Nearest Centroid Neighbour, An Alternative in Pattern Recognition for Detecting New Tasks in a Mobile Robot Simulator

E. RANGEL [1] and R. BARANDELA [2]

[1] Depto. de Inteligencia Artificial, Universidad Americana de Comercio e Informática, Av. Pungarabato Ote 103, Cd. Altamirano, Gro. México.
[2] Laboratorio de Reconocimiento de Patrones, Instituto Tecnológico de Toluca, Ave. Tecnológico s/n, 52140 Metepec, México.
erangel_lugo@hotmail.com

Abstract

In this paper we show a knowledge-based system by speech recognition in the input data, it allows to execute a mobile robot simulator of Ayllu's routines. Furthermore, it can be able for detecting new tasks using "reject" techniques. Commonly for the classification phase in speech recognition are used the supervised neural networks (the BackPropagation Algorithm [BPA] has been extensively used), and have shown a best performance. However when the knowledge-based system uses an incomplete training sample, the neural networks (BPA) are not a good choice. Hence, the knowledge-based systems must use some techniques with "reject". Thus, there are techniques with "reject" based about Nearest Neighbor for detecting new classes. We present an alternative for the "reject" techniques, using k-NCN rule applied to the incomplete training sample problem. Novel definition of Averaged Distance with Euclidean and Manhattan distance criterion are concerning to the Nearest Centroid Neighborhood is also introduced.

Keywords: Nearest Centroid Neighbour rule, Neural Networks (BackPropagation Algorithm), Reject Techniques, Averaged Distance Technique, Speech Recognition, Ayllu´s Task.

1. Introduction

The expert systems commonly need a knowledge base module and inference engine [**1**]. The knowledge base has questions and answers concerning the solution for the assigned problem. The inference engine is used for the discrimination from the information included in the knowledge base; also it is in charge of the decisions performance by the system. However, in the supervised methods, a knowledge-based system, which has learning, uses a training sample [**2**]. From the training sample depends the learning phase, thus the supervised method will learn from it. The neural networks (BPA) [**3**], in the learning phase, use the training sample to obtain the generalized weights. In the classification phase, the weights of the neural networks are used and it is not necessary the use of the training sample any more [**4**]. However, in the nearest neighbor rule [**5**], the training sample is used in both phases: Learning and Classification.

In the mobile robotics area, commonly before practicing in real time any experiment, it is used a simulator where all the moves and roads to executing with the robot are studied [**6**]. In this simulator the programming errors are observed, and can be corrected later. Some well-known mobile robots are the Pioneer's collection of ActivMedia [**7**]. There are many programming languages for a mobile robot of ActivMedia [**8**]; some of them are Saphira, Java, Ayllu, and C++. The routines for Shapira, C++, and Java must always be programmed and it depends on the routine to be executed, which might be difficult for programming. The Ayllu language in its user's manual [**9**] has some routine, command and tasks included that can be adapted to the necessity. These routines are very easy for programming or for adapting, allowing to execute the command or tasks without any problem, or damage suffered by the robot. Some well-known Ayllu´s routines are four and allow you to execute different task and commands. The routines are named: StraightLine, Wanderer, Goto and Getto. In speech recognition area, it is possible to understand spoken messages by the human voice through from the computer. That is possible, when some techniques concerning language structure, automatic learning, Hidden Markov Models, semantic analysis, analogical frequency analysis and digital frequency analysis are used [**10**]. There has never existed a speech recognition system that understands all the messages input introduced by the human. However, there are many systems, which allow you to work with an acceptable error, even with the very restricted semantic [**11**]. The speech recognition systems learn with a training sample that has frames from voice and text captured [**12**]. The frame is a feature vector that has a smaller part from digital frequency. This frame is labeled in this phase. The classification phase in the speech recognition systems, is also named recognition phase. In this phase, the system allows the identification of human voice; to be able to execute the commands introduced by the microphone. In areas from the speech recognition and mobile robotics; the neural networks (BPA) have been very studied. In speech recognition, they have been commonly used for the classification of frames [**13**] and in mobile robotics to execute the road commands in real time when environment images are used [**14**, **15**]. In those applications there has never been studied the incomplete training sample problem. In neural networks area have not been studied the incomplete training sample problem with BPA [**3**]. Although there are neural networks named *RBF* (Radial Basis Function) [**16**] that allow to create new classes (or new nodes in the output layer) when a new pattern is not mapping to the classes contained in the training sample, which the neural network has learned with. In this work, we have not considered the neural networks *RBF*, because they still are in research process.

The problem of incomplete training data in supervised machine learning methods is receiving growing attention. Incomplete data means that one or more classes are not represented in the training sample. It has been observed that this situation, which arises in several practical domains, may produce an important deterioration of the classification accuracy, in particular with patterns belonging to the classes that are not contained in the training sample [**17**, **18**]. Thus, the knowledge-based systems think that new patterns (of another new classes) are corresponding to the contained classes into training sample, but the systems do not know them. However, the new patterns (of new classes) are labeled as patterns of the contained classes into training sample. Barandela and Najera [**19**] have studied the incomplete training sample problem in pattern recognition area. They show some techniques for detecting new classes in pattern recognition systems with incomplete training sample, but they have employed "reject" [**20**] for detecting only one class in theirs experiments. Furthermore, they employed k-means [**21**] for clustering to the rejected patterns and Depuration [**22**] method for "cleaning" to the "rejected training sample" and these "reject patterns" (once cleaned) can be incorporing into "original training sample". In this work, we have employed similar techniques, in particular *Averaged Distance* [**19**] thus it shows well performance. We have modified the Nearest Neighbor criteria by Nearest Centroid Neighbour criterion. Furthermore, we employed another modifications concerning to the distance function criteria. We show the comparison of results obtained for Nearest Centroid Neighbor and Nearest Neighbor Rules, by *Averaged Distance* techniques employing Manhattan and Euclidean distance functions criterion.

