UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingeniería

A Hybrid Method for Characters Recognition Using Ant Colony Feature Selection, KNN, and Reducts

Cristhian Iván Cola Pilicita

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 2022

UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

HOJA DE CALIFICACIÓN
DE TRABAJO DE FIN DE CARRERA

A Hybrid Method for Characters Recognition Using Ant Colony Feature Selection, KNN, and Reducts

Cristhian Iván Cola Pilicita

Nombre del profesor, Título académico	**Noel Pérez Pérez, Phd.**
Nombre del profesor, Título académico	**Julio Ibarra-Fiallo, Msc.**

Quito, 6 de mayo de 2022

© DERECHOS DE AUTOR

Nombres y apellidos: Cristhian Iván Cola Pilicita

Código: 00140858

Cédula de identidad: 1727008466

Lugar y fecha: Quito, 6 de mayo de 2022

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UNPUBLISHED DOCUMENT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

Este trabajo aborda el desarrollo de un método mejorado para la selección de características de grandes conjuntos de datos y una estrategia para disminuir la complejidad de su clasificación. En el campo de la clasificación supervisada, el uso de conjuntos de datos muy grandes implican nuevos desafíos para los investigadores debido al aumento del tiempo de procesamiento y recursos computacionales. Para afrontar este problema el método propuesto hace uso del paradigma divide y vencerás, y la aplicación de un método embebido con una etapa de filtrado y un método envolvente para seleccionar las características mas importantes. Para este trabajo, se considera únicamente las letras mayúsculas y números del conjunto de datos EMNIST. Con ello, en busca de disminuir la complejidad, se realiza un árbol de decisión binario dividiendo el problema original de clasificación entre letras y números en subproblemas. En cada nodo de decisión del árbol binario se busca un reducto como base para seleccionar las mejores características mediante el método embebido. Como resultados, para cada nodo se obtuvieron subconjuntos de menor número de características con alto desempeño en la clasificación pero con algunas observaciones en cuanto a la habilidad de discernimiento. Además, se constata como la distribución de las muestras afecta el desempeño del clasificador y de los reductos.

Palabras clave: selección de características, reconocimiento de patrones, complejidad exponencial, reconocimiento de caracteres escritos a mano, selección de características con colonias de hormigas, árboles de decisión binarios.

ABSTRACT

This work addresses the development of an improved method for feature selection and a strategy to classify very large datasets. In the field of supervised classification, the use of large amounts of data implies new challenges for researchers due to high processing time. To deal with this problem, the proposed method makes use of the divide-and-conquer paradigm and the application of an embedded method with a filtering stage and an enveloping method to select the most important features. For this work, only the uppercase letters and numbers of the EMNIST dataset are considered. With this, in order to reduce complexity, a binary decision tree is made by dividing the original classification of letters and numbers problem into subproblems. At each decision node of the binary tree, a reduct is searched as the basis for selecting the best features using the embedded method. As a result, subsets with less than 50% of the total features were generated for each node. It is important to emphasize that some observations are made regarding performance. In addition, it is verified how the distribution of the samples affects the performance of the classifier and the reducts.

Keywords: feature selections, pattern recognition, exponential complexity, handwritten characters recognition, ant colony feature selection, binary decision trees.

TABLA DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

I. INTRODUCTION

Feature selection is defined as the problem of finding the most compact and informative set of features to achieve an improved classifier performance (Perez et al, 2015), (Subasi et al, 2019), (Fernandez et al, 2015). During the last years, the feature selection problem has become more important due to the impact of very large databases on classification algorithms. For instance, in the handwritten characters recognition field, very large datasets imply great challenges for researchers due to the number of features and the independent samples as mentioned in (Pires et al, 2019), (Cilia et al, 2019), (Scheidegger et al, 2021). As a result of the use of these databases, the computational cost is affected in terms of resources and time where the performance of classifiers may not be the desired (Radaideh et al, 2019), (Ayesha et al, 2020).

To deal with this problem, many techniques were proposed using the approach of dimensionality reduction. For instance, principal component analysis (Bro et al, 2014), linear discriminant analysis (Schein et al, 2021), and ISOMAP (Geng et al, 2021). Nevertheless, there are certain problems related to its performance. The PCA according to (Wang et al, 2022) is not sufficient for finding differences between samples, LDC has poor performance with large features and small samples (Qu et al, 2017), and ISOMAP is very slow due to the complexity of its algorithms (Hyodo et al, 2012).

In the same way, the techniques that use the feature selection approach, according to (Perez et al, 2015) can be classified as filters, wrappers, and embedded methods. Filters perform a ranking of features based on the entire data set and only the best rated are considered. Wrappers work with machine learning classifiers as a black box where the subsets are evaluated in order to choose the best scored. Embedded methods perform feature selection prior to training a machine learning model. The application of wrappers improves

the performance of classifiers. For instance, the research of Torres et al (2021) describes the application of Rough Set Theory, artificial neural networks, and genetic algorithms as wrapper method. An overview of the assessment metrics evidences a significant improvement in handwritten numbers classification.

