ISBN: 978-981-18-5852-9
WCSE 2022 Spring Event: 2022 9th International Conference on Industrial Engineering and Applications
doi: 10.18178/wcse.2022.04.030

Diagnostic of Autism Spectrum Disorder

Elena Acevedo [1+], Dinora Orantes [2] and Antonio Acevedo [1]

[1] Instituto Politécnico Nacional. ESIME Zacatenco, Mexico City, Mexico

[2] Instituto Politécnico Nacional, Centro de Investigación en Computación, Mexico City, Mexico

Abstract. According to the World Health Organization, Autism Spectrum Disorder affects 1 in 160 children, a disturbance of neurodevelopment characterized by symptoms such as difficulties in interaction and social communication, narrow interests, and repetitive behaviors. In this work, we diagnose Autism Spectrum Disorder using Machine Learning (ML) tools through the supervised training of Multi-Layer Perceptron and KNN classifiers. The validation algorithms were Hold-Out and K-Fold Cross Validation for both methods. The precision with the Multi-Layer Perceptron was 100% with Hold-Out and K-Fold Cross Validation, and with the KNN classifier, the accuracy was 91% and 87%, respectively.

Keywords: Artificial intelligence, Machine learning, Diagnosis, autism spectrum disorder

1. Introduction

Autism spectrum disorder (ASD) is a developmental disorder that affects communication and behavior. Although autism can be diagnosed at any age, it is said to be a developmental disorder because symptoms generally appear in the first two years of life [1]. In addition, it is called a spectrum disorder because people with ASD can have a range of symptoms [2], and it is seen as a spectrum that can range from very mild to severe. Nevertheless, many (but not all) individuals with ASD require lifelong support of some kind [3].

In Mexico, it is estimated that 1 in 115 children have Autism Spectrum Disorder (ASD) [4]. However, some institutions, such as the Mexican Institute of Social Security (IMSS) and the Secretariat of Health and Assistance (SSA), do not have the devices for screening and/or detecting ASD [5], regardless of the level of medical care. Only 19 of the 32 states of the Mexican Republic have any medical devices to carry out the diagnosis of ASD, but the number of them is not enough. Therefore, an intelligent system is proposed in this work that allows diagnosing ASD through parameters obtained from questions.

We consider the diagnosis problem a classification task; therefore, we applied two classifiers to achieve the diagnosis.

2. Related Work

Vaishali and Sasikala [6] used some classifiers included in the WEKA software. They analyze the same dataset that we used in this work. A Swarm intelligence-based single-objective binary firefly feature selection wrapper was applied, and they found that the classifiers work better with only ten features from the 21 original features. The validation algorithm was 10-Fold Cross-Validation. The accuracy results were: Naïve Bayes with 95.55%, J48 with 92.12%, Support Vector Machine with 99.66%, K-Nearest Neighbors with 87.67, and Multilayer Perceptron with 99.66%.

A Feedforward Neural Network technique was applied to detect ASD [7]. The authors used the Modified Checklist for Autism in Toddlers (MCHAT) dataset with 14,995 records. The result was a total correct classification percentage in the range of 99.56-99.76.

Wang et al. [8] applied functional connectivity (FC)-based algorithm for classifying autism and control using support vector machine-recursive feature elimination (SVM-RFE). First, they performed feature selection with the FC and used SVM-RFE as a classifier. Then, they applied two validation algorithms: 4-Fold Cross-Validation and Leave-One-Out. As a result, they report the following results: for 4-Fold Cross-

[+] Corresponding author. Tel.: +52 5534077544; fax: -----.
E-mail address: eacevedo@ipn.mx.

Validation, an accuracy of 90.60% (sensitivity 90.62%, specificity 90.58%), and for Leave-One-Out a precision in the range of 75.00%–95.23%.

A mobile application was proposed [9], which provides the user interface capturing questionnaire data, then a Convolutional Neural Network performs the screening. The authors report an accuracy of 98.88% with the 10-Fold Cross-Validation algorithm.

Wingfield et al. [10] also proposed a mobile application. The dataset was a mixture of two datasets: Diagnostic and Statistical Manual of Mental Disorders 41 and M-CHAT, and modified to include cultural considerations. 10-Fold Cross-Validation was applied. They used some classifiers from WEKA: Adaptive Boosting, Decision Tree (J48), Multi-Layer Perceptron, Naïve Bayes, Ruled-based model, Random Forest, Sequential minimal optimization, and Support Vector Machine. The range of the accuracy was 88%-93%.