2. Experimented Strategies

The NN rule is a well-known supervised non-parametric classifier that combines its conceptual simplicity and an asymptotic error rate conveniently bounded in terms of the optimal Bayes error [**18, 23**]. In its classical manifestation, given a set of n previously labeled prototypes or training sample (*TS*), this classifier assigns any given instance to the class indicated by the label of the closet prototype in the *TS*. More generally, the k-NN [**24**] rule maps any instance to the pattern class most frequently represented among its *k* closets neighbors. Nevertheless, the NN classifier also suffers from certain drawbacks. The performance of these rules, as with any non-parametric method, is extremely sensitive to incorrectness or imperfections in the training sample. On the other hand, its applicability to real-time problems, with a large set of training patterns of high dimensionality, can become prohibitive because of the immense computational load required for searching the nearest neighbor of each new pattern in the training sample. As a technique for internally biasing the discrimination procedure, we have modified the *Euclidean* metric criteria (which, the NN and k-NN rules have originally employed) by the *Manhattan* distance function criteria (To see equation 2.1). With this modification, when classification of new pattern *Y* is attempted, and in the search through the training sample of its nearest neighbor, the following quantity must be computed for each training instance:

$$D_m\ (x\,,y) = \sum_{i=1}^{d} |x_i - y_i| \qquad (2.1)$$

The *Manhattan* distance function criterion has been employed with 1-NN [**25**] as classification criterion for reducing training samples. Furthermore, *Manhattan* was used for improving to the classification in some real datasets and imbalanced training samples [**26**]. In this work, we present a new proposal using the *Manhattan* distance function criteria for the classification and "reject" techniques with k-NCN rule. The following sections show some related techniques used.

2.1.- The Manhattan k-NCN Classification (*New proposal*)

The nearest centroid neighborhood concept is basically focused on the idea that the neighborhood of a point is subject to two complementary constraints. First, by the *distance criterion*, the k neighbors of a sample, say p, are as near to it as possible. The distance criterion commonly used is the *Euclidean* metric. Second, by the *symmetry criterion*, their centroid is also as close to p as possible. Note that the traditional nearest neighborhood used by the k-NN rules as well as the *LVQ* algorithms [**27**] just takes the first property into account, and so the nearest codebook vectors may not be symmetrically distributed around the sample p in the input space. Let p be a point whose k nearest centroid neighbors (*NCN*) should be found in a set of samples (training set), $X = \{m_1,\ldots, m_n\}$. These k neighbors can be searched for through an iterative procedure [**28**] in the following way: *a)* The first *NCN* to p correspond to its *NN*, say q_1. *b)* The i´th neighbor, q_i , $i \geq 2$, is such that the centroid of this and all previously selected neighbors, say $q_1,\ldots,q_{i-1}$ is the closets to p. This procedure gives rise to a kind of neighborhood in which spatial distribution of neighbors is taken into account because of the centroid criterion. Furthermore, proximity of the k-NCNs to the sample is guaranteed because of the incremental nature of the way in wich they are obtained from the first NN. Using this neighborhood, it is possible to define the so-called k-NCN classification rule [**29**] as follows: *1)* Find the k-NCNs to p, say $X^P = \{m^P_1 , \ldots, m^P_k\}$, where $k \leq n$. *2)* Assign to p the class with a majority of votes among its k-NCNs in the set X^P (resolve ties randomly).

The design of the *Manhattan* k-NCN Classification, consists of the modification of the distance criterion (do not modify centroid criterion), changing only the *Euclidean* distance function criterion (commonly used) by a *Manhattan* distance criterion above-mentioned (To see equation 2.1). The procedure for *Manhattan* k-NCN Classification can be written as follows:

1. Let TS_i the i^{th} prototype of the training sample. Let f the features number. Let C_i the i^{th} centroid obtained for each prototype. Assign to the k value. Let A the integer number, which it represents the current *Manhattan* k-NCN. Put $A = 0$.
2. For each new pattern X:

```
• Find the 1-NN (using the Manhattan distance function) which must be assigned as the Manhattan 1-NCN.
    • Put $A$=1
    • For finding the next Manhattan k-NCNs in iterative form as follows:
    a) Put $j = A$; Let $q_j$ the $j$th-NCN of Manhattan.
    b) Calcule the Centroids among the NCN $q_j$ (in this time) with each prototype $TS_i$. Calcule this
       Centroid how follows:
    c) For $h = 1$ to $f$ do
           If ( $j==1$ ) Then $C_{i,h} = (q_{1,h} + TS_{i,h}) / 2$
           Else $C_{i,h} = (q_{1,h} + \ldots + q_{j,h} + TS_{i,h}) / (j+1)$
       End-For
    d) Calcule the distances among the new pattern $X$ and each Centroids $C_i$. Use the Manhattan distance
       function. Let $TS_c$ the prototype with minor distance. Where $c$ is the number of the prototype NCN.
    e) Put $A = A + 1$; Put $q_A = TS_c$.
    f) If $A \geq k$ break
```

- Once finding the *Manhattan* k-NCN´s, X is assigned to the majority class among those *Manhattan* k-NCN. If a majority class does not exist among those *Manhattan* k-NCNs, the results are decided randomly. The idea with the *Manhattan* k-NCN is to improve the classification in some applications [**25**, **26**], using centroid criterion and *Manhattan* distance function.

2.2.- Frames Selection Procedure in the Speech Recognition

This technique was used for obtaining the features selection of the frame (In system's module concerning by speech recognition). The process for selecting all the frames [**10**] can be written as follows: *1)* To Convert the analogical frequency to digital frequency. *2)* To Select a smaller digital frequency frame [approximately 30 ms]. *3)* To Save the frame with the features, in the range from 30 ms. *4)* Forward 10 ms and repeat the process *2,3,4* until to obtain the features from all digital frequency. *5)* When Step *4* has Finalized do:

```
If (Learning Phase) Then "Call human expert, for assigning the label to the frame"
Else "The Classification rule assigns the label to the frame" End.
```

2.3.- The Neural Network (BAP) used

The best results of the neural network (BAP) [**3**] were obtained when we have only used three layers: Input Layer with 20 nodes, Output Layer with 5 nodes and the Hidden-Layer with 21 nodes. The reason of learning used was:

$$\Delta w_n = -\eta_n \left| E''_w(n) \right|^{-1} E'_w(n) \qquad (2.3.1)$$

The process iterative for the learning phase of backpropagation algorithm can be written as follows:

1. For each vector x_k in the training sample.
2. Compute the lineal functions of the network output and the subtraction: $\varepsilon_k = (d_k - s_k)$.
3. The network weights are updated by event (for each prototype x_k):

$$W_{kj}(t+1) = W_{kj}(t) + \Delta_p W_{kj}(t) \qquad (2.3.2)$$

Were:

$$\Delta_p W_{kj} = \eta \, (y_{pk} - o_{pk}) \, f_k(neta_{pk}) \, i_{pj} \qquad (2.3.3)$$

Then, the updating from weights can be written as follows:

$$W_{kj}(t+1) = W_{kj}(t) + \eta (y_{pk} - o_{pk}) \, i_{pj} \qquad (2.3.4)$$

4. Repeat the steps 1-3 for all the prototypes (P), which are included in the training sample.
5. If the squared error means:

$$E = \frac{1}{2P} \sum_{k=1}^{P} \varepsilon_k^2 \qquad (2.3.5)$$

has a small value and acceptable, the learning phase is finished. Otherwise, repeat from the Step 1 with all the prototypes (x_k) of the training sample.