The Rough Set Theory is a powerful tool that handles feature reduction preserving the discernment ability (Torres et al, 2021), (Polkowski et al, 2000). From it, the concept of reduct is defined as the subset that has useful information and allows to differentiate objects of the entire database (Torres et al, 2021). Based on this definition, a reduct can be used as a starting point to build an improved wrapper feature selection method. However, the process of search reducts for the entire database becomes an NP-hard problem (Rodriguez et al, 2020). Then as is mentioned in (Torres et al, 2021) a minimal length reduct can be employed with other feature selection techniques. In the state of art, the algorithm developed for finding reducts with minimal length called min Reduct is described in (Rodriguez et al, 2020).

The objective of this research is to face the problem of feature selection and the classification of a large dataset. To solve these issues, the investigation uses concepts of RTS and an embedded method. The dataset for testing is extracted from (Cohen et al, 2017) considering only upper case letters and numbers. Due to this, the computational complexity is increased, and the divide and conquer paradigm is used over the dataset classification. According to (Smith, 1985), this paradigm is used to divide the original problem into small problems to get better performance. As embedded method, the research uses the advanced binary ant colony. According to (Kashef et al, 2015), it has a filter method and a modified method based on the ant colony system with a K-Nearest Neighbor algorithm as the classifier. Those are implemented to generate the best subsets with a high value for the fitness function.

II. MATERIALS AND METHODS

In this section, the basic concepts are described in the following three subsections to understand the purpose of this research. This includes Rough Set Theory, K-Nearest Neighbor, and Advanced Binary Ant Colony Optimization. The last subsections are focused on the dataset and the combination of all components to explain in detail the proposed method.

A. Rough Set Theory (RST)

An informal system is the representation of an information database as a table. Each row represents an object and every column a feature or attribute (Polkowski, 2000). Formally, let I, an information system, defined as $I = (U, A)$, such that U is not-empty finite set of objects called the universe and A is a non-empty finite set of attributes (Polkowski, 2000). For every attribute of A satisfies the following mapping $a: U \rightarrow V_a$, where V_a is called the value set of A. Attributes of A are divided in two types, condition C and decision attributes D, where it follows that $A = C \cup D \mid C \neq \emptyset \; and \;\; C \cap D \neq \emptyset$.

Let $B \subseteq A$ for a condition attributes, the indiscernibility relation is defined as

$$IND(B \mid D) = \{(x, y) \in U^2 \mid [a(x) = a(y) \in B] \wedge [\delta(x) = \delta(y)] \qquad (1)$$

where $a(x)$ is the value of A for any object x, and $\delta(x)$ is the value of the decision attribute (Torres et all, 2021). Considering that, the concept of reduct is defined using the indecernibilitytion relation. In other words, a reduct of an information system is a subset, which allows discerning between objects belonging to different classes. Formally the next definitions are presented for characterizing reducts:

Definition 1. Let D the set of decision attributes and C the set of condition attributes of a decision system DS, the subset $B \subseteq C$ is a decision reduct of DS if it satisfies the conditions described in (Torres et al, 2021).

Definition 2. Let DM be a discernibility matrix and r_k be a row of DM, r_k is a superflow row of DM if there exists a row r in DM such that $\exists i\ |(r[i]\ <\ r\_\{k\})\ \wedge\ \forall i\ |(r[i]\ \leq r\_\{k\}[i])$ where $r[i]$ is the $i-th$ element of the row r (Torres et al, 2021).

Definition 3. Let BM be a basic matrix and L be an ordered list of condition attributes. L is associated twitha super–reduct if and only if in the sub–matrix of BM considering only the attributes in L, there is not zero row (a row with only zeros) (Torres et al, 2021).

Note that $minReduct$ algorithm (Rodriguez et al, 2020) works with these definitions, this is why it is considered.

B. K-Nearest Neighbor (KNN)

K-Nearest Neighbor (KNN) is an algorithm used for supervised classification based on the distance between training and test data sets. The main idea is to determine the classes of the samples according to their closest proximity to others. For this purpose, the Euclidean distance is used according to the definition in Equation 2. In addition, a k value is used to represent the nearest neighbors for a specific test sample. Thereby, the distance to k training samples is computed, and the majority of the classes of k are used to predict the class of the test sample based on the minimum distance.

Definition 4. Let x_i an input sample with $m > 0$ features such that $x_{i,1},\ x_{i,2},\ x_{i,3}\ , \ldots , x_{i,m}$ represents each feature that belong to x_i (Peterson, 2009). The Euclidean distance between two samples x_i and x_j is defined as

$$d(x_i, x_j) = \sqrt{(x_{i,1} - x_{j,1})^2 + (x_{i,2} - x_{j,2})^2 + \cdots + (x_{i,m} - x_{j,m})^2} \qquad (2)$$

C. Advanced Binary Ant Colony (ABACO)

Advanced Binary Ant Colony Optimization (ABACO) is a metaheuristic algorithm used for feature subset selection that takes inspiration from the behavior of real ants. When a path to a food source is found, it is marked by a pheromone to guide other ants to it. The quantity of the laid pheromone depends on the distance, quality, and quantity of the food source found (Kashef et al, 2015). The previously laid trail is detectable for other ants and with a high probability can decide to follow it. Thus the trail chosen is reinforced by new pheromones. This process is a feedback loop where the trail with a high amount of pheromone is more attractive for ants to follow it. In this way, an indirect method of communication is established via a pheromone trail and allows to find the shortest path between the food source and their colony (Dorigo et al, 1996).

