build a subset of features that preserves the accuracy; in the same range as if we use the whole set of features to train and test the model. The paper presents some experimental results obtained by using a discriminator constructed over NMIST data set to discuss assessment metrics of the model with the reduced set of features. We also contrast with other reported classifiers error rate over the same data set with the error rate of the new discriminator.

Key words: handwritten number classification, typical testors, multi-layer neural network, genetic algorithms, accuracy, evolutionary strategy, fitness function.

TABLA DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

I. INTRODUCTION

The concept of testor has been adapted to different applications over the years, one of them has been using testors to determine how much relevant information a set of features stores to describe a group of objects. This information is helpful to classify and recognize those objects (Alba-Cabrera et al, 2016). Being able to determine which features are relevant and which are redundant is an important task in the field of supervised classification, as well as in pattern recognition problems (Liu et al, 2007). This is known as feature selection and it is the process of selecting a subset of features to reduce future search, training over data for classification or prediction spaces that go according to certain evaluation criteria for each of their fields (Alba-Cabrera et al, 2016).

The objective of reducing the dimensionality of the feature space is to find a minimum set of them that preserves the essential information and allow to distinguish and identify the compared classes, facilitating the training of models for areas of pattern recognition or data mining (Mafarja et al, 2019), (Saxena et al, 2019), (Wang et al, 2019), (Zhou et al, 2019).

A testor has the same ability to differentiate between objects that belong to different classes as the entire set of features does. This is why it has been used in supervised pattern recognition as in (Carrasco et al, 2004), (Pons-Porrata et al, 2007), (Lopez-Perez et al, 1997), (Valev et al, 1991). The subspace where to find the testors and the cardinality of the set of all testors is enormous since they are searched among all possible subsets of the columns of the matrix where every column is a feature (Alba-Cabrera et al, 2016), (Lazo-Cortez et al, 2001), (Valev et al, 2003).

Some state-of-the-art algorithms include: LEX (Alganza et al, 2003), YYC (Alba-Cabrera et al, 2014), all-NRD (Asaithambi et al, 2004), CUDA based hill-climbing (Piza-

Davila et al, 2017), Fast-BR (Lias-Rodriguez et al, 2013), FAST-CT-EXT (Sanchez-Diaz et al, 2014) which focus on returning the whole set of typical testors; it could be possible to compare the accuracy of prediction over all testors on these algorithms if it guarantees to find the whole set of typical testors, nevertheless in most real cases the exponential complexity does not allow it. It is also reported an algorithm to find minimum length typical testors (Piza-Davila et al, 2020). However, heuristic algorithms as UMDA (Diaz-Sanchez et al, 2011), PHC (Piza-Davila et al, 2015), HC (Sanchez-Diaz et al, 2014) have a better performance over large data sets returning a subset of the of typical testors. The output of all types of algorithms focus on trying to reach the highest cardinality of the returned subset of typical testor or minimum-length subset of them. Therefore, we propose focusing on how effective can we build a testor so that we can use its properties of classification properly, since there are problems where the entire set or either a big subset is required and reducing the amount of features with no loss of accuracy will speed up the prediction process (Sanchez-Diaz et al, 2014). Like in (Gallegos et al, 2016) where testors are used to improve the diagnosis of breast cancer cells. Furthermore, there are several problems in which there is a necessity to find an optimal discriminator (or close to it), that allows maximizing the performance of the classification by reducing the number of features used; specially in the fields of diagnostic diseases as in previous example and in (Ortiz-Posadas et al, 1996), feature selection for text classification (Carrasco-Ochoa et al, 2004) and categorization (Pons-Porrata et al, 2007). Unlike other techniques developed for feature selection, testors have focused on this purpose, especially a certain type of them known as irreducible or typical (Betch et al, 2019), (Fernandez et al, 2019), (Lazo-Cortes et al, 2015), (Sayed et al, 2019), (Singh et al, 2019).

The time complexity increases as the number of features grow. Finding testors with previous algorithms do not give metrics over how useful will be one testor over another or

even the whole set when using it for discrimination. Furthermore, considering how a testor can be used in practice, is why we propose to build one. In fact, (Diaz-Sanchez et al, 2011) considers that each typical testor can be recognized as a local optimum for discrimination. To achieve it we start from evolutionary strategy to reach typicality as near as possible. The objective function is also defined in terms of accuracy on testing, over a trained only with selected features multi-layer feed-forward back-propagation neural network. Thus, each result can be compared in terms of its efficiency in a reasonable computer time under given conditions of structure of the testor like the number of features desired in it. In this sense, prediction accuracy plays the roll of fitness function over the evolution strategy; it selects not only the typical features but also the ones that increases the prediction accuracy of the model.

In this work we propose by using previous evolutionary techniques and neural networks an approach to build an accuracy enough over prediction testor. As first we formally introduce the concept of an typical testor for a boolean matrix, we describe the multi-layer feed-forward neural networks, and also genetic algorithm strategy previously used to achieve an equal or closed enough typical testor with UMDA fitness function (Alba-Cabrera et al, 2000) as theoretical background; we detail our proposed method. After we present results and analysis of the study. Finally, some conclusions and future work is presented.