In this work, have been programmed two Neural Networks: In First experiment, with an incomplete training sample (the original dataset) and in the second experiment, with all the classes (the adapted dataset). In both experiments, the Neural Networks have 20 nodes in their Input Layer, concerning to the features that in the training samples are contained (To see Table 1). However, in first experiment the Output Layer has 5 nodes, thus in the training sample five classes are contained (To see Table 1). In second experiment, the Output Layer has 9 nodes, thus in the training sample nine classes are contained (To see Table 2). Finally, the Hidden-Layer experiments were carrying out employed some variants: *1)* One Hidden-Layers with 21 nodes. *2)* Two Hidden-Layers with 21 nodes. *3)* Three Hidden-Layers with 21 nodes. *4)* One Hidden-Layers with 100 nodes. *5)* Two Hidden-Layers with 100 nodes. On the other hand, for the classification phase the weights obtained in learning phase (for each Neural Network) are used.

2.4.- Reject with Averaged Distance NN

This "reject" technique has two phases: *1) Learning Phase* and *2) Classification or "Reject" Phase.* In *Learning Phase,* the training sample is edited (some patterns of the training sample are deleted). It requires, in the *Reject Phase,* for training as well as for *classification,* the searching of *k* nearest neighbors.

The Learning Phase can be written as follows: **1***)* For each class j represented in the training sample (TS) do: **2)** Let C_j be the set of training patterns with class label j. **3)** For each x_i in C_j do: **3.a)** Find the k-NN of x_i in TS - { x_i }. ***3.b)*** If there is a majority of NNs from C_j Then set d^i = mean value { d (x_i , x_t)} , x_t in the NNs and in C_j } Else, x_i is removed (edited) from the TS . **4)** Let T_j = mean value { d^i) } , x_i in C_j (Thus, T_j saving the mean value of the calculed distances for each class. T_j will be used in classification or reject phase). ***The Classification or Reject Phase*** (as reference set it is employed TSE: TS without the patterns removed in the *Learning Phase*) can be written as follows: **1)** For each new pattern X to be classified do: **2)** Find the k-NN of X in the TSE. **3)** Let q the class label most represented into these NNs of X. **4)** Set d^X = mean value { $d(X,x_m)$ } , x_m in the NNs of X and in C_q. **5)** If $d^X \leq T_q$, then X is assigned to the class q Else, X is rejected as a possible member of a new class.

This technique above-mentioned is named *"Reject with Averaged Distance"*, and it employs *Euclidean* distance function criteria. For the *New Proposal*: *"Manhattan Reject Averaged Distance"* technique, we have employed the Averaged Distance technique but with a *Manhattan* Distance function criterion and value for $k=2$.

2.5.- Reject with Averaged Distance NCN (*New Proposal*)

This technique is a smaller variant for *Averaged Distance NN* (above-mentioned). Hence, has also two phases: *1) Learning Phase* and *2) Classification or "Reject" Phase.* In *Learning Phase*, the training sample is edited with k-NCN rule (some patterns of the training sample are deleted). It requires, in the *Reject Phase*, for training as well as for *classification*, the searching of k Nearest Centroid Neighbors.

The Learning Phase can be written as follows:

```
For each class $j$ represented in the training sample (TS) do:
    • Let $C_j$ be the set of training patterns with class label $j$
    • For each $x_i$ in $C_j$ do
        a) Find the k-NCN of $x_i$ in TS - { $x_i$ }
        b) If there is a majority of NCNs from $C_j$
           Then set $d^i$ = mean value { d ( $x_i$ , $x_t$ ) } , $x_t$ in the NCNs and in $C_j$ }
           Else, $x_i$ is removed (edited) from the TS
    • Let $T_j$ = mean value { $d^i$ ) } , $x_i$ in $C_j$ (Thus, $T_j$ saving the mean value of the calculed
      distances for each class. $T_j$ will be used in classification or reject phase).
```

The Classification or Reject Phase (as reference set it is employed TSE: TS without the patterns removed in the *Learning Phase*).

```
    • For each new pattern $X$ to be classified do:
        a) Find the k-NCN of $X$ in the TSE
        b) Let $q$ the class label most represented into these NCNs of $X$.
        c) Set $d^X$ = mean value { $d(X,x_m)$ } , $x_m$ in the NCNs of $X$ and in $C_q$
        d) If $d^X \leq T_q$ , then $X$ is assigned to the class $q$
           Else, $X$ is rejected as a possible member of a new class.
```

This technique above-mentioned is named *"Reject with Averaged Distance NCN"*, and we have employed it with *Euclidean* distance function criteria. For the another *New Proposal*: *"Manhattan Reject Averaged Distance NCN"* technique, we have employed the Averaged Distance NCN technique, but with a *Manhattan* Distance function criterion and value for $k=2$.

3. Experiments and Results

First, it was necessary to carry out the implementation from the mobile robot simulator. This application was developed in Borland C++ Builder programming´s language. The mobile robot simulator allows the execution from the well-known four Ayllu´s routines: **StraightLine** (The Robot Walks while the road has not obstruction, if it detects some obstacle, it goes backward). **Wanderer** (The Robot Walks while the road has not obstruction, if it detects some obstacle it executes a rotation. Also this routine allows you to carry out a Wall Pursuit {Following – Wall }). **Goto** (This routine is used for arriving at the coordinate {X, Y} defined by the user, if some obstacle is detected in the road, the robot stops or it executes the *StraightLine* routine). **Getto** (This routine is used for arriving at the coordinate {X, Y} defined by the user, when there are some obstacles in the road. The robot solves doing a rotation and calculating the coordinate again to be able to arrive at {X, Y}). The most interesting Ayllu´s routine is *Getto;* thus it allows solving obstruction in the road. If the coordinate {X,Y} is situated in a different room, where the robot is present, *Getto* can be able to arrive very easily and without any problems, or damage suffered by the robot. Finally, has been programmed the "adapted module" for executing the new voice-tasks. The Ayllu's routine *Getto* can be able to realice many taks, if the values for *X* and *Y* are modified.