For feature selection, the problem is represented with a fully connected graph where nodes are the features of a data set and the edges between them are the options to select the next node (Kashef et al, 2015). To achieve better results, a binary selection is implemented with two sub-nodes assigned to each feature in the graph, that represent the selection or deselection of the corresponding feature. The artificial ants are the agents responsible for finding the optimal subset that is based on the probabilistic function presented in Equation 3. An ant can only choose one sub-node 1 or 0 of a feature F_i. In other words, it can be selected or deselected and advances to the next node. When the option is chosen, all unvisited nodes are considered for selection and the process is repeated. This is followed by all ants in the colony and each of them should visit all features (Kashef et al, 2015).

Once all ants have finished, the generation is complete and the evaluation of its routes by a fitness function is executed. The best subset evaluated is chosen and the evaporation of pheromones on all edges is triggered. Then the pheromone of the nodes is updated according to Equation 6.

The probabilistic function of transition, denoting the probability of an ant at node $F_{i,x}$ to choose an edge for connecting with $F_{j,y}$ is defined as the following.

$$P_{ix,jy}^{k}(t) = \begin{cases} \frac{\tau_{ix,jy}^{\alpha}\eta_{ix,jy}^{\beta}}{\sum l\, \tau_{ix,l0}^{\alpha}\eta_{ix,l0}^{\beta}+\sum l\, \tau_{ix,l1}^{\alpha}\eta_{ix,l1}^{\beta}} & if\ j,l\ are\ unvisited\ nodes\ and\ edges \\ 0 & otherwise \end{cases} \quad (3)$$

The heuristic value $\eta_{i,j}$ represents the attractiveness of the edges between a node i and j. Here, the F-Score Equation 4 mentioned in (Kashef et al, 2015) is used as a measure to evaluate the discrimination capability of a feature i.

$$\text{F-Score}_i = \sum_{k=1}^{c} \frac{\left(\overline{x_i^k} - \overline{x_i}\right)^2}{\sum_{k=1}^{c}\left[\frac{1}{\left(N_i^k - 1\right)}\sum_{j=1}^{N_i^k}\left(x_{ij}^k - \overline{x_i^k}\right)^2\right]}, (\text{i} = 1, 2, \ldots, \text{n}) \quad (4)$$

the variable c represents the number of classes in dataset; n, the number of features; N_i^k, the total of samples of the feature i in class k, where $(k = 1,2,3, \ldots, c)$ and $(i = 1,2, \ldots, n)$; x_{ij}^k, the $j-th$ training sample for the feature i in class k such that $\left(j = 1,2, \ldots, N_i^k\right)$; x_i, the mean value of feature i considering all classes and x_i^k, the mean of feature i of the samples in class k (Kashef et al, 2015). The heuristic follows the next criterion for the sub-nodes:

$$\eta_{i0,j0} = (\epsilon/n)\sum_{k=1}^{n}\text{F-Score}_k$$
$$\eta_{i0,j1} = \text{F-Score}_j$$
$$\eta_{i1,j0} = (\epsilon/n)\sum_{k=1}^{n}\text{F-Score}_k$$
$$\eta_{i1,j1} = \text{F-Score}_j \quad (5)$$

where n represents the total number of features and ϵ is a constant $\in (0,1)$ (Kashef et al, 2015).

The amount of pheromone $\tau_{i,j}$ between two nodes i and j depends of the best ant's route. To model this process the next equations are defined.

$$\tau_{i,j}(t) = \left[(1-\rho)\tau_{i,j}(t) + \Delta\tau_{i,j}^{g}(t)\right]_{\tau_{min}}^{\tau_{max}} \quad (6)$$

$$\Delta\tau_{i,j}^{g}(t) = \begin{cases} \frac{Q}{F^{g}} if arc(i,j) is in T_g \\ 0 \quad otherwise \end{cases} \quad (7)$$

Here ρ is a constant of pheromone evaporation, $\Delta\tau_{i,j}^{g}(t)$ is the amount of pheromone laid on edge (i,j) by the best global ant of generation at iteration t. In Equation 7, Q is a constant related to the total pheromone laid by an ant, and F^{g} is a cost function that can be a classification error rate. The variable x and y means selection or deselection node, that is $x, y \in [0,1]$ where 1 means that the node is selected and 0, deselected. The constants α and β are the influence on pheromone amount τ and heuristic value η in Equation 3.