II. MATERIALS AND METHODS

A. Typical Testor

Let U be a collection of objects, these objects are described by a set of n features and are grouped into l classes. By comparing feature to feature each pair of objects belonging to different classes, we obtain a matrix $M = [m_{ij}]_{pxn}$ where $m_{ij} \in \{0,1\}$. $m_{ij} = 0$ and $m_{ij} = 1$ means that the objects of pair denoted by i are similar or different respectively, in the feature j. Let $I = \{i_1,...,i_p\}$ be the set of the rows of M and Let $J = \{j_1,...,j_n\}$ the set of labels of its columns.

Let a and b two rows of M.

Definition 1: We say that $a<b$ if $\forall\ i\ a_i \leq b_i$, and $\exists\ j$ such that $a_j \neq b_j$ (Ruiz-Shulcloper et al, 1980).

Definition 2: a is a basic row of M if there is no other row less than a in M (Ruiz-Shulcloper et al, 1980).

Definition 3: The *basic matrix* of M is the matrix B only containing all different basic rows of M.

Let $T \subseteq J$, B^T be a subset of features obtained from B eliminating all columns not belonging to the set T.

Definition 4: Let p be a row of B^T; we say that p is a zero row if it contains only zeros (Piza-Davila et al, 2017).

Definition 5: A set $T = \{j_{k1},...,j_{ks}\} \subseteq J$ is a testor of M if no zero row in B^T exists (Piza-Davila et al, 2017).

Definition 6: The feature $x_i \in T$ is called a non-removable feature of T if there exists a row p in B^T such that if we eliminate from B^T the column corresponding to x_i, the remaining row p is a zero row of $B^{T-\{xi\}}$. Otherwise, x_i is called a removable feature (Piza-Davila et al, 2017).

Definition 7: A set $T = \{j_{k1},...,j_{ks}\} \subseteq J$ is called typical with respect to the M matrix and of the collection U if it is a testor and each feature $x_i \in T$ is a non-removable feature of T (Lazo-Cortes et al, 2001).

Proposition 1: The set of all typical testors of M is equal to the set of all typical testors from the basic matrix B (Lazo-Cortes et al, 2001).

Let $\psi^*(M)$ be the set of all typical testors of the matrix M.

According to proposition 1, to search over the set $\psi^*(M)$ it is very convenient to find the matrix B. Taking into account that B has equal or less number of rows than M, the efficiency of the algorithms should improve with B than with M (Lazo-Cortes et al, 2001).

B. Multi-layer feed-fordward back-propagation Neural Networks (FFBP)

FFBP neural networks have been positioned as the most used type of neural networks (Haykin et al, 2004). They have been applied in several fields like prediction as in (Weytjens et al, 2019), image recognition as in (Dosovitskiy et al, 2020), chemistry problems as in (Abdi-Khanghah et al, 2018) among others (Ghorbani et al, 2018), (Orrù et al, 2020), (Jia et al, 2020), (Auer et al, 2020). A FFBP is built by neurons, which are ordered by layers. The first layer is the input layer and the last one is known as the output layer. All the layers in between are called hidden layers.

Let Γ be the mapping function that relates for each neuron *i* a subset $\Gamma(i) \subseteq V$ which consists of all ancestors of the given neuron.

The subset $\Gamma^{-1}(i) \subseteq V$ consists of all predecessors of the *ith* neuron. Each neuron in a specific layer is connected with all the neurons of the next layer. The connection between the *ith* and *jth* neuron is characterised by the weight coefficient w_{ij} and the *ith* neuron by the bias coefficient θ_i.

The weight coefficient measures the degree of significance of the given connection in the neural network. The output value also called as activity of the *ith* neuron x_i holds that:

$$x_i = h(\xi_i)$$

$$\xi_i = \theta_i + \sum_{j \in \Gamma^{-1}(i)} w_{ij} x_j$$

where ξ_i is the potential of the *ith* neuron and function $h(\xi_i)$ is the transfer function or activation function. The supervised adaptation process varies the threshold coefficients θ_i and weight coefficients w_{ij} to minimize the objective function which relates computed and required output values. The back-propagation algorithm disperses the output error from the output layer through the hidden layers to the input layers so that the connection between the neurons can be recurrently calculated on training looking forward to minimize the loss function in each training iteration (Svozil et al, 1997).

C. Genetic Algorithms (GA)

Genetic Algorithms are heuristic based search approaches, specially applicable to optimization problems. The main reason that make them useful in practice is their flexibility over a wide range of problems (Kramer, 2017).

GAs begin with an initial population -set of initial points- and select a set of potential points to generate a new population. This operation is repeated until a stop criteria is reached. So it works by creating new populations of points (usually called chromosomes or individuals) by applying a set of genetic operators to the previous population of selected points (Diaz-Sanchez et al, 2011).