3. Methods and Materials

In this section, the classification algorithms are presented together with the validation methods. Also, the dataset used in this work is described.

3.1. K-Nearest Neighbors Algorithm

Contributions to the congress are welcome from throughout the world. Manuscripts may be submitted to

The *k*-nearest neighbor algorithm [11] is a technique for classifying objects based on closest training examples in the problem space. The *k*-nearest neighbor algorithm is among the simplest of all machine learning algorithms: similar things exist nearby and are close to each other. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its *k* nearest neighbors (*k* is a positive integer). If $k = 1$, then the object is simply set to the class of its nearest neighbor. Now, the algorithm will be described.

It will be assumed that the instance domain, X, is endowed with a metric function p. This is $p: X \times X \rightarrow \Re$ is a function that returns the distance between any two elements of X. For example, if $X = \Re^{d,}$ then p can be the Euclidean distance, as it is shown in Eq. (1).

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

Let $S = (x_1, y_1), \ldots, (x_m, y_m)$ be a sequence of training examples. For each $x \in X$, let $\pi_1(x), \ldots, \pi_m(x)$ be a reordering of $\{1, . . . , m\}$ according to their distance to x, $p(x, x_i)$. That is, for all $i < m$,
$p(x, x_{\pi_i(x)}) \leq p(x, x_{\pi_{i+1}(x)})$

For a number k, the k-NN rule for binary classification is defined as follows:

input: a training sample $S = (x_1, y_1), \ldots, (x_m, y_m)$

output: for every point $x \in X$, return the majority label among $\{y_{\pi i(x)}: i \leq k\}$

3.2. Artificial Neural Networks

An Artificial Neural Network is a set of single neurons interconnected to simulate the brain of the human architecture and function.

ANN has the skill to generalize the data from previous examples to find and converge the new ones [12].

ANN models can be implemented in software to perform classification, diagnosis, and pattern recognition tasks.

Multi-Layer Perceptron (MLP) is feedforward and supervised neural network. The architecture of the Multi-Layer perceptron is characterized by having its neurons grouped in layers of different levels. Each of the layers is made up of a set of neurons, and there are three different types of layers: the input layer, the hidden layers, and the output layer, as shown in Figure 1.

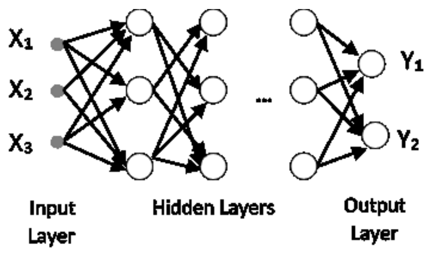

Fig. 1: The basic architecture of a Multi-Layer Perceptron.

From Figure 1, it can be observed that the connections of the Multi-Layer perceptron are always directed forward, which means the neurons of one layer are connected with the neurons of the next layer; hence they are also called forward-fed networks. The connections between neurons are associated with a real number called the connection weight. All neurons in the network are also associated with a threshold.

For the propagation of the input patterns for the Multi-Layer Perceptron, the following must be defined:

C: No. of layers of the MLP. Therefore, there are C-2 hidden layers.

nc: neurons in layer c, for $c = 0, 1, 2, \ldots, C-1$.

$\boldsymbol{W^c_{ij}}$ is the weight matrix associated with the connections from layer c-1 to the layer c for $c = 1, 2, \ldots, C-2$, where $\boldsymbol{w^c_{ij}}$ is the weight of the neuron connection i of layer c-1 to the neuron j of layer c.

$\boldsymbol{U^c = (u^c_i)}$ s the vector of thresholds of the neurons of the layer c for $c = 1, 2, \ldots, C-1$.

$\boldsymbol{a^c_i}$ is the activation of the neuron i of layer c.

The activations of neurons are activated in the following way.

- Activation of input layer neurons (a^0_i). The neurons of the input layer are responsible for transmitting signals received from outside to the network. Therefore, Eq.(2) shows the activations of the input layer.

$$a^0_i = x_i \text{ para } i = 1, 2, \ldots, n_1 \tag{2}$$

where $X = (x_1, x_2, \ldots, x_3)$ represents the pattern from the input of the network.