Algorithm 1 Pseudo code of advanced ant colony algorithm

```
Data: One reduct, training and test dataset
Result: [ ] ← The best min length subset
a [] ← features after F-Score analysis
 max-gens ← set max generations
 α and β ← set the weights for probabilistic function
 ρ ← the pheromone decreasing constant
 Q ← the amount of pheromone laid by ants
 n-ants ← the number of ants for colony

while not reached max generations do
    ants = generateAnts(n-ants)
     paths = []
     while not reached max number of ants do
         paths.append(ants[iter].findPath())
          iter = iter + 1
    end
    bestRoute = findTheBestPath()
     updateGlobalResult(bestRoute)
     update-fermone-map(bestAntRoute)
end
```

Figure 1. Pseudo Code of Advanced Ant Colony Algorithm

D. EMNIST Database

The EMNIST database is an extension to handwritten letters of MNIST that is derived from NIST Special Database 19 and provides 552.670 samples for training data divided into 344.307 digits, 208.363 uppercase, and 70587 samples for testing data divided in 58.646 digits, 11.941 uppercase (Cohen et al, 2017). For this research, only the uppercase letters and numbers are considered. A total of 52248 images of 28 pixels per 28 pixels were used. Due to the high dimensional data that machine learning models have to handle, the problem of classification becomes interesting.

E. Proposed method

In this section, the proposed method is introduced to obtain the minimum length subset of features that can discriminate between elements of different classes. For this purpose, the algorithm is divided into three stages.

The objective of the first stage is to build a binary decision tree following the divide and conquer paradigm. This is intended to reduce the complexity of the training stage for the classifier KNN. To start the binary decision tree, a classifier process, that works with the union of uppercase letters and numbers dataset, is established in the root node. Hence, a number or letter is predicted. To continue the process of classification, two nodes connected to the root are considered. For simplicity, these nodes are called A and B. Node A is a decision node that works only with letters, and node B with numbers. On the node A, a classifier process of curved and straight uppercase letters is implemented. For node B, the algorithm implemented in (Torres et al, 2021) can be used for number classification. Figure 2 illustrates the binary decision tree.

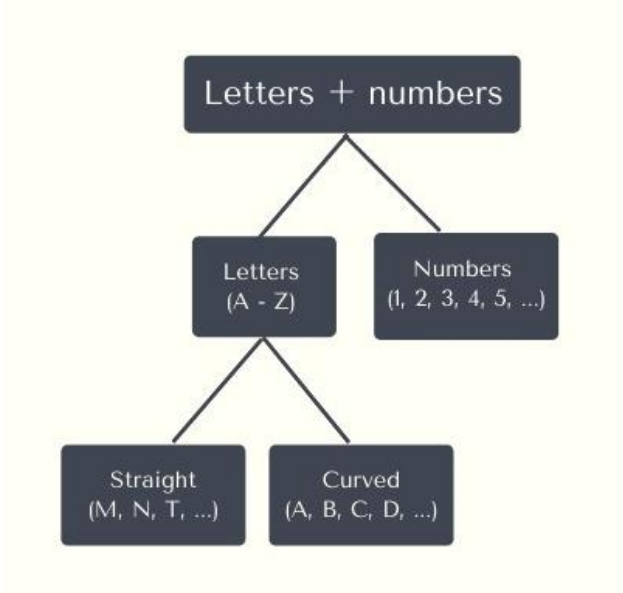

Figure 2. Representation of binary decision tree

The second stage is to find one minimum length reduct for each node in the binary decision tree. Considering the training dataset, only randomly three images were selected by the node. In the EMNIST dataset, each pixel is in the range of [0, 255], so the dataset is binarized with a threshold value equal to 100. In other words, if any value is greater than the threshold, it becomes 1 otherwise 0. The number of samples and threshold was determined by experimentation to ensure the best performance of the next step.

The second step is the execution of $minReduct$ algorithm. The difference mentioned in (Torres et al, 2021) must be considered, that is from the comparison matrix place each column with one on the rows as close to left as possible, so an approximation to an upper-triangular matrix is formed. Any changed column has to be stored in order to translate to the original form of the comparison matrix. The third step is to search for a maximum length limit and every time evaluate if it is a reduct or not. Once one reduct is found, proceed to the next stage.

The third stage is to improve the classification performance of the reduct over the entire data set. For its implementation, a filter method is performed using Equation 4 to rank the importance of each feature. Then, the ant colony of ABACO as the wrapper method is

used to get subsets and rank on the KNN classifier. The precision score and the mean squared error of KNN are used as fitness and penalty functions. Finally, the best subset of each node is found.

F. Experimental setup

This section describes the data processing for each node of the binary decision tree, and the parameters of ABACO and KNN following the purpose method.

1) Data processing for *minReduct*

Considering each node, 3 random samples were chosen from the training dataset to build a new dataset. In order to perform a binary classification, for the root node, two classes were formed, which are named 1 for uppercase letters and 0 for numbers. In the case of the node, A, two classes are formed curved and straight letters. This division is based on the arbitrary criterion that considers the morphology of each letter.

Then, each image of the new datasets builds a basic matrix. With an intensity threshold value of 100, the binarization process is applied to the basic matrix. After that, the comparison matrix is computed where all superfluous rows are deleted and a close enough to a triangular matrix representation is formed.

2) Configuration for *minReduct*

For each node of the binary tree decision, the algorithm is executed until the maximum length of one reduct is found. Once the value is obtained, the algorithm is restarted with this value as the new maximum length. Until a minimal length reduct is found this process is repeated.