Classical genetic operators are selection, crossover and mutation. Crossover recombines the genetic information contained in the parents of two individuals picked from the selection set, and mutation applies modifications to certain values (alleles) of variables (genes) in the points (Mühlenbein, 199).

D. Database MNIST

The MNIST is a database of handwritten numbers. It is a widely used data set in machine learning. Handwriting recognition is a difficult problem and a good test for learning algorithms. The MNIST database has become a standard test. It collects 60,000 training images and 10,000 test images, taken from a previous database, simply called NIST1. These are black and white, normalized images centered at 28 pixels per side (LeCun et al, 1998a).

E. Proposed method

This section introduces an algorithm that searches for an optimal solution as a subset of features, in terms of efficiency over discrimination between classes of a collection U, based on a desired reduction of the total number of features. Depending on the density of the elements of each class l of the collection U we choose up to n objects of each class, this is for practical calculation purposes; we call this sub-collection U'. The first step is to obtain the basic matrix B over the collection $U' \subseteq U$. If needed, previous image prepossessing is recommended to reduce noise.

The second step is to remove all the columns of *B* where all the rows are *0*. We do this because this features do not help to discriminate. If a testor contains any of them, we can remove the feature or features since it will not generate a zero row by removing it. Furthermore, removing all this columns reduces the complexity on searching over the subspace of all possible combination of features.

Before introducing the Discriminator Construction Algorithm, we present its components. For this purpose we will divide them in two parts.

Let *P* be a set of *N* randomly chosen subsets of features of *B* of size *nxm* such that $x_k = \{x_{k1},\dots,x_{km}\} \in P$ with $x_{ki} \in \{0,1\}$, $k = 1,\dots,N$ and $i = 1,\dots,m$. x_k is called a *chromosome.*

1) Genetic Components

As in (Alba-Cabrera, 2000) we use Univariate Marginal Distribution Algorithm (UMDA). The fitness function is defined as:

$$f(x_k) = \alpha \frac{t(x_k)}{n} + (1-\alpha)\frac{p(x_k)}{\sum_{v=1}^{m} x_{kv}}$$

We define $T \subseteq x_k$ by eliminating all columns in x_k where $x_{ki} = 0$. Where, $t(x_k)$ is the number of non-zero rows in B^T, $p(x_k)$ is the number of typical features of B^T and α is a weighting coefficient. Recall that for any chromosome x_k, $0 \leq f(x_k) \leq 1$ where x_k is a typical testor if $f(x_k) = 1$. For initial and next generations we used a population size of 20 *chromosomes*.

2) FFBP Components

The components for FFBP selected for modeling testor performance during evolution has one input layer, two hidden layers and one output layer. Let m' be the total number of test cases we explicit define accuracy of a chromosome as $g(x_k) = \frac{\delta(x_k)}{m'}$.

Where $\delta(x_k)$ is the number of correct predictions of the trained neural network over a common test set. Recall that for any chromosome x_k, $0 \leq g(x_k) \leq 1$ where $f(x_k)$ approaching to *1* means a better performance on discriminating between the classes of U'.

Discriminator Construction Algorithm (DCA)

Finally, we are able to describe DCA algorithm. We first present data required and expected result. After we set up variable names to their representation in the algorithm.

Algorithm 1: Discriminator Construction Algorithm

```
Data: Basic Matrix B
Result: Collection of subsets T of the columns of B
P ⟵ generate initial population;
HP ⟵ empty list with the same size of P;
S ⟵ solution list;
t ⟵ set a threshold for typicality in chromosomes to start training neural networks;
O(x_k) ⟵ set objective function to f(x_k);
while not reached max number of iterations do
    while not reached number of solutions do
        for chromosome in P do
            fv ⟵ Calculate O(x_k) ;
            if fv > α and fv > all fv in HP then
                Append chromosome into HP with it fitness value;
                if HP.size > P.size then
                    Remove the element of HP with lowest fv;
                end
            end
            if fv > minFitnessValue then
                Append chromosome into S;
            end
        end
        f̄v ⟵ fitness value average;
        if f̄v > t then
            O(x_k) ⟵ set objective function to g(x_k);
        end
        BP ⟵ Join top scored elements of P with all elements in HP;
        MF ⟵ calculate marginal frequency over BP;
        P ⟵ Generate a new population with MF distribution;
        NMF ⟵ calculate marginal frequency over new P;
        if NMF = MF then
            Mutate P;
        end
    end
end
return S
```

Figure 1. Pseudo Code of proposed DCA Algorithm.

F. Experimental setup

For $U' \subseteq U$ we decided to choose *50* random objects of each class from training MNIST data set proceeding to binarize them to calculate *B*. With *B* calculated we removed all the zero columns in it, keeping record of the original indexes.

For the Genetic Components we settled α to *0.2* as we have higher probability of finding testors of large length (Alba-Cabrera et al, 2000). We set the maximum number of iterations to 100 and the solution length to 1 as stop conditions. We searched only for one discriminator. Mutation was performed over *1%* of the non-removed features in every generation.