- Activation of hidden and output layers neurons c (a^c_i).The hidden neurons of the network process information received by applying an activation function f to the sum of the products of the activations received by their corresponding weights as Eq.(3) shows:

$$a^c_i = f\left(\sum_{j=1}^{n_{c-1}} a^{c-1}_j w^c_{ji} + u^c_i\right) \tag{3}$$

for $i = 1, 2, \ldots, n_c$ and $c = 1, 2, \ldots, C-2$. Where a^{c-1}_j are the activations of the neuron of layer c-1.

The goal of neural network training is to minimize the cost function by finding the appropriate weights for the network connections, trying to ensure the best generalization. The discovery of these weights is performed by an algorithm called backpropagation.

In the feedforward process, the activations are propagated forward. In the Backpropagation algorithm, the error made by the network is propagated backward while the network weights are updated. The optimizer is in charge of generating better and better weights. Its operation is based on calculating the gradient of the cost function (partial derivative) for each weight (parameter/dimension) of the network. Since the error must be minimized, each weight is modified in the (negative) direction of the gradient. The update of the parameters (weights and thresholds) is represented by Eq.(4).

$$w(n) = w(n-1) - \alpha \frac{\partial e(n)}{\partial w} \tag{4}$$

where $e(n)$ is the error for pattern n and α is the learning ratio or rate, a parameter that influences the magnitude of the displacement on the surface of the error.

3.3. Validation Algorithms

- Hold-Out

The dataset is partitioned into two subsets. One of them is used to train the classifier, and the other is for testing. The most common percentage of partitioning is 80% for training and 20% for testing.

- K-Fold Cross-Validation

It consists of sectioning the dataset into *K* parts. One part *k* is for the test set and the other part *k*-1 for the training set, the training set is what the model is trained with, and a part is left out for testing; this is repeated *k* times while changing the test part one by one until all tests have been performed the *k* parts. Figure 2 illustrates an example with *k* = 5.

Iteration					
1	Test				
2		Test			
3			Test		
4				Test	
5					Test

Fig. 2: If *k* = 5, the dataset is partitioned into five sections, the algorithm iterates five times. A different block is selected as a testing dataset in each iteration, and the other four blocks are used for training.

3.4. Dataset

The name of the dataset is Autistic Spectrum Disorder Screening Data for Children (https://archive.ics.uci.edu/ml/datasets/Autistic+Spectrum+Disorder+Screening+Data+for+Children++). There are 292 records corresponding to 292 children between 4 and 11 years old. Each record has 21 attributes: age, gender, ethnicity, country of residence, and others. The values of the attributes are the type of binary, categorical, string, and integer. All the features were converted to integer numbers.

4. The Proposed Architecture

The classifiers were developed with Python 3.7. language, using Anaconda Navigator and Jupyter Notebook, and Tensorflow library. It was implemented in a Laptop Omen core i7, 32GB-RAM.

The dataset was unbalanced, then the function *SMOTE* from Python was applied to balance the data; therefore, we obtained better results.

The parameters for the K-Nearest Neighbors classifier were a *k* = 16 and the Euclidian distance.

In the case of the Neural Network, after performing several tests, we found the optimal number of hidden layers was five. Therefore, the number of neurons for each layer is shown in Table 1.

Table 1: The title should be in Capitalization of the first word

Hidden layer	**Number of neurons**
1	500
2	400
3	300
4	350
5	750

The activation of the neurons of the hidden layer is *ReLU*, and the activation of the output layer is *Sigmoid*. The optimization function was Adam. In this case, 300 epochs and a batch size of 100 were applied.

5. Results

The Hold-Out (HO) and K-Fold Cross Validation algorithms were applied to validate the results of both classifiers. We used the 80-20 percentage for HO. Also, we obtained the following metrics: Confusion Matrix, Accuracy, Specificity, Sensitivity, and Precision. First, we present the results for the artificial neural network.

In Table 2, we can observe that the neural network confused just one record. These results support the metrics shown in Table 3.