3) Data processing for *ABACO*

For the root node of the binary decision tree, the training dataset has 10000 numbers and 10010 uppercase letters, and for the node A, the training dataset includes 19994 uppercase letters. In order to get the most important features, the F-Score analysis is

performed using Equation 4 over the training datasets. It means that only those features with an F-Score value of more than zero are considered for the next step of the algorithm.

4) *ABACO* parameters and configuration

The following values are recommended for the ABACO feature selection algorithm according to (Kashef et al, 2015), the maximum generation is set to 100, the evaporation coefficient of the pheromone laid by ants ρ is set to 0.05, the maximum, and the minimum pheromone intensity are established to 0.1 and 6 respectively. Also, the influence of pheromone α is established at 1 ,and the heuristic β is 0.2. The beta value is lower than the alpha value due to the F-Score for some cases is a very tiny fraction, so to give more importance, a low number for β is used. The initial pheromone intensity τ_0 is set to 1 for an equal distribution of pheromone and the amount laid by an ant as 1.

To evaluate the subsets generated in each generation by the algorithm, the precision score is used to rate the correct positive predictions (Grandini et al, 2020). The mean square error is used as a penalty for bad predictions (Magnussen et al, 2019). The purpose of evaluating the subsets with these scores is to find the best subset that discriminates correctly between the classes of the samples based on the KNN classifier. For the classifier, the k neighbors parameter is established to 11 for the classification in the root node and 9 for the node *A*. Note that these values were determined by experimentation as mentioned in (Taunk, 2019). Additionally, the Euclidean distance is used as a measure between two samples defined in Equation 2.

5) Assessment metrics

In order to validate the discrimination ability of the best subset after ABACO finishes its execution, the confusion matrix (Luque et al, 2019), precision vs recall (Flach et al, 2015), and ROC (Tharwat, 2021) curves are computed to show the performance of the classifier. Additionally, the precision, accuracy, and recall scores are used to get a complete analysis of

the predictions. Note that all source code is implemented in Python version 3.9 with the Scikit learn library (Pedregosa, 2011).

III. RESULTS AND DISCUSSION

In this section, the graphs of assessment metrics and confusion matrix are presented and discussed for the two nodes of the binary decision tree. For the first and second level, a reduct with 11 features (columns) was found. The subsets obtained by ABACO represent 39.66 % and 42.44 % of the total amount of features.

A. Performance evaluation

For the root node, 4002 images were used to test the proposed method over unknown images. To analyze the performance of classification and prediction, the confusion matrix and scores are presented in Figure 3 and Table 1 respectively.

Table 1. Metrics for the root node of binary decision tree

Score	Value [%]
Accuracy	85.15
Precision	93.87
Recall	77.27
Error	13.89

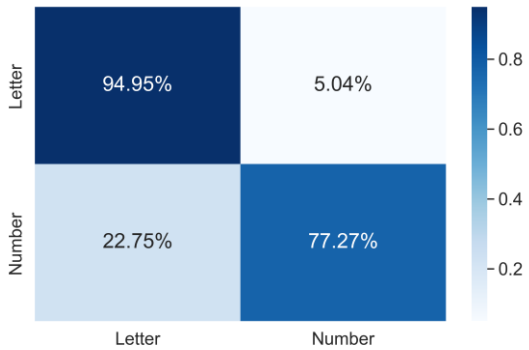

Figure 3. Confusion matrix for the root node of the binary decision tree

From Table 1, according to the precision score, the predictions of images as letters are correct at 93.87%. Therefore, few letter images were predicted as numbers. To support it, the confusion matrix demonstrates that only 5.04% of images classified as letters were originally numbers. On the other hand, the recall score shows that 77.27 % of images, originally identified as letters, were correctly classified. According to the confusion matrix, this value is moderately low because 22.75 % of letters were classified as numbers. In general, the classifier predicted some cases correctly this can be verified by the accuracy value.

In order to assess the performance, the precision vs recall and ROC curves are presented. Figure 4 left, shows an area under the curve close to 1, so the classifier has a high performance differentiating two classes. On the other hand, Figure 4 right indicates the average precision whose value is 0.942. Considering that high scores of precision and recall represent a better performance of classifiers. The value calculated means that the classifier predicts wrongly some cases affected by the letters classified as numbers.