For FFBP we set its topology as follows: the input layer and all the hidden layers has *relu (x)= max{0,x}* as activation function. The first hidden layer has *52* neurons, and the second one has *26*. The output layer has softmax as activation function which is defined as:

$$\sigma(z)_i = \frac{e^{zj}}{\sum_{k=1}^{K} e^{zk}}$$

Where *z* is a vector of dimension *k* and *j = 1,...,k*. In our case *z* has dimension *10* as we have that number of classes. Also we used Sparse Categorical Cross-entropy loss function, set batch size equal to *1/5* of the training samples and used *10* epochs which definitions are detailed in (Haykin, 2004), (Liu et al, 2016).

Since *DCA* requires a threshold to change fitness function from $f(x_k)$ to $g(x_k)$ we decide to set *t* as the maximum accuracy that the model can reach under a batch size of *1/5* of the training samples and the double epochs than used in $g(x_k)$ minus *0.04*. Once a solution was found it was translated its original indexes.

For assessment metrics let us call *discriminator* to one solution found by *DCA*. Following the same topology of $g(x_k)$ and as the length of the selected features of $discriminator$ turns the design of the neural network to be a unique model. This model was trained with Stratified K-Folds cross-validation for *5* folds over the train set each fold with 20 epochs whit the same batch size. For one-vs-all multi-class classification we also

used Stratified K-Folds cross-validation for model training over the same topology but with only two folds over the train set (Ramezan et al, 2019). We measured model's classification performance by accuracy, loss, multi-class precision. For calculate model precision we used three approaches: micro, macro and weighted. This types of precision contribute to present how samples and classes contribute to detailed view of model precision. We also measured multi-class log loss (Hazan et al, 2011), one-vs-all ROC curves and AUC scores (Wandishin et al, 2009), and one vs all precision vs recall metrics (Powers, 2020). Finally, we compared the *discriminator* error rate to some test error reported on the MNIST documentation.

Since the selection criteria was developed directly in evolution we choose the first solution that DCA report. All source code was implemented in *Python* language version 3.7.10 (Python Core Team, 2019) with the scikit-learn (SKlearn) library (Pedregosa et al., 2011) and Keras (Chollet, 2018).

III. RESULTS AND DISCUSSION

A total of *50* generations were needed to build a *discriminator* with *21.81%* of the total features. We present in Figure 2 some evolution steps with the corresponding feature selection for each generation.

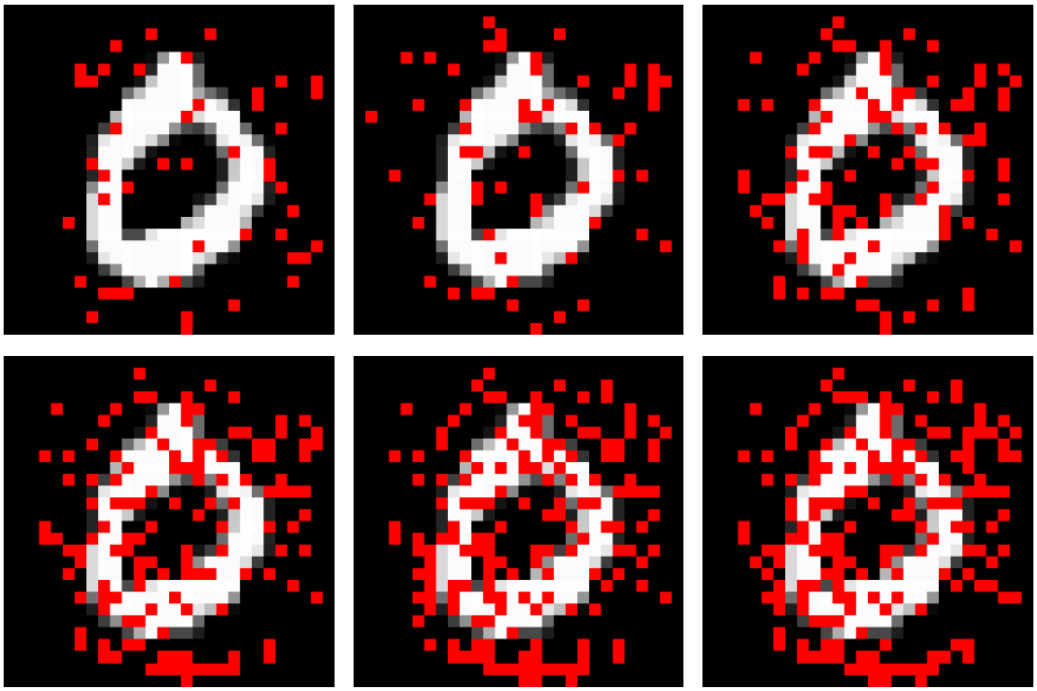

Figure 2. Generations 0, 10, 20, 30, 40, 50. With left right up down direction respectively.