Table 2: The Confusion Matrix Results for the Proposed Artificial Neural Network, with Hold-Out Validation Algorithm

True Negatives	***False positives***
33	0
False Negatives	***True positives***
1	27

Table 3: The Metrics for the Proposed Artificial Neural Network

Accuracy	98%
Sensitivity	96%
Specificity	100%
Precision	100%

Figure 3 shows the ROC curve. In Figure 3, it can be observed that the curve is almost perfect; the only false negative distorts it a bit.

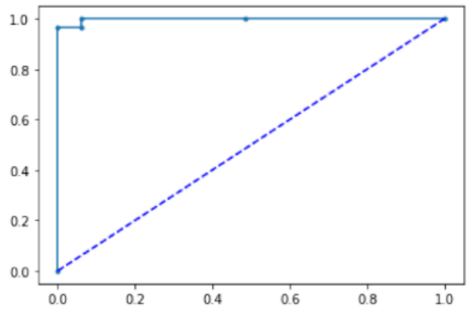

Fig. 3: The ROC curve for a Multi-Layer Perceptron with Hold-Out validation.

Now, in Table 4, the neural network results from the confusion matrix are shown. These results are from the K-Fold Cross-Validation algorithm with $K = 3$.

Table 4: The Confusion Matrix Results for the Proposed Artificial Neural Network with the K-Fold Cross-Validation Algorithm

Iterations	1	2	3
True Positives	50	50	49
True Negatives	51	50	50
False Positives	0	0	0
False Negatives	0	1	1

From Table 4, it can be observed that the Multi-Layer Perceptron makes a mistake confusing a record as another class. This behavior can be seen graphically in Figure 4. Table 5 shows the metrics calculated for this method.

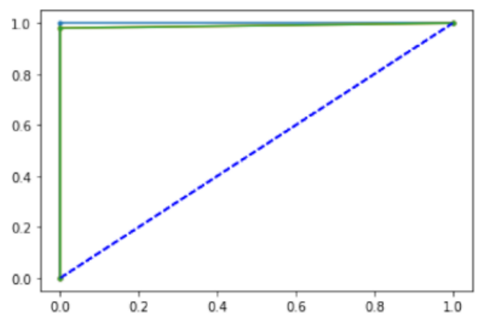

Fig. 4: The ROC curve for a Multi-Layer Perceptron with K-Fold Cross-Validation

Figure 4 shows a correct classification (full blue line) which corresponds to the first iteration of the algorithm in Table 4, where there are not false negative nor false positive classifications.

Table 5: The Metrics for the Proposed Artificial Neural Network

Accuracy	99%
Sensitivity	100%
Specificity	98%
Precision	100%

Now, the results for the KNN classifier are presented, in both cases, for Hold-Out and K-Fold Cross-Validation algorithms. The first result is shown in Table 6.

Table 6: The Confusion Matrix Results for the Proposed KNN Classifier, with Hold-Out Validation Algorithm

True Negatives	***False positives***
30	3
False Negatives	***True positives***
0	28

In this case, from Table 6, the classifier fails three times when it confuses three records as positive.

Table 7 shows the other metrics. We can observe that the confusion matrix is related to these metrics. In Table 6, there are no false negatives; the sensitivity is 100% in Table 7.

Table 7: The metrics for the proposed KNN with K = 16.

Accuracy	95%
Sensitivity	100%
Specificity	91%
Precision	91%

Because of the behavior of the KNN method, the ROC curve (Figure 5) is a little slanted to the right.

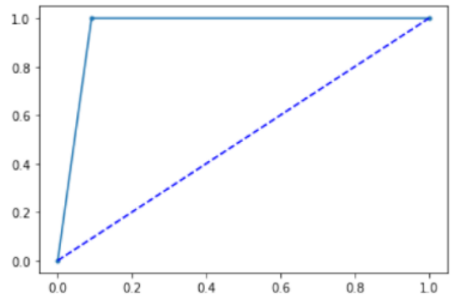

Fig. 5: The ROC curve for the KNN classifier with Hold-Out.

Table 8 shows the confusion matrix. From Table 8, it can be observed that there are false positives and negatives in all the iterations. However, only in the first iteration, one false positive and no false negatives appear. This result is reflected in Figure 6, where the full blue line shows the best behavior. The other lines are more slanted to the right.