The knowledge-based system experiments were carried out repeatedly in five phases: **1)** The Learning phase. **2)** The First Classification phase. **3)** The Reject phase. **4)** *The Adapting phase.* **5)** *The Second Classification* phase. The knowledge-based system proposal in this work uses an internal expert system for the language recognition: English and Spanish. The expert system allows you to identify the words text to be well understood and to avoid confusions from assigned label in the learning and classification phases.

3.1.- The Learning Phase for The Knowledge-Based System Model

In *Figure 1* the process used in the learning phase it is shown. This process consists of the incorporation of prototypes in the training sample and also in the knowledge base. In the training sample, we have incorporated the frames obtained from digital frequency. These frames were obtained with the procedure above-mentioned (To See Section 2.2). However, in the knowledge base, there have been added some text words captured by the keyboard concerning the voice captured by the microphone. Then the knowledge base is a small dictionary of question and answers that has routine words in the languages English and Spanish. For the incorporation of the prototype (frame) in the training sample, first the human voice by the microphone was captured; immediately it was saved in "wave" file extension. Then the wave file was opened to be applied the *Frames Selection Procedure*. Once labeled the frame, it was incorporated in the training sample and the text word spoken was added in the dictionary of the knowledge based, where the answers and questions are related with the vector's frame features. For the incorporation of the other prototypes it was necessary to repeat this process. The Learning phase was finalized when in the training sample many prototypes of different strata were added. For the different strata the voices from men and women are both considered. In *Table 1* the characteristics of the training sample resultant from the learning phase are shown. In *Table 1*, the incomplete training sample for the four well-known Ayllu´s routines is shown. The file of the training sample was named: VOICE-ROUTINES. However, in the *Table 3*, another characterization from the training sample: VOICE-ROUTINES (when the cross validation has been used) is shown. On the other hand, the first experiment above-mentioned for the first neural network (BAP) used in this work, did use the training sample VOICE-ROUTINES (*Table 1* and *Table 3*) in its learning phase to obtain the generalized weights that will be used in the *First Classification Phase*.

3.2.- The First Classification Phase for The Knowledge-Based System Model

The classification phase was carried out as follows: *Using Cross Validation.* In the dataset VOICE-ROUTINES (*Table 3*), five-fold cross validation was employed (80% for the training sample, 20% for a test set and 20% for an additional test set with new classes). In *Table 3*, you can observe that the prototypes for the classes: *5, 6, 7 and 8* in the *Training Sample* are not contained. However, in the *Test Sample* some prototypes of the new classes have been incorpored. In the *Table 5* the results obtained of the classification using the rules: 1-NN, k-NN, k-NCN and the first experiment with the neural network (BAP) are shown. It is observed in the *Table 5,* that the classification criterions are the geometric mean (*g*) [**17, 26, 30**] and the global accuracy [**17**, **26**]. We have considered very interesting to know the values by class in the accuracy, which allows you to observe the geometric mean criteria. It is important to observe as the value of the *(g)* shows 0.00 % in its performance. It happens because all the errors in the new classes (*5*, *6*, *7* and *8*) have been obtained; and the training sample does not contain these classes. If you can observe, the global accuracy low values have shown too (To See *Table 5*). The goal in this phase is to show that there are many errors in the patterns of the new classes, because the system still is not ready for working.

3.3.- The Reject Phase for The Knowledge-Based System Model

In this phase, all the new patterns have been "classified" or "rejected" employed some techniques above-mentioned and reported in [**19**]. If the new pattern is well classified, then its corresponding task is executed. Otherwise, the new pattern is "rejected" and it is saving in the "rejected traning sample". In *Table 6* the results for reject techniques are shown. If you can observe in *Table 6* the *2-NN* and *Modified 2-NN* techniques, they have rejected many patterns of the new and known classes. The ideal goal with reject techniques might be to reject just the patterns of the new classes, but it was not possible. However, the Averaged Distance techniques, they have rejected few patterns of the known classes and many patterns of the new classes. Therefore, we have considered optimal these Averaged techniques. In *Table 6* you can observe that the sum of the percentages for the rejected patterns in known and new classes, they do not show the *100%*, because some patterns have been well classified and accepted by the system. Finally, the new proposal *Manhattan Averaged Distance* has shown well performance in the known and new classes.

3.4.- The Adapting Phase for The Knowledge-Based System Model

Once finalized the reject phase, all the new patterns incorpored into “rejected training sample” are processed. First, we have employed *k-means* [**21**] algorithm with *k=4* for clustering the “rejected training sample”, thus we have four new classes. After, we have employed *Depuration* and *Depuration k-NCN* [**26**] techniques (by separated in two different experiments) for cleaning the “rejected training sample”, so we have obtained a “cleaned rejected training sample” (If the Averaged Distance NN has been used, then we employed *Depuration* technique. If Averaged Distance NCN has been used, then we employed *Depuration k-NCN* technique). The “rejected training sample” contains rejected patterns of all the classes (known and new classes). These rejected patterns in their majority are cleaned, but can still have some patterns of the known classes. The ideal goal with the “cleaned rejected training sample” is to contain only patterns of the new classes, but in this phase is not possible yet. Finally, all the patterns contained in the “cleaned rejected training sample” are incorpored into “original training sample” and we have obtained an “adapted training sample” that it will become in the “original training sample” when the “adapted training sample” has been deleted. In *Table 2,* the characterization of the adapted dataset with the new classes is shown. However, In *Table 4,* another characterization from the adapted training sample: VOICE-ROUTINES (when the cross validation has been used) is shown.