A. Performance evaluation

The *discriminator* reported from DCA has a length of *171* features which represents a *21.81%* of the total amount of features. By reducing data sets matrices to those only containing the selected features and training a FFBP as described before we end up with the an accuracy on training of *99.65%* and a validation accuracy of *97.83%* on testing. In addition, the model reported a loss of *0.0191* on training and a loss of *0.081* on testing. Detailed view of this metrics versus epochs for each fold are presented in Figure 3.

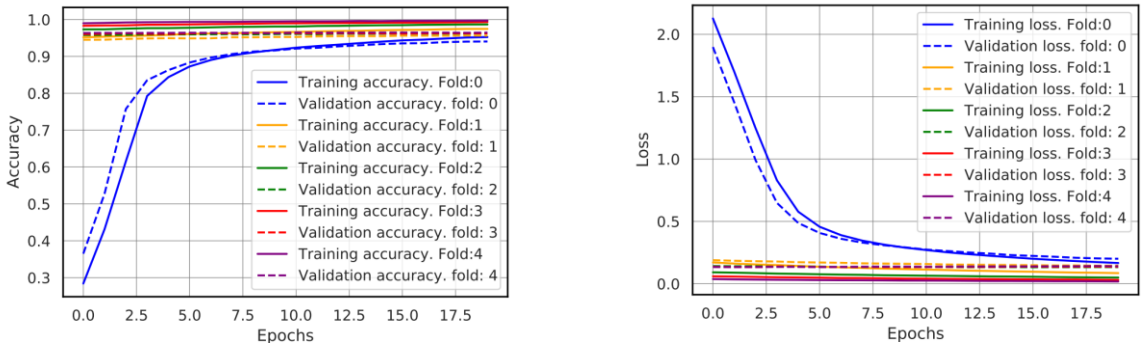

Figure 3. Graphic view of Accuracy vs Epochs (left) and Loss vs Epochs (rigth)

As we can see training and validation almost converge later on, even though at the beginning the curves are slightly different. This means any variation between the training and validation curves is going to be statistical rather than systematic so the model fits the data properly. From Figure 3 we can also mention that the model is not overfitted.

The confusion matrix in Figure 4 shows how the model performs on classifying between multiple classes. As seen below, almost the whole diagonal is almost highlighted. This describes a high rate between predicted labels and true labels. Results are presented in percentages.

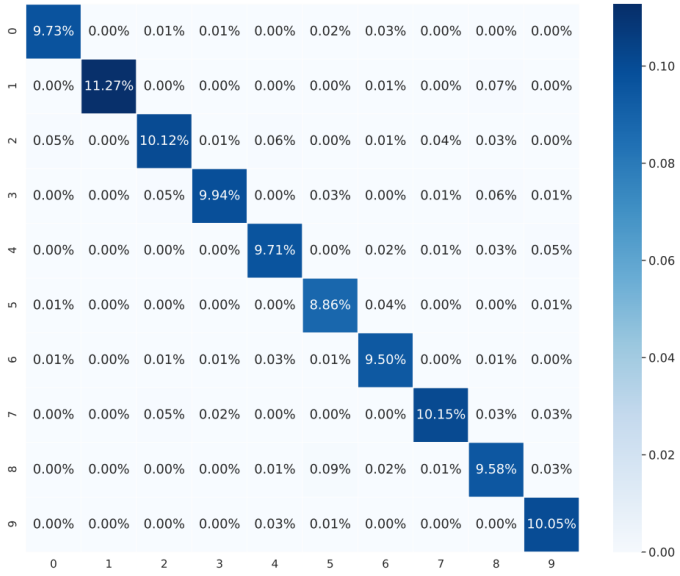

Figure 4. Confusion Matrix for *discriminator* in percentage.

The global prediction shows that in most of the time the model classifies correctly between all the classes but more than that it also shows in which classes it has problem. The errors in prediction of the model with the highest percetages are those made between 4 and 2 classes, 5 and 8 classes, and 8 and 3. However this error is not greater that 0.1% so it is not considered relevant. On the other hand, as mentionend the diagonal is highlighted which means in most cases every class is correctly identified. Furthermore, every class prediction rate is near 10% of the total amount, and since there are ten classes it also show balance between data and its prediction.

For micro-averaged precision we obtained a precision value of *97.15%*. For macro-averaged a precision value of *97.14%*. For weighed-average a precision value of *97.15%*. Therefore the model is consistent and predicts accurate for distinct classes.

To show in more detail this fact we plotted the ROC curves for all one-vs-all with the corresponding AUC. Furthermore, precision versus recall plot for one-vs-all shows another perspective of the reported precision values.