Table 8: The Confusion Matrix Results for the Proposed KNN Classifier with the K-Fold Cross-Validation Algorithm

Iterations	1	2	3	4	5	6	7
True Positives	22	20	19	20	20	21	19
True Negatives	21	18	20	18	17	19	19
False Positives	1	4	2	4	4	2	2
False Negatives	0	1	2	1	2	1	3

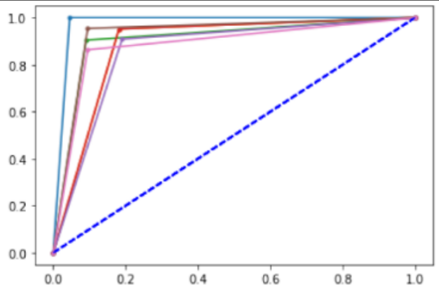

Fig. 6: The ROC curve for the KNN classifier with Hold-Out.

Table 9 shows the quantitative metrics for the KNN classifier when we applied K-Fold Cross-Validation algorithm. From Table 8, we can observe that there are more false positives than false negatives, that is the reason that in Table 9, sensitivity is less that specificity.

Table 9: The Metrics for the KNN Method

Accuracy	90%
Sensitivity	87%
Specificity	93%
Precision	87%

Table 10 compares the accuracy results from the related work with the results of the proposed classifiers.

From Table 10, it can be observed that our proposal obtained the 100% of classification. It has to be highlighted that the work of reference [6] used the same dataset that this work. Our proposal shows a classification of 100%. Vaishali and Sasikala did not mention the architecture of the Neural Network that they proposed in their work; however, our architecture shows a perfect classification.

6. Conclusions

The obtained results suggest that Machine Learning algorithms are suitable for diagnosing autism spectrum disorder. A Multi-Layer Perceptron (ML) and K-Nearest Neighbors algorithms were applied in this work, and the results of ML showed the best percentage of classification. The results of our proposal are competitive with those in State-of-Art. The percentage of classification obtained from the reference [6] and this work indicates that Multi-Layer Perceptron algorithms are the best option for diagnosing autism spectrum disorder.

7. Acknowledgements

The authors would like to thank the Instituto Politécnico Nacional (COFAA, EDI, and SIP), the CONACyT, and SNI for their support to develop this work.

8. References

[1] National Institute of Mental Health https://www.nimh.nih.gov/health/topics/autism-spectrum-disorders-asd

[2] https://medlineplus.gov/autismspectrumdisorder.html

[3] C. Lord, M. Elsabbagh, G. Baird, J. Veenstra-Vanderweele. Autism spectrum disorder, *Lancet*, 2018, **392**: 508–520,.

[4] Z. Ramírez, N. López *et al*. Trayectoria de atención y demora diagnóstica en casos con trastorno del espectro autista. Technical report, Instituto Nacional de Psiquiatría Ramón de la Fuente Muñiz, 2017.

[5] Z. Ramírez, N. López *et al*. Infraestructura disponible para la atención de los trastornos del espectro autista en el sistema nacional de salud. Technical report, Instituto Nacional de Psiquiatría Ramón de la Fuente Muñiz, 2018.

[6] R. Vaishali, R. Sasikala. A machine learning based approach to classify autism with optimum behavior sets, *International Journal of Engineering & Technology,* 2018, **7** (4), 4216-4219.

[7] L. Achenie, A. Scarpa, R. Factor, T. Wang, D. Robins, D. McCrickard. A Machine Learning Strategy for Autism Screening in Toddlers, *J Dev Behav Pediatr*, 2019, **40**:369–376.

[8] C. Wang, Z. Xiao, J. Wu. Functional connectivity-based classification of autism and control using SVM-RFECV on rs-fMRI data, *Physica Medica*, 2019, **65**: 99–105.

[9] S. Shahamiri, F. Thabtah. Autism AI: a New Autism Screening System Based on Artificial Intelligence, *Cognitive Computation*, 2020, **12**:766–777.

[10] B. Wingfield, S. Miller, P. Yogarajah, D. Kerr, B. Gardiner. A predictive model for paediatric autism screening, *Health Informatics Journal*, 2020, **26**(4): 2538– 2553.

[11] S. Shai and B. Shai. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014, 258-259.

[12] F. Chollet. *Deep Learning with Python*, New York: Manning Shelter Island, 2018.