3.5. - The Second Classification Phase for the Knowledge-Based System Model

Before to begin this phase, we have employed the *Depuration* techniques again for cleaning the “adapted training sample”, which is the “original training sample” now. This second application of the *Depuration* techniques were carried out because might be that some patterns (in the “adapted original training sample”) of known classes are still not well labeled. Once they have been corrected, we have begun the second experiment with the neural network BPA (which it has *9* nodes in its output layer). The learning phase for the BPA, in this experiment with the “adapted original training sample” has been used. First, we have employed the cross validation for comparing with the results of the section *3.2*. In *Table 7* the results obtained of the second classification phase using the rules: 1-NN, k-NN, k-NCN and the second experiment with the neural network (BAP) are shown. It is observed in the *Table 7*, that the classification criterions are the geometric mean *(g)* and the global accuracy. How you can observe, the knowledge-based system is ready for working now, because the accuracy and geometric mean (g) criteria, they have shown well performance. In *Figure 2* the process used in the second classification phase (when the system is ready for working) it is shown. This process consists of the identification of new prototypes that arrive to the system. A microphone captures these prototypes. They can only be captured one by one, the system does not allow capturing two or more prototypes at the same time. For the identification of a new prototype (frame), first it was captured the human voice by the microphone, immediately was saved in "wave" file extension. Once captured the human voice, this is converted in frames. These frames are obtained with the procedure above-mentioned (section 2.2). Then the wave file was opened to be applied the Frames Selection Procedure. After, it is used the inference engine of the internal expert system to find in the knowledge base the text words corresponding to the new frame and to be able to do the language discrimination: Spanish or English. Once identified the language spoken, the classification rule (BPA, 1-NN, k-NN or k-NCN according to be the case) is used to assign the label to the new prototype. Since the new prototype has class identification (label) already it can execute the mobile robot simulator with the routine concerning to the assigned class. For the identification of the other new prototypes and the execution of Ayllu´s routines, it was necessary to repeat this process. Finally, the knowledge-based system for executing the new tasks, it reviews the “adapted module” (above-mentioned in section 3) and it employs *Getto* routine modifying the *X* and *Y* parameters. If you can observe in *Table 2*, the additional new tasks are four: *Left, Back, Right and Shutdown*. In *Left* and *Right* are only modified the *Y* value for *Getto*. In *Back X* value is modified. In *Shutdown* routine, the robot is power off (in simulation puts *X=0* and *Y=0*).

4. Conclusions and Future Works

The new proposal: *Manhattan Averaged Distance k-NCN* reject technique has obtained the best results in most of the cases. However, the Averaged Distance k-NN rule has obtained similar results in comparison with the new proposal. In addition, the neural network (BAP) has not shown to be very efficient for the incomplete training sample problem when its *learning phase* has not been executed again. Although the neural network (BAP) is very fast in the classification phase, in this work we observe that the execution times with the NN rules (k-NN and k-NCN) are as very similar as the neural network. Then, the execution times are very similar in all the techniques. On the other hand, ours experiements for Averaged Distance and Averaged Distance NCN techniques, also were carried out with training sample of real datasets taken from the UCI Repository [**31**]. In each dataset, five-fold cross validation was employed (80% for the training sample and 20% for a test set). We have deleted only one class for each iteration. Then, we experimented with *c* iterations for each incomplete dataset (e. g. If training sample has two classes, we employed the cross validation two times. First, we have deleted patterns of the class 0 from the TS and we have used the complete test set. Second, we have deleted patterns of the class 1 from the TS and we have used the complete test set). In Addition, the TS is incomplete, when the Test Set has all the classes. Results to be presented here after represent the averaged values of the five replications and *c* iterations for each dataset (To See *Table 8*). A future work is to use more techniques for clustering that to be able for detecting the number of cluster without having defined the *k* value beforehand. Another future work is to employ a *Radial Basis Function* Neural Network for comparing the result with Averaged techniques. Finally, will be necessary changing the mobile robot simulator, by a real application with the robot: Pioneer DX2 of ActivMedia.

References

[**1**].- Scarabino, JC (2000): "Sistemas Expertos: Aspectos técnicos", [en línea] 5campus.org, Inteligencia Artificial <http://www.5campus.org/leccion/sistexp at> [23/Mayo/2002].

[**2**].- Sangster A. and Wilson A. (1991). “Knowledge Based Learning”. Expert Systems for Information Management, Vol. 4, No 1, pp. 49-63.

[**3**].- D. E. Rumelhart and J.L. McClelland (1987) "Parallel Distributed Processing:Explorations in the Microstructure of Cognition". Vol I, II and III. Cambridge. MA: MIT Press.

[**4**].- A.Benallegue and A.Bennani-Hassa. (1993) “Adaptive controller for robot manipulators with elastic joints using passive feedback systems approach”. In *Proc. of the IMACS/IFAC 2nd Int. Symposium on Mathematical and Intelligent Models in System Simulation*, pages 12-16.

[**5**].- Barandela, R. and M. Juarez (2001). “Ongoing Learning for Supervised Pattern Recognition”. Submitted to SIBGRAPI-2001, Brazil.

[**6**].- De Lope Asiaín J. (1998). "Mobile Robot Simulator", [en línea]. Departamento de Sistemas Inteligentes Aplicados Universidad Politécnica de Madrid. <http://www.sia.eui. upm.es/~jdlope/mrs/worlds.shtml>

[**7**].- ActivMedia Robotics (2002). "Intelligent Mobile Robots & Bases". [En Linea] ActivMedia Robotics, LLC. <http://www.activrobots.com/ROBOTS/ index.html> [23/Mayo/2002].

[**8**].- ActivMedia Robotics (2002). "ActivMedia Robotics Interface for Applications". [En Linea] ActivMedia Robotics, LLC. <http://www .activrobots. com/ SOFTWARE/ index. html> [23/Mayo/2002].

[**9**].- ActivMedia Robotics (2002). “ Ayllu Downloads Distributed Behavior-Based Control Ayllu Manual v 1.5". [En Linea] ActivMedia Robotics, LLC. <http://activmedia.com/> [23/Mayo/2002].

[**10**].- Moreno L. J. (2000). "Desarrollo de un reconocedor de dígitos con distinción de énfasis: Capítulo II. CSLU ToolKit - Reconocimiento de voz basado en Frames". Universidad de las Américas-Puebla, Escuela de Ingeniería Departamento de Ingeniería en Sistemas Computacionales. <http:// mailweb. udlap.mx/~tesis/lopez_m_j/capitulo2.html> [23/Mayo/ 02].

[**11**].- Lleida S. E. (2000). "Reconocimiento Automático del Habla". Centro Politécnico Superior-Universidad de Zaragoza. <http:// www.gtc.cps.unizar.es/~eduardo/>[23/May/02]

[**12**].- Hosom, J.P., Cole R., and Fanty M. (1999). "Speech Recognition Using Neural Networks at the Center for Spoken Language Understanding". NSF Graduate Research Traineeships award (grant number 9354959) and the CSLU Member companies. <http://cslu.cse.ogi.edu/tutordemos/nnet_recog/recog.html> [23/Mayo/2002].

[**13**].- Cid, A. (1999). "Uso de las RNA para el reconocimiento de voz". International Sensible Artificial Intelligence Network of Thoughts. <http://www.intersaint.org/acid/rvpm5. htm> [23/Mayo/2002].