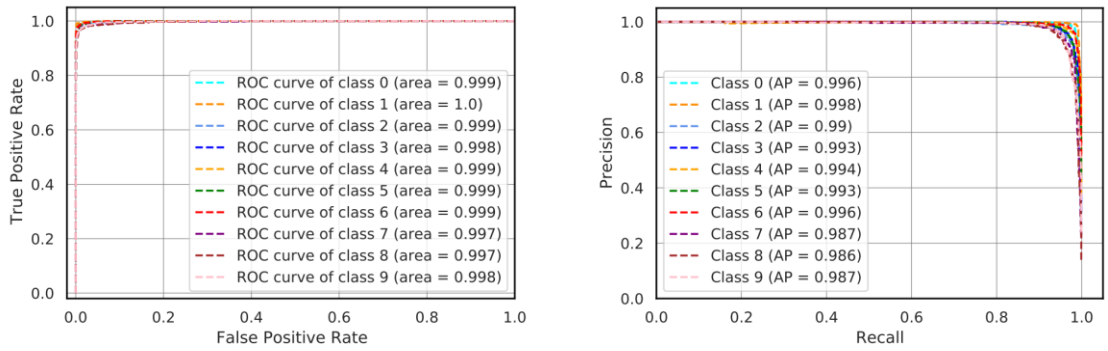

Figure 5. ROC curve one vs-all (left) and Precision versus Recall (rigth)

Figure 5 shows an area under the curve closer to 1 for every one versus all cases. This means that the model has an strong performance in distinghuis between all classes. This

scenario allows us to explicit interpret that those points chosen by *discriminator* are good enough to be able to clearly classify between all classes. The presicion measures the ability of the model to predict each of the classes; since the higher precision and recall scores, the better performance of the model, with an average presicion closer to 1, the model presented can distinguish between all classes with precision.

Finally, we calculate multi-class log loss with a value of 0.2240 which means the model assigns a high probability for each class to predict correctly. For this value in particular the fact of having 10 classes makes the reported value a positive metric.

B. State of art based comparison

Despite the fact that there is no other *discriminator* in the literature, we can evaluate the performance of the model against the error rate reported in the MNIST documentation.

Table 1. MNIST documentation comparison to DCA + FFBP.

Method	Features(%)	Error(%)
K-nearest-neighbors, Euclidean (L2)	100	5.00
Boosted stumps	100	7.70
40 PCA + quadratic classifier	100	3.30
SVM, Gaussian Kernel	100	1.40
3-layer NN, 300+100 hidden units	100	3.05
Convolutional net, cross-entropy	100	0.60
DCA + FFBP	**21.81**	**2.18**

From table 1, it is possible to notice that the *discriminator* is the only model that used less features of the data set. Due to the state of the art-based comparisons we are going to focus on test error, it should be noted that the error rate is clearly in the range of most of them. Therefore, the proposed model based on DCA could be considered less complex and faster in computational time, since it selects some features of the entire set.

We would like to empjasize that DCA + FFPB method reported an error rate of only 2.18% while using 21.81% of the features. Other models such as KNN or even SVM could not reach this error rates even with all the features. However, it would be desired to compare how all this models perform with feature selection of the *discriminator* so then stablish if those selected features preserve error rates of what is presented in literature. Eventhough, for FFPB we see that this features are enough for truly classify and preserve accuracy and errors rate of a similar model.

IV. CONCLUSIONS AND FUTURE WORK

This work proposed an algorithm (DCA) to build a discriminator for a group of classes. With the knowledge of testors we were able to establish a starting point for an intelligent search. With evolution strategy we searched for some features with properties of testors and typicality. Once this features were found, we changed the fitness value to reach the accuracy goal for this reduced featured set. In this point FFBP play with their accuracy on predicting played the role of fitness function. With this strategy we built a discriminator with *21.81%* of the total amount of features. Taking in consideration all calculated assessment metrics and the interpretation of them, with the discriminator we can build a model that predicts with a precision over *97%* and it can distinguish between all the classes. Therefore we conclude that DCA algorithm is able to build a reduced features discriminator for training and testing over MNIST data set.

Furthermore, experimentation is showing us that the computational cost represented by calculating testors is totally worth it, the substantial reduction in the number of variables that can be used confirms that not all the information is necessary when classifying objects. Also, the successful coordination of algorithms, evolutionary and neural networks is also important, which is manifested in the coherence of the classification results. Finally, interleaving two optimization functions, one for the selection of a discriminator and the other for training the network, shed light on the way forward to obtain robust discriminators with a substantially lower number of characteristics.

As future work, we propose (1) to use typical testors properties to look for a minimum-length optimal discriminator, (2) to explore other feature selection techniques to compare their ability to preserve accuracy over prediction models, as well as (3) to reduce complexity over calculations to search over bigger spaces.

REFERENCIAS BIBLIOGRÁFICAS

Abdi-Khanghah, M., Bemani, A., Naserzadeh, Z., & Zhang, Z. (2018). Prediction of solubility of N-alkanes in supercritical CO2 using RBF-ANN and MLP-ANN. *Journal of CO2 Utilization, 25, 108-119*.

Alba-Cabrera, E., Godoy-Calderon, S., & Ibarra-Fiallo, J. (2016). Generating synthetic test matrices as a benchmark for the computational behavior of typical testor-finding algorithms. *Pattern Recognition Letters*, *80*, 46-51.