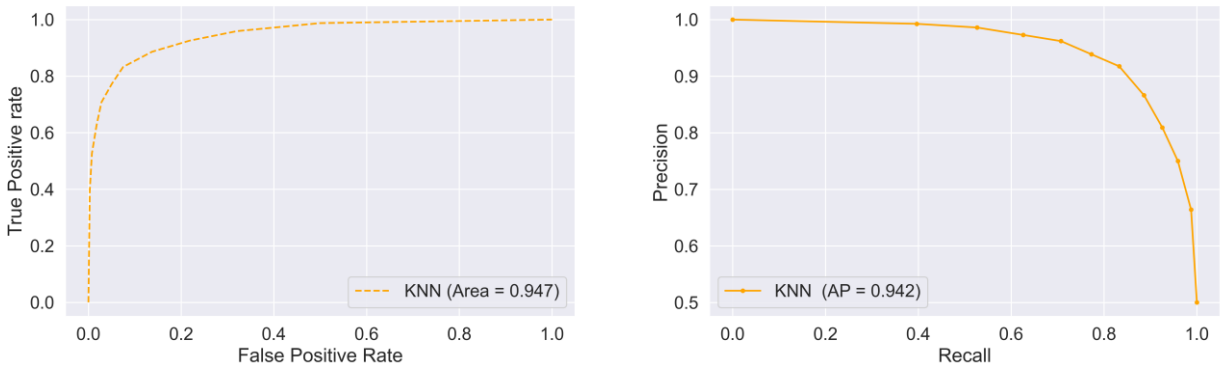

Figure 4. ROC (left) and Precision vs Recall (right) curves for the root node

Evidently, some misclassified characters affect the performance of predictions. In general, 13.89 % are considered as error. For this, a search to detect these abnormal cases were done and presented. Note that only 5 first cases are considered due to their relevance.

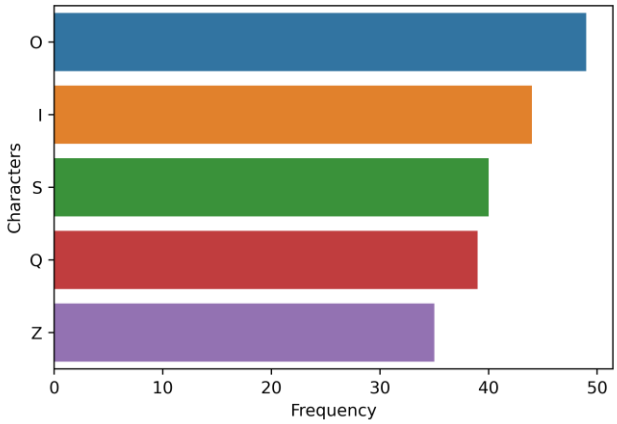

Figure 5. Misclassified characters for the root node of binary decision tree

Taking the first case in Figure 5, 49 samples of the letter "O" were classified as numbers. This error can be produced due to the similarity between characters "O" and "0". In a real application, if the human brain does not receive additional information, the recognition will be erroneous. Hence, for the remaining cases, the error can be admissible.

Score	**Value [%]**
Accuracy	93.03
Precision	96.58
Recall	89.22
Error	6.96

Table 2. Metrics for the node *A* of binary decision tree

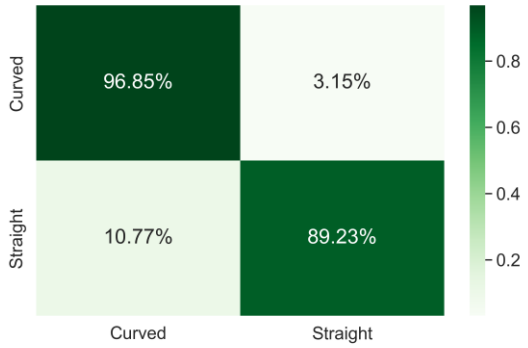

Figure 6. Confusion matrix for the node *A* of the binary decision tree

For the node *A*, a dataset with 8242 images was used for testing. According to Table 2 the accuracy score of 93.03 % means that most of the predictions were made correctly. To

support that in Figure 6, 95.14 % of straight and 89.23 % of curved letters were classified correctly over the entire testing dataset. The precision score shows that the classification was correct in 96.58 % for straight letters. In fact, only 3.15 % of images were classified as curved. The recall score indicates a good sensibility for straight letters where 89.22 % are

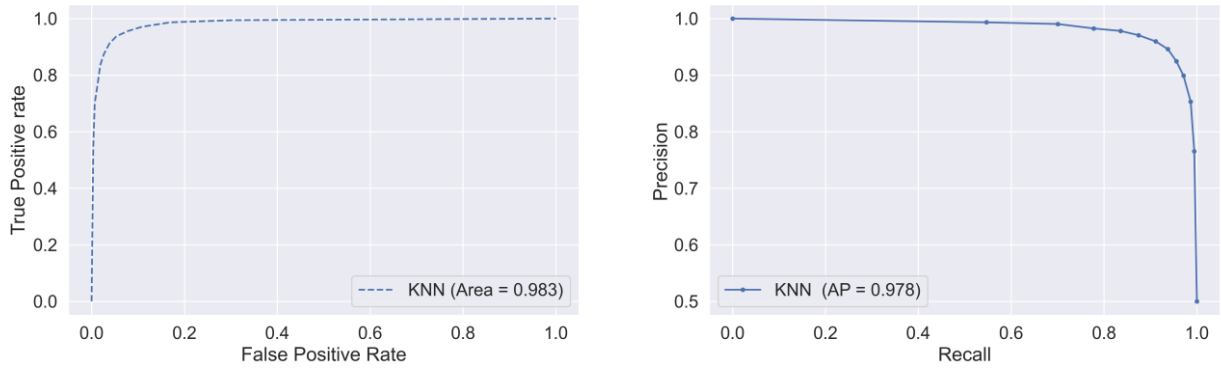

Figure 7. ROC (left) and Precision vs Recall (right) curves for the node *A*

Considering, the precision and recall curve presented in Figure 7 right, the average precision value of 0.978 indicates that the predictions were labeled correctly. The area under the curve in Figure 7 left means high performance in the discrimination between the two classes.

