A 1-NN Preclassifier for Fuzzy k-NN Rule

Draft text for: *Proc. 13th Int. Conf. Pattern Recognition*, pages D-234 - D-238, Wien, Austria, Aug 25-29, 1996. IAPR, Techn. Univ. Vienna.

Adam Jóźwik
Institute of Biocybernetics and Biomedical Engineering, PAS
Trojdena 4, 02-109 Warsaw, Poland adamj@ibbrain.ibb.waw.pl

Leszek Chmielewski Waldemar Cudny Marek Skłodowski
Association for Image Processing, Inc., Poland, and Institute of Fundamental Technological Research, PAS
Świętokrzyska 21, 00-049 Warsaw, Poland {lchmiel,wcudny,msklod}@ippt.gov.pl

Abstract

A two-stage classification scheme is presented. In the first stage it is decided which of the two rules, 1-NN or fuzzy k-NN, will be used in the second stage. The proposed approach leads to significant acceleration of the learning process as well as the classification phase.

1. Introduction

The *NN* rules and their modifications belongs to the most powerful methods of statistical pattern recognition. There exist many publications devoted to methods based on the *NN* idea. Overview of large part of these methods may be found in [2]. The most popular however, is the *k-NN* rule.

To construct the *k-NN* classifier the distance measure and the best number k of *NN*s must be chosen. As a distance the Euclidean or the "city" measure can be used. To select the best k it is necessary to calculate a probability of misclassification for $k = 1, 2, ..., m$, where m is a number of objects in the training set, called also a learning or a reference set. The k which offers the minimum misclassification probability should be taken as an optimum.

The misclassification probability can be estimated from the learning set by the "leave one out" method. It consists in classification of each of the m objects from the training set by the *k-NN* rule using the set of remaining $m-1$ objects as a reference set. For the *k-NN* rule, this can be realized extremely easy. If the learning set contains m objects then the "leave one out" method costs exactly the same number of computations as an estimation of the misclassification rate which uses the training set of $m-1$ objects and the test set containing m objects.

The classical *k-NN* rule can be replaced by the fuzzy *k-NN* rule introduced in [7] and later discussed in [1].

Now, some brief information about the fuzzy *k-NN* rule will be given. The membership value of an object in the class i can be written as a binary vector $v_b = [0, 0, ..., 1_i, ..., 0_{nc}]$, where the value 1 appears in the i-th position and nc denotes the number of classes. The classical *k-NN* rule consists now in calculating the mean vector $v_f = [k_1/k, k_2/k, ..., k_i/k, ..., k_{nc}/k]$ of all binary membership vectors that correspond to the k nearest neighbors. The symbol k_i denotes the number of objects from the class i present among the k nearest neighbors. So, $k_1 + k_2 + ... + k_i + ... + k_{nc} = k$. The classified object is assigned to the class i which corresponds to the highest value of ki/k. It is easy to notice that the class membership vectors do not have to be binary. Similarly as the vector v_f, they can assume the fuzzy form. In this way, the fuzzy *k-NN* rule has been defined. The definition of the *k-NN* rule in the above-mentioned way is more general but offers no special advantage in the case when crisp (nonfuzzy) input and output is considered, which is the case we are going deal with. The advantage can be more apparent, if one applies the learning process for the fuzzy *k-NN* rule, described below.

The learning of the fuzzy *k-NN* rule generates a series of trials

$$(W_0, k_0, er_0), (W_1, k_1, er_1), ..., (W_j, k_j, er_j), ... \quad (1)$$

where W_0 is a primary binary membership matrix with m rows and nc columns, k_0 is the optimum number of *NN*s and er_0 is the error rate offered by the k_0-*NN* rule and calculated by the "leave one out" method. W_1 is a fuzzy membership matrix obtained by reclassifying all the objects in the learning set by *$(k_0 + 1)$-NN* rule. Each of the classified objects appears also in the reference set. That is the reason why $k_0 + 1$ instead of k_0

is taken. Next, for W_1, the values k_1 and er_1 can be found, also by the "leave one out" method. To calculate the error rates er_j the crisp outputs of the fuzzy *k-NN* rule are taken into account. The learning stops when $er_j + 1$ becomes greater than er_j or it is equal to er_j and $k_j + 1$ is greater than k_j, since a smaller value of k is preferred. Finally, the fuzzy k_j-*NN* rule with the fuzzy membership matrix W_j and crisp decision is used. It offers the error rate equal to er_j. More detailed description of the learning process for the fuzzy *k-NN* rule can be found in [1, 7].