Alba-Cabrera, E., Ibarra-Fiallo, J., Godoy-Calderon, S., & Cervantes-Alonso, F. (2014, November). YYC: A fast performance incremental algorithm for finding typical testors. In Iberoamerican Congress on Pattern Recognition (pp. 416-423). Springer, Cham.

Alba-Cabrera, E., Santana, R., Ochoa-Rodriguez, A., & Lazo-Cortes, M. (2000). Finding typical testors by using an evolutionary strategy. In *Proceedings of the 5th Ibero American Symposium on Pattern Recognition* (p. 267).

Auer, A., Strauss, M. T., Strauss, S., & Jungmann, R. (2020). nanoTRON: a Picasso module for MLP-based classification of super-resolution data. *Bioinformatics, 36(11), 3620-3622*.

Asaithambi, A., & Valev, V. (2004). Construction of all non-reducible descriptors. *Pattern Recognition, 37*(9), 1817-1823.

Alganza, Y. S., & Porrata, A. P. (2003). LEX: A new algorithm for calculating typical testors. *Revista Ciencias Matematicas, Cuba, 21*(1), 85-95.

Becht, E., McInnes, L., Healy, J., Dutertre, C. A., Kwok, I. W., Ng, L. G., ... & Newell, E. W. (2019). Dimensionality reduction for visualizing single-cell data using UMAP. *Nature biotechnology*, *37*(1), 38-44.

Carrasco-Ochoa, J. A., & Martínez-Trinidad, J. F. (2004, August). Feature selection for natural disaster texts classification using testors. In *International Conference on Intelligent Data Engineering and Automated Learning* (pp. 424-429). Springer, Berlin, Heidelberg.

Chollet, F. (2018). Keras: The python deep learning library. *Astrophysics Source Code Library*.

Diaz-Sanchez, G., Piza-Davila, I., Sanchez-Diaz, G., Mora-Gonzalez, M., Reyes-Cardenas, O., Cardenas-Tristan, A., & Aguirre-Salado, C. (2011, September). Typical testors generation based on an evolutionary algorithm. In *International Conference on Intelligent Data Engineering and Automated Learning* (pp. 58-65). Springer, Berlin, Heidelberg.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.

Fernandez, D., Gonzalez, C., Mozos, D., & Lopez, S. (2019). FPGA implementation of the principal component analysis algorithm for dimensionality reduction of hyperspectral images. *Journal of Real-Time Image Processing, 16*(5), 1395-1406.

Gallegos, A., Torres, D., Álvarez, F., & Soto, A. T. (2016). Feature Subset Selection and Typical Testors Applied to Breast Cancer Cells. *Res. Comput. Sci., 121*, 151-163.

Ghorbani, M. A., Deo, R. C., Karimi, V., Yaseen, Z. M., & Terzi, O. (2018). Implementation of a hybrid MLP-FFA model for water level prediction of Lake Egirdir, Turkey. *Stochastic Environmental Research and Risk Assessment, 32(6), 1683-1697.*

Haykin, S., & Network, N. (2004). A comprehensive foundation. *Neural networks, 2*(2004), 41.

Hazan, E., & Kale, S. (2011, December). Newtron: an Efficient Bandit algorithm for Online Multiclass Prediction. In *NIPS* (Vol. 11, pp. 891-899).

Jia, X., Cao, Y., O'Connor, D., Zhu, J., Tsang, D. C., Zou, B., & Hou, D. (2021). Mapping soil pollution by using drone image recognition and machine learning at an arsenic-contaminated agricultural field. *Environmental Pollution, 270, 116281*.

Kramer, O. (2017). Genetic algorithms. In *Genetic algorithm essentials* (pp. 11-19). Springer, Cham.

Lazo-Cortes, M. S., Martínez-Trinidad, J. F., Carrasco-Ochoa, J. A., & Sanchez-Diaz, G. (2015). On the relation between rough set reducts and typical testors. *Information Sciences, 294*, 152-163.

Lazo-Cortes, M., Ruiz-Shulcloper, J., & Alba-Cabrera, E. (2001). An overview of the evolution of the concept of testor. *Pattern recognition, 34*(4), 753-762.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P., "Gradient-based learning applied to document recognition." *Proceedings of the IEEE, 86*(11):2278-2324, November 1998.

Lias-Rodriguez, A., & Sanchez-Diaz, G. (2013). An algorithm for computing typical testors based on elimination of gaps and reduction of columns. *International Journal of Pattern Recognition and Artificial Intelligence, 27*(08), 1350022.

Liu, H., & Motoda, H. (Eds.). (2007). *Computational methods of feature selection*. CRC Press.

Liu, W., Wen, Y., Yu, Z., & Yang, M. (2016, June). Large-margin softmax loss for convolutional neural networks. In ICML (Vol. 2, No. 3, p. 7).

Mafarja, M. M., & Mirjalili, S. (2019). Hybrid binary ant lion optimizer with rough set and approximate entropy reducts for feature selection. *Soft Computing, 23*(15), 6249-6265.