[**14**].- Carreira-Perpinan, M. A. (1995): “Compression neural networks for feature extraction: Application to human recognition from ear images”. MSc thesis, Faculty of Informatics, Technical University of Madrid, Spain.

[**15**].- J. Tani (1996)."Model-based Learning for Mobile Robot Navigation from the Dynamical Systems Perspective," IEEE Trans. On System, Man and Cybernetics Part B, Vol.26, No.3, pp. 421-436.

[**16**].- Renals, S. and Richard R. (1989). “Phoneme Classification Experiments Using Radial Basis Functions”. In Proceedings of the IEEE International Joint Conference on Neural Networks (IJCNN’89), 1, pp. 461-467.

[**17**].- Kubat, M. and S. Matwin. (1997). “Addressing the Curse of Imbalanced Training Sets: One-Sided Selection”. Proceedings of the 14th International Conference on Machine Learning, 179-186.

[**18**].- Barandela, R.,Sánchez JS.,García,V.,Rangel,E.(2001).“Fusion of techniques for handling the imbalanced training sample problem”. In: Procedings of 6th Ibero-American Symposium on Pattern Recognition, Brasil, 31-40.

[**19**].- Barandela, R. and T. Najera (2002). "Supervised Pattern Recognition with Incomplete Training Samples". Enviado a: Workshop on Statistical Pattern Recognition, Canadá.
[**20**].- Barandela, R. (1987). "The Nearest Neighbor rule: An empirical study of its methodological aspects." Ph. D. thesis, Institute of Cybernetics, Berlin.
[**21**].- Duda, R. O. and Hart P. E. (1973). "Pattern Classification and Scene Analysis". Wiley, New York.
[**22**].- Barandela, R., and E. Gasca (2000). "Decontamination of training samples for supervised pattern recognition methods". In: Advances in Pattern recognition, Lecture Notes in Computer Science, vol. 1876, F. Ferri et al. (eds.), Springer, Berlin, 621-630.
[**23**].- Cover, T. M. and Hart, P. E. (1967). "Nearest Neighbor Pattern Classification". IEEE Transactions on Information Theory, Volume IT-13, January, pages 21-27.
[**24**].- Fix, E. and Hodges, J. L., Jr. (1952). "Discrimination analysis: small sample performance". USAF School of Aviation Medicine, Randolph, Field, Tex. Project 21-49-004, Rept. 11, August.
[**25**].- Skalak, D. B. (1994). "Prototype and Feature Selection by Sampling and Random Mutation Hill Climbing Algorithms". In: Proceedings of the Eleventh International Conference on Machine Learning (ML94). Morgan Kaufmann, pp. 293-301.
[**26**]. - Rangel, E. (2002). "Vecinos Envolventes para Variantes de la Regla del Vecino más Cercano". Ph.D. Thesis, Instituto Tecnológico de Toluca, México.
[**27**]. - T. Kohonen (1995). Self-Organizing Maps, Springer-Verlag: Berlin, Germany.
[**28**]. - Chaudhuri, B. B. (1996). "A New definition of neighborhood of a point in multi-dimensional space". Pattern Recognition Letters 17, 11-17.
[**29**]. - J.S. Sánchez, F. Pla and F.J. Ferri (1997). "On the use of neighborhoods-based non-parametric classifier". Pattern Recognition Letters, Vol. 18, pp. 1179-1186.
[**30**].- Rangel, E. and García, O. (2002). "Nearest Centroid Neighbour, An Alternative in the Speech Recognition for the Execution from a Mobile Robot Simulator". Procedings of 9th International Congress On Computer Science Research (CIICC´02), pp. 141-152, October 23-25-Puebla(México),2002.ISBN: ISBN-970-18-8526-0.
[**31**].- Merz C.J. and P.M. Murphy. (1998). "UCI Repository of Machine Learning Databases", University of California at Irvine, Departament of Information and Computer Science. http://www.csi.uci.edu/ ~mlearn

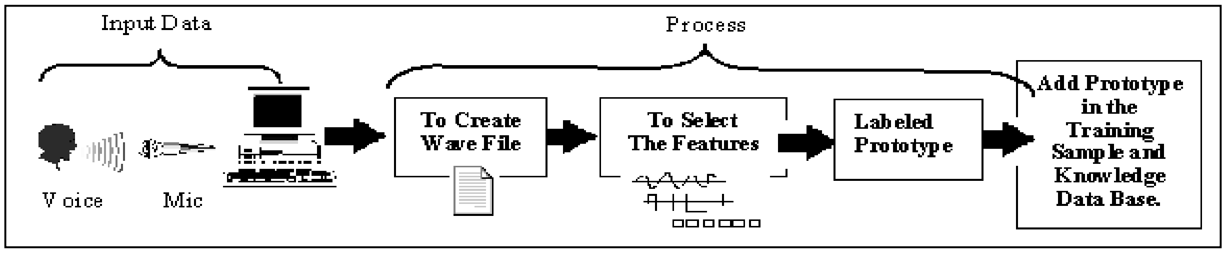


Figure 1. Learning Phase

TABLE 1. CHARACTERIZATION OF THE ORIGINAL DATASET.

VOICE-ROUTINES			
#of Features		20	
#of Classes		5	
# Global of prototypes (Frames)		1000	
Class	# prototypes by class	Routine to execute	
		English	Spanish
0	200	StraightLine	LineaRecta
1	200	Wanderer	Rotacion
2	200	Goto	Ir_a
3	200	Getto	Resuelve_Ir_a
4	200	Empty	Line_In

TABLE 6. THE GLOBAL % OF REJECTED PATTERNS IN THE "REJECT PHASE" IS SHOWN. FIGURE IN PARENTHESIS THE STANDARD DESVIATION.FOR 2-NN AND MODIFIED 2-NN TECHNIQUES *T=0.25* HAS BEEN EMPLOYED.

REJECT TECHNIQUE	VOICE-ROUTINES	
	Rejected for Known classes	Rejected for New classes
2-NN Reject Technique	45.12% (0.15)	50.88%(0.12)
Modified 2-NN Reject Technique	43.30% (0.23)	54.70% (0.2)
Averaged Distance k-NN	32.40% (5.23)	23.24% (9.6)
Manhattan Averaged Distance k-NN	30.82% (6.65)	24.72%(10.2)
Averaged Distance k-NCN	30.89% (6.56)	35.34% (9.8)
Manhattan Averaged Distance k-NCN	**27.22%** (7.08)	**37.78%**(10.9)

TABLE 2. CHARACTERIZATION OF THE ADAPTED DATASET (WITH NEW CLASSES).