The objects represented in the training set and the objects to be classified may be described by different units. For this reason the data ought to be standardized. For this purpose the following equation can be used: $x[i,j] := (x[i,j] - mv[j])/sd[j]$, where $x[i,j]$ is the value of j-th feature for the i-th object, $mv[j]$ is the mean value of the j-th feature and $sd[j]$ is its standard deviation. The values $mv[j]$ and $sd[j]$ are derived only from the training set. We shall use this equation as it assigns equal weights to all features.

2. The 1-NN preclassifier

Let us assume that the reference set X consists of nc subsets: $X_1, X_2, ..., X_{nc}$, and each of them contains objects corresponding to only one class. With these sets we associate certain positive real numbers $e_1, e_2, ..., e_{nc}$ defined in the manner given below:

$$e_i = \max_{x_j \in X_i} d(X_i - x_j, x_j)\,, \tag{2}$$

where $d(.,.)$ denotes a distance function. We also define areas $A_1, A_2, ..., A_{nc}$:

$$A_i = \{x : d(X_i, x) \leq e_i\}\,. \tag{3}$$

Now, we can formulate the *1-NN* classification rule for a preclassifier. The object x is classified by the *1-NN* rule if and only if it belongs exactly to one of the areas A_i. If x does not belong to any the areas $A_i, i = 1, 2, ..., nc$, then the classification is refused. When x belongs simultaneously to some of the areas A_i, then the object is classified by the fuzzy *k-NN* rule. So, the preclassifier decides which kind of the two classification rules will be applied to form the final decision.

Let us denote by A the set of all objects from the reference set that belong to at least two areas $A_j, j = 1, 2, ..., nc$. It is strongly recommended to perform the feature selection to minimize the size of the set A. As far as feature selection strategy is concerned we will apply the forward and backward feature selection strategies [3] and choose the one which gives a better result. If the number of features is small then the full review of all possible feature combinations is recommended.

3. The (1-NN, fuzzy k-NN) rule

The *1-NN* preclassifier presented above recognizes whether the classified object belongs to only one of the areas $A_i, i = 1, 2, ..., nc$, appears in the intersection of some A_i, or it is outside of each A_i. If it lies exactly in one of A_i then the final decision is created by the *1-NN* rule. In the last case the "I don't know" decision is assumed. The object that lies in the intersection of some A_i will be classified by the fuzzy *k-NN* rule which requires the learning session. The classification rule obtained in this way, using the *1-NN* preclassifier, we will call the (*1-NN*, fuzzy *k-NN*) rule.

The classification quality of the (*1-NN*, fuzzy *k-NN*) rule can be estimated by the "leave one out" method. By virtue of Eqs. (1) and (3), all the objects from the training set that do not belong to the previously defined set A will be correctly classified. The decisions will be assigned by *1-NN* rule. The objects, misclassified during the realization of the "leave one out" method, must be from the set A. They will be classified by the fuzzy *k-NN* rule. Thus, the learning scheme (1) can be constrained to the these rows of the matrices $W_0, W_1, ..., W_j, ...$ which correspond to the objects appearing in the set A. Generally, the (*1-NN*, fuzzy *k-NN*) rule is not equivalent to the fuzzy *k-NN* rule. The equivalence holds place only when the set A contains all the objects from the training set.

We recommend two feature selection sessions: one for the preclassifier to minimize the size of the previously mentioned set A, and another one to minimize an error rate for the fuzzy *k-NN* rule.

4. The illustrating example

At present, we shall illustrate the (*1-NN*, fuzzy *k-NN*) rule described in previous sections using the well known IRIS data, used for the first time in [4]. Since, we had no access to this original paper, we have taken the data from [6]. The IRIS data consists of three classes: 1. iris setosa, 2. iris versicolor, and 3. iris virginica, and contains by 50 objects from each class. The objects are described by four features: 1. length of leaf, 2. width of leaf, 3. length of petal, 4. width of petal.