Mühlenbein, H., Mahnig, T., & Rodriguez, A. O. (1999). Schemata, distributions and graphical models in evolutionary optimization. *Journal of Heuristics, 5*(2), 215-247.

Orrù, P. F., Zoccheddu, A., Sassu, L., Mattia, C., Cozza, R., & Arena, S. (2020). Machine learning approach using MLP and SVM algorithms for the fault prediction of a centrifugal pump in *the oil and gas industry. Sustainability, 12(11), 4776*.

Ortíz-Posadas, M. R., Martínez-Trinidad, J. F., & Ruiz-Shulcloper, J. (1996). A new approach to differential diagnosis of diseases. *International journal of bio-medical computing, 40*(3), 179-185.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). *Scikit-learn: Machine learning in Python. the Journal of machine Learning research, 12*, 2825-2830.

Piza-Davila, I., Sanchez-Diaz, G., Lazo-Cortes, M. S., & Rizo-Dominguez, L. (2017). A CUDA-based hill-climbing algorithm to find irreducible testors from a training matrix. *Pattern Recognition Letters, 95*, 22-28.

Piza-Dávila, I., Sánchez-Díaz, G., Lazo-Cortés, M. S., & Villalón-Turrubiates, I. (2020). An Algorithm for Computing Minimum-Length Irreducible Testors. *IEEE Access, 8*, 56312-56320.

Piza-Davila, I., Sanchez-Diaz, G., Aguirre-Salado, C. A., & Lazo-Cortes, M. S. (2015). A parallel hill-climbing algorithm to generate a subset of irreducible testors. *Applied Intelligence, 42*(4), 622-641.

Pons-Porrata, A., Gil-García, R., & Berlanga-Llavori, R. (2007, November). Using typical testors for feature selection in text categorization. In *Iberoamerican Congress on Pattern Recognition* (pp. 643-652). Springer, Berlin, Heidelberg.

Powers, D. M. (2020). Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061*.

Python Core Team. (2019). *Python 3.6.9: A dynamic, open source programming language.* Python Software Foundation. Available: https://www.python.org.

Ramezan, C., Warner, T., & Maxwell, A. (2019). Evaluation of sampling and cross-validation tuning strategies for regional-scale machine learning classification. *Remote Sensing, 11*(2), 185.

Ruiz-Shulcloper, J., Soto, A., & Fuentes, A. (1980). A characterization of the typical testor concept in terms of a notable set of columns. *Rev Cien Mat (in Spanish), 1*(2), 123-134.

Sanchez-Diaz, G., Diaz-Sanchez, G., Mora-Gonzalez, M., Piza-Davila, I., Aguirre-Salado, C. A., Huerta-Cuellar, G., ... & Cardenas-Tristan, A. (2014). An evolutionary algorithm with acceleration operator to generate a subset of typical testors. *Pattern Recognition Letters, 41*, 34-42.

Sayed, G. I., Hassanien, A. E., & Azar, A. T. (2019). Feature selection via a novel chaotic crow search algorithm. *Neural computing and applications, 31*(1), 171-188.

Saxena, A., Saxena, K., & Goyal, J. (2019). Hybrid technique based on DBSCAN for selection of improved features for intrusion detection system. In *Emerging Trends in Expert Applications and Security* (pp. 365-377). Springer, Singapore.

Singh, D., & Singh, B. (2019). Hybridization of feature selection and feature weighting for high dimensional data. *Applied Intelligence, 49*(4), 1580-1596.

Svozil, D., Kvasnicka, V., & Pospichal, J. (1997). Introduction to multi-layer feed-forward neural networks. *Chemometrics and intelligent laboratory systems, 39*(1), 43-62.

Valev, V., & Asaithambi, A. (2003, October). On computational complexity of non-reducible descriptors. In *Proceedings Fifth IEEE Workshop on Mobile Computing Systems and Applications* (pp. 208-211). IEEE.

Valev, V., & Zhuravlev, J. I. (1991). Integer-valued problems of transforming the training tables in k-valued code in pattern recognition problems. *Pattern Recognition, 24*(4), 283-288.

Wandishin, M. S., & Mullen, S. J. (2009). Multiclass ROC analysis. *Weather and Forecasting, 24*(2), 530-547.

Weytjens, H., Lohmann, E., & Kleinsteuber, M. (2019). Cash flow prediction: MLP and LSTM compared to ARIMA and Prophet. *Electronic Commerce Research, 1-21.*

Wang, M., Wu, C., Wang, L., Xiang, D., & Huang, X. (2019). A feature selection approach for hyperspectral image based on modified ant lion optimizer. *Knowledge-Based Systems, 168*, 39-48.

Zhou, H., Zhang, Y., Zhang, Y., & Liu, H. (2019). Feature selection based on conditional mutual information: minimum conditional relevance and minimum conditional redundancy. *Applied Intelligence, 49*(3), 883-896.