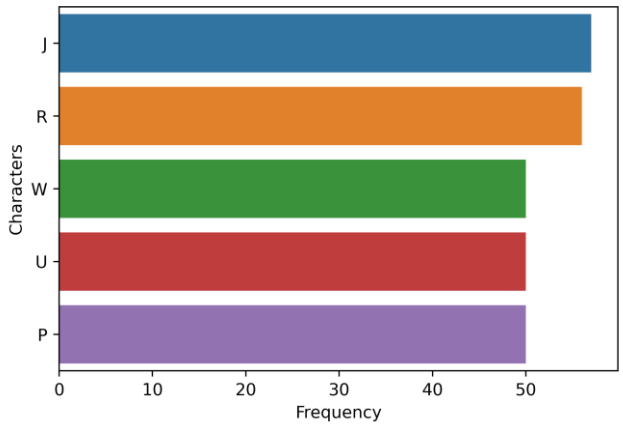

Figure 8. Misclassified characters for the node *A* of binary decision

On the other hand, the classifier presents an error score of 6.96 %. Considering Figure 8, 57 samples of the letter "J" and 56 of the letter "R" are labeled as straight letters. This error is related to the morphology of the letter, where some samples are similar to the "I" and "E"

letters. For the next cases, the error has a similar explanation. As a consequence, the identification task is easily prone to errors in all letters.

The overall performance of the binary decision tree is affected by the morphology of the binary groups formed at each level. Despite this, the classification has high performance in the recognition of the class characters. The first level in comparison to the second had a better performance due to the reduced complexity from the classification of letters and numbers to only letters.

B. State of art based comparison

The feature selection methods mentioned in the research of Torres et al (2021), Naqvi (2012), and Duangsoithong et al (2010) are used to compare the performance of the proposed method. It is important to emphasize that the recommendations of parameters and settings by the authors were considered for this comparison. In addition, the accuracy score of KNN was used as a fitness function, and the error rate was employed to evaluate the predictions.

Table 3. Evaluation for the dataset used on the root node

Method	**Features (%)**	**Error (%)**
20 Reducts + Genetic algorithms	21.94	14.55
Pearson correlation analysis + SFFS	33.28	15.74
Mutual information + SFS	40.05	16.75
Reducts + ABACO	**36.66**	**13.90**

From Table 3, notices that the Reducts + ABACO is the method that generated a subset with better performance than the other subsets. In fact, the error rate obtained by the subset is 13.90%. This may be possible due to the performance of the ant colony algorithm selecting features. In comparison to the Pearson correlation analysis + SFFS and Mutual information + SFS, the error is higher due to the complexity of the wrappers methods mentioned in (Naqvi, 2020), and (Duangsoithong, 2010). On the other hand, the number of features of the subset generated by the proposed method compared to the subset generated by

20 Reducts + Genetics algorithms is higher due to the complexity of the ABACO algorithm. In other words, for the proposed method more generations are needed to further reduce the size of the subset. Compared to the other methods, the number of selected features remains in the range of 30% to 40%.

In the same way, Table 4 reports the results for node A. Notices that the error rate of the subset generated by Reduct + ABACO is 6.97 % with 42.44 % features. Here the proposed method generated a larger subset of features compared to the subsets generated by the other methods. This means that more generations are needed to obtain better performance in terms of feature reduction. In contrast to the error rate generated by Pearson correlation analysis + SFS and Mutual information, the proposed method obtained a better performance.

Table 4. Evaluation for the dataset used on the node *A*

Method	**Features (%)**	**Error (%)**
20 Reducts + Genetics algorithms	20.15	6.50
Pearson Correlation Analysis + SFS	28.58	8.14
Mutual information + SFS	35.07	7.40
Reduct + ABACO	**42.44**	**6.97**

In general, the proposed method had a good performance due to the implementation of the ant colony algorithm. The best subset was generated by the proposed method of Torres et al (2021). This can be explained by the use of 20 reducts which improve the performance of the classifier.

IV. CONCLUSIONS AND FUTURE WORK

Based on the divide and conquer paradigm, this research work presents a strategy to face the feature selection problem. To prove the strategy, the original dataset was divided into two datasets: uppercase letters and numbers. In addition, the uppercase letter dataset was divided into two datasets: curved letters and straight letters. Each of these divisions is represented as a decision node in the binary decision tree. The proposed method is applied to every decision node and a reduct is found. Based on the reduct found, the embedded method is applied using ABACO with F-Score as a filter method and the precision score of the KNN classifier as the fitness function.

The results present two subsets with 39.66% and 42.44% of features for each decision node of the binary decision tree. Taking into consideration the assessment metrics for the classification task of the root node, the constructed subset generated few erroneous predictions. Despite this, it keeps the ability to discriminate. On the other hand, for the node A, the subset generated has a significant improvement in contrast to the classifier of the root node. Due to the assessment metrics show that the majority of predictions were correct. Therefore, it is important to emphasize that the distribution of samples in the training dataset affect directly the performance of the classifier. Thus, the results show that the application of the divide and conquer paradigm improves the performance of classifiers.