VOICE-ROUTINES			
# of Features		20	
# of Classes		9	
# Global of prototypes (Frames)		1800	
Class	# prototypes by class	Routine to execute	
		English	Spanish
0	200	StraightLine	LineaRecta
1	200	Wanderer	Rotacion
2	200	Goto	Ir_a
3	200	Getto	Resuelve_Ir_a
4	200	Empty	Line_In
5	200	Back	Regresar
6	200	Right	Derecha
7	200	Left	Izquierda
8	200	Shutdown	Apagar

TABLE 3. CHARACTERIZATION OF THE DATASET FOR *TABLE 1*, WHEN THE CROSS VALIDATION HAS BEEN EMPLOYED.

VOICE-ROUTINES				
# of Features			20	
# of Classes			5	
# Global of prototypes (Frames)			1000	
# prototypes by class				
Training Sample				
Class 0	Class 1	Class 2	Class 3	Class 4
160	160	160	160	160
Test Sample				
Class 0	Class 1	Class 2	Class 3	Class 4
40	40	40	40	40
New Classes only for Test Sample				
Class 5	*Class 6*	*Class 7*	*Class 8*	
40	40	40	40	

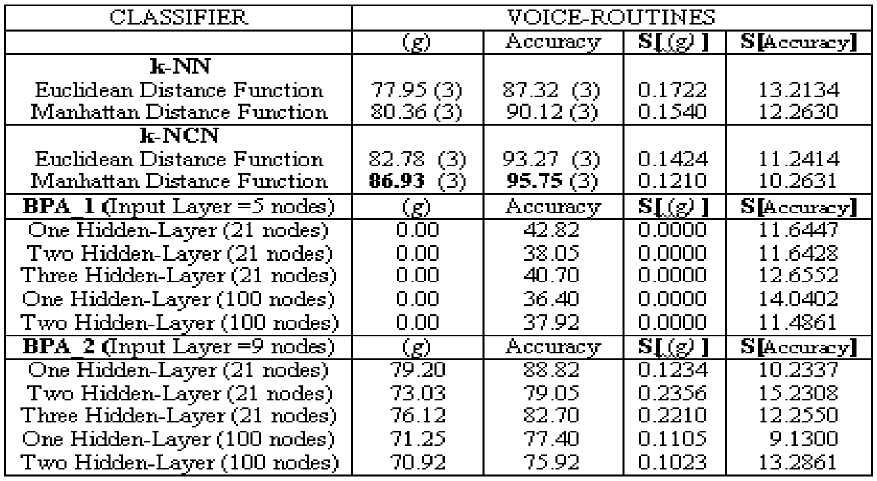
TABLE 4. CHARACTERIZATION OF THE DATASET FOR *TABLE 2*, WHEN THE CROSS VALIDATION HAS BEEN EMPLOYED.

VOICE-ROUTINES				
# of Features			20	
# of Classes			9	
# Global of prototypes (Frames)			1800	
# prototypes by class				
Training Sample				
Classes Added in Learning Phase				
Class 0	Class 1	Class 2	Class 3	Class 4
160	160	160	160	160
Classes Detected in Reject Phase				
Class 5	*Class 6*	*Class 7*	*Class 8*	
160	160	160	160	
Test Sample				
Classes Added in Learning Phase				
Class 0	Class 1	Class 2	Class 3	Class 4
40	40	40	40	40
Classes Detected in Reject Phase				
Class 5	*Class 6*	*Class 7*	*Class 8*	
40	40	40	40	

TABLE 5. CLASSIFICATION PHASE WITH A VARIANT 1. IN THIS TABLE THE ACCURACY IN TWO CRITERIONS ARE SHOWN. ALSO, THE STANDARD DESVIATION **S[.]** FOR EACH CRITERION ARE SHOWN. FIGURE IN PARENTHESIS THE k VALUE USED. THE BPA HAS *5* NODES IN ITS OUTPUT LAYER AND IT HAS *20* NODES IN ITS INPUT LAYER.

CLASSIFIER	VOICE-ROUTINES			
	(*g*)	Accuracy	**S[**(*g*)**]**	**S[**Accuracy**]**
k-NN				
Euclidean Distance Function	0.00 (3)	47.28 (3)	0.0000	11.4238
Manhattan Distance Function	0.00 (3)	50.81 (3)	0.0000	10.4730
k-NCN				
Euclidean Distance Function	0.00 (3)	48.17 (3)	0.0000	10.4828
Manhattan Distance Function	0.00 (3)	**65.15** (3)	0.0000	9.4731
Multi-Layer Perceptron BackPropagation (BPA)	(*g*)	Accuracy	**S[**(*g*)**]**	**S[**Accuracy**]**
One Hidden-Layer (21 nodes)	0.00	42.82	0.0000	11.6447
Two Hidden-Layer (21 nodes)	0.00	38.05	0.0000	11.6428
Three Hidden-Layer (21 nodes)	0.00	40.70	0.0000	12.6552
One Hidden-Layer (100 nodes)	0.00	36.40	0.0000	14.0402
Two Hidden-Layer (100 nodes)	0.00	37.92	0.0000	11.4861

TABLE 7. IN THIS TABLE WELL PERFORMANCE IS SHOWN FOR THE SECOND CLASSIFICATION PHASE. THE ACCURACY IN TWO CRITERIONS ARE SHOWN. ALSO, THE STANDARD DESVIATION S[.] FOR EACH CRITERION ARE SHOWN. FIGURE IN PARENTHESIS THE k VALUE USED.