Future work will be aimed to: integrating artificial neural networks instead of KNN; creating and applying the proposed method to new nodes following the divide paradigm and extending to multiclass classification problems.

REFERENCES

Al-Radaideh Q. & Mohammed A. (2019). An Arabic text categorization approach using term weighting and multiple reducts. *Soft Computing*, 23, 14, 5849–5863.

Ayesha S., Hanif M., & Talib R. (2020). Overview and comparative study of dimensionality reduction techniques for high dimensional data. *Information Fusion*, 59, 44–58.

Bro R. & Smilde A. (2014). Principal component analysis. *Analytical methods*, 6, 9, 2812–2831.

Cilia N., De Stefano C., Fontanella F., & Scotto di A. (2019). A ranking-based feature selection approach for handwritten character recognition. *Pattern Recognition Letters*, 121, 77–86.

Cohen, G., Afshar, S., Tapson, J., & van Schaik, A. (2017). EMNIST: an extension of MNIST to handwritten letters.

Dorigo M., Maniezzo V., & Colorni A. (1996). Ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 26, 1, 29–41.

Duangsoithong, R., Windeatt, T. (2010). Correlation-Based and Causal Feature Selection Analysis for Ensemble Classifiers. In: Schwenker, F., El Gayar, N. (eds) Artificial Neural Networks in Pattern Recognition. ANNPR 2010.

Flach P. & Kull M. (2015). Precision-recall-gain curves: Pr analysis done right. *Advances in Neural Information Processing Systems*, 28.

Fernández A., Gómez A., Lecumberry F., Pardo A., & Ramírez I. (2015). Pattern recognition in latin america in the “big data” era. *Pattern Recognition*, vol. 48, no. 4, pp. 1185–1196, 2015.

Geng X., Zhan D., & Zhou Z.. (2005). Supervised nonlinear dimensionality reduction for visualization and classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 35, 6, 1098–1107.

Grandini M., Bagli E., & Visani G. (2020). Metrics for multi-class classification: an overview.

Hyodo M., Yamada T., Himeno T., and Seo T. (2012). A modified linear discriminant analysis for high-dimensional data. *Hiroshima Mathematical Journal*, 42, 2, 209 – 231.

Kashef S. & Nezamabadi-pour H. (2015). An advanced aco algorithm for feature subset selection. *Neurocomputing*, 147, 271–279.

Luque A., Carrasco A., Martín A., & de las Heras A. (2019). The impact of class imbalance in classification performance metrics based on the binary confusion matrix. *Pattern Recognition*, 91, 216–231.

Magnussen S., McRoberts R., & Tomppo E. (2009). Model-based mean square error estimators for k-nearest neighbour predictions and applications using remotely sensed data for forest inventories. *Remote Sensing of Environment*, 113, 3, 476–488.

Naqvi, S. (2011). A Hybrid Filter-Wrapper Approach for Feature Selection.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825--2830, 2011.

Peterson L. (2009). K-nearest neighbor. *Scholarpedia*, 4, 2, 1883.

Pérez N., López M., Silva A., & Ramos I. (2015). Improving the mann–whitney statistical test for feature selection: An approach in breast cancer diagnosis on mammography. *Artificial Intelligence in Medicine*, 63, 19–31.

Pires A. & Branco J. (2019). High dimensionality: The latest challenge to data analysis. *Arvix*.

Polkowski L. & Skowron A. (2000). Rough sets: A tutorial.

Qu T. & Cai Z. (2017). An improved isomap method for manifold learning. *International Journal of Intelligent Computing and Cybernetics*, 10, 30–40.

Rodríguez V., Martínez-Trinidad J., Carrasco-Ochoa J., Lazo- Cortés M., and Olvera-López J. (2020). Minreduct: A new algorithm for computing the shortest reducts. *Pattern Recognition Letters*, 138.

Scheidegger F., Istrate R., Mariani G., Benini L., Bekas C., & Malossi C. (2021) Efficient image dataset classification difficulty estimation for predicting deep-learning accuracy. *The Visual Computer*, 37.

Schein, A., Saul, L.K. & Ungar, L. (2003). A Generalized Linear Model for Principal Component Analysis of Binary Data. *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, in *Proceedings of Machine Learning Research*, 240-247.

Subasi A. (2019). Chapter 4 - feature extraction and dimension reduction. *Practical Guide for Biomedical Signals Analysis Using Machine Learning Techniques*. Ed. Academic Press, 193–275.

Smith D. (1985). The design of divide and conquer algorithms. *Science of Computer Programming*, 5, 37–58.

Taunk K., De S., Verma S., & Swetapadma A. (2019). A Brief Review of Nearest Neighbor Algorithm for Learning and Classification.

Tharwat A. (2021). Classification assessment methods, 17.

Torres E., Monserrate, I., & Ibarra J. (2021). A New Approach for Optimal Selection of Features for Classification based on Rough Sets, Evolution and Neural Networks. *Universidad San Francisco de Quito*.

Wang S., Bai L., Chen X., Wang Z., & Shao Y. (2022). Divergent projection analysis for unsupervised dimensionality reduction. *Procedia Computer Science*, 199, 384–391.