CLASSIFIER	VOICE-ROUTINES			
	(g)	Accuracy	S[(g)]	S[Accuracy]
k-NN				
Euclidean Distance Function	77.95 (3)	87.32 (3)	0.1722	13.2134
Manhattan Distance Function	80.36 (3)	90.12 (3)	0.1540	12.2630
k-NCN				
Euclidean Distance Function	82.78 (3)	93.27 (3)	0.1424	11.2414
Manhattan Distance Function	**86.93** (3)	**95.75** (3)	0.1210	10.2631
BPA_1 (Input Layer =5 nodes)	(g)	Accuracy	S[(g)]	S[Accuracy]
One Hidden-Layer (21 nodes)	0.00	42.82	0.0000	11.6447
Two Hidden-Layer (21 nodes)	0.00	38.05	0.0000	11.6428
Three Hidden-Layer (21 nodes)	0.00	40.70	0.0000	12.6552
One Hidden-Layer (100 nodes)	0.00	36.40	0.0000	14.0402
Two Hidden-Layer (100 nodes)	0.00	37.92	0.0000	11.4861
BPA_2 (Input Layer =9 nodes)	(g)	Accuracy	S[(g)]	S[Accuracy]
One Hidden-Layer (21 nodes)	79.20	88.82	0.1234	10.2337
Two Hidden-Layer (21 nodes)	73.03	79.05	0.2356	15.2308
Three Hidden-Layer (21 nodes)	76.12	82.70	0.2210	12.2550
One Hidden-Layer (100 nodes)	71.25	77.40	0.1105	9.1300
Two Hidden-Layer (100 nodes)	70.92	75.92	0.1023	13.2861

TABLE 8. – AVERAGED DISTANCE TECHNIQUES EMPLOYING K=2. IN THIS TABLE THE % OF REJECTED PATTERNS OF THE KNOWN CLASSES AS NEW CLASSES ARE SHOWN. AFTER FROM ± THE STANDARD DESVIATION IS SHOWN.

Real Dataset	**AVERAGED DISTANCE REJECT TECHNIQUES EMPLOYING k=2**							
	AVERAGED DISTANCE		**MANHATTAN AVERAGED DISTANCE**		**AVERAGED DISTANCE NCN**		**MANHATTAN AVERAGED DISTANCE NCN**	
	Known Classes	New Classes	Known Classes	New Classes	Known Classes	New Classes	Known Classes	New Classes
Cancer	8.34 ± 0.60	35.46 ± 47.7	6.18 ± 3.45	35.10 ± 48.2	8.27 ± 1.32	35.8 ± 47.2	6.33 ± 3.66	35.39 ± 47.8
Heart	16.48 ± 4.44	20.00 ± 5.76	17.03 ± 6.80	24.25 ± 6.54	14.81 ± 5.75	19.07 ± 1.83	16.11 ± 8.64	22.59 ± 5.23
Liver	14.78 ± 3.69	17.53 ± 9.22	16.08 ± 3.89	17.82 ± 8.81	13.62 ± 2.46	16.81 ± 8.60	15.79 ± 4.71	17.53 ± 9.22
Pima	17.45 ± 7.67	23.07 ± 2.12	18.82 ± 7.21	25.09 ± 5.73	17.97 ± 8.59	22.74 ± 1.48	18.95 ± 8.87	24.77 ± 4.34
Sonar	24.39 ± 0.68	41.21 ± 1.71	23.90 ± 1.37	41.70 ± 1.03	22.68 ± 0.34	41.70 ± 1.72	22.92 ± 1.37	42.92 ± 0.69
Vehic	54.94 ± 9.80	19.84 ± 3.75	53.42 ± 10.1	20.02 ± 4.39	59.59 ± 6.84	21.12 ± 3.74	56.52 ± 9.68	20.82 ± 3.55
Wine	38.62 ± 6.45	27.25 ± 6.28	33.13 ± 9.88	27.45 ± 6.61	38.43 ± 9.38	28.62 ± 5.95	31.17 ± 8.66	28.23 ± 7.34
Iris	21.33 ± 7.57	32.66 ± 1.15	22.88 ± 7.89	32.66 ± 1.15	22.00 ± 8.96	32.44 ± 1.01	20.66 ± 8.81	32.44 ± 1.01
Bridge	68.57 ± 15.6	11.57 ± 11.6	57.89 ± 10.9	11.12 ± 12.5	67.51 ± 14.55	11.72 ± 12.1	51.27 ± 8.30	10.37 ± 12.8
Image	41.49 ± 3.47	13.97 ± 9.20	38.88 ± 2.50	13.37 ± 8.74	41.95 ± 3.06	13.83 ± 9.05	39.96 ± 2.99	13.90 ± 9.20
Zoo	42.31 ± 2.50	14.28 ± 15.9	39.74 ± 2.19	14.28 ± 15.9	40.89 ± 2.56	14.28 ± 15.9	36.89 ± 2.41	14.28 ± 15.9
Glass	52.91 ± 15.6	9.16 ± 14.2	54.25 ± 14.6	9.58 ± 14.1	57.41 ± 15.94	9.83 ± 14.7	59.16 ± 17.1	9.83 ± 14.6
----------	----------	----------	----------	----------	----------	----------	----------	----------
Averaged	33.47 ± 6.51	22.17 ± 10.7	31.85 ± 6.74	22.70 ± 11.1	33.76 ± 6.65	22.33 ± 10.2	**31.31** ± 7.10	**22.76** ± 10.9

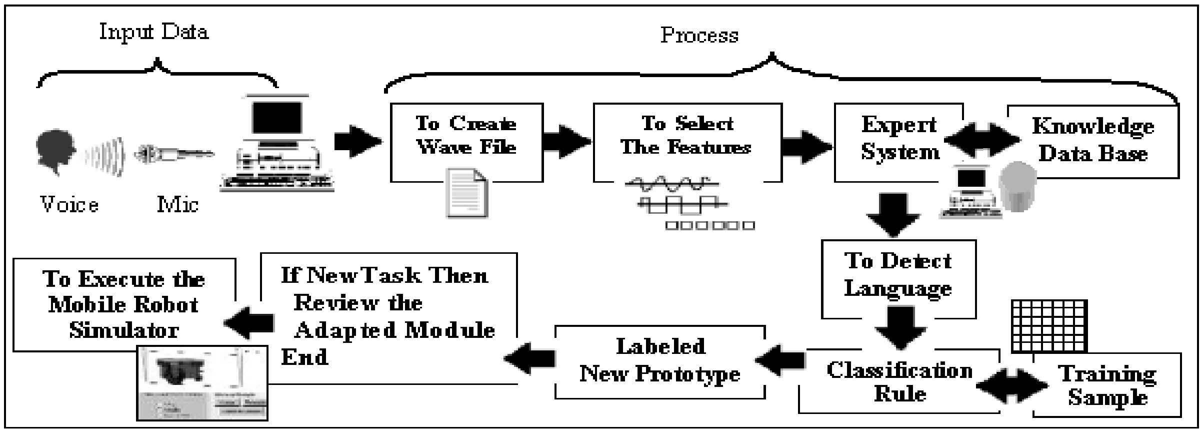


Figure 2. The Second Classification Phase