The set A, defined in the section 2, contained 58 objects in the case when all 4 features were used. After the feature selection performed by the full review of all possible combinations of features, the size of the set A,

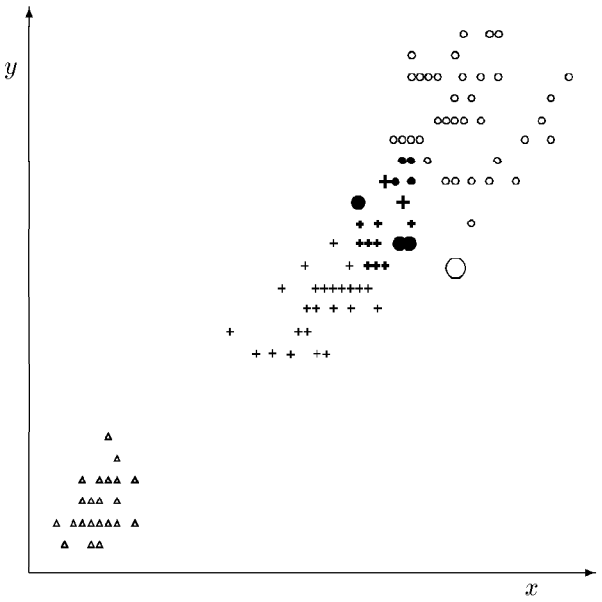
matrix		W_0		W_1		W_2	
obj	cla	2	3	2	3	2	3
53	2	1.00	0.00	0.71	0.29	0.71	0.29
57	2	1.00	0.00	0.86	0.14	0.86	0.14
71	2	1.00	0.00	0.29	0.71	0.29	0.71
73	2	1.00	0.00	0.71	0.29	0.71	0.29
78	2	1.00	0.00	0.43	0.57	0.36	0.64
84	2	1.00	0.00	0.71	0.29	0.71	0.29
86	2	1.00	0.00	0.86	0.14	0.86	0.14
107	3	0.00	1.00	0.86	0.14	0.86	0.14
120	3	0.00	1.00	0.71	0.29	0.71	0.29
124	3	0.00	1.00	0.29	0.71	0.29	0.71
127	3	0.00	1.00	0.29	0.71	0.29	0.71
128	3	0.00	1.00	0.29	0.71	0.29	0.71
134	3	0.00	1.00	0.71	0.29	0.71	0.29
139	3	0.00	1.00	0.29	0.71	0.29	0.71
150	3	0.00	1.00	0.14	0.86	0.22	0.78
opt. k_i		7		2		11	
error		3.30%		3.33%		0.00%	

for the two features, 3 and 4, was 29 objects. These objects have been shown in the figure, where features 3 and 4 play the role of x-axis and y-axis, respectively. Not all of the objects can be seen since some of them are covered by others. As a criterion for the feature selection, the minimum number of objects in the set A was taken. The set A contains the objects from the training set which lie in the class overlap areas, i.e., in the intersections of two or more areas A_i, determined according to the formulas (2) and (3). The size of the training set is 150 objects. Hence, thanks of the preclassifier, approximately 29 out of 150 (19.3%) objects will be recognized by the fuzzy *k-NN* rule. All the remaining objects can be classified by the *1-NN* rule.

It was discovered that all these 29 objects were found in the intersection of A_2 and A_3. The intersections of A_1 and A_2 as well as of A_1 and A_3 were empty. Thus, in the second stage it is sufficient to deal with the two-class problem. The fuzzy *k-NN* classifier must decide between the class 2 and 3.

Let us now consider the estimation of the misclassification rate by use of the "leave one out" method. By virtue of the formulas (2) and (3) all the objects that do not belong to the set A will be correctly classified. There are 121 $(150 - 29)$ such objects. Thus, the objects that may be misclassified are those which form the set A.

It is well known that the feature selection not only simplifies classification but also improves the results. Therefore, it is worthwhile to perform the feature selection also for the fuzzy *k-NN* rule. Similarly as in the case of the *1-NN* preclassifier, we decided to use the



full review of all possible feature subsets. As a criterion, the misclassification rate for the classical *k-NN* rule with the optimum number k_{opt}, chosen also by the application of the "leave one out" method was used. Again, as in the case of the preclassifier, the same two features, 3 and 4, were selected. For all the features 7 objects out of 29 were misclassified. The two selected features, mentioned above, offered 5 misclassifications.

Now, let us discuss the result of the learning process for the fuzzy *k-NN* rule. The sequence (1) was constrained to the objects belong to the set A. The table given above shows the behaviour of the learning process. To reduce the size of this table only the rows which changed the binary form into the fuzzy one have been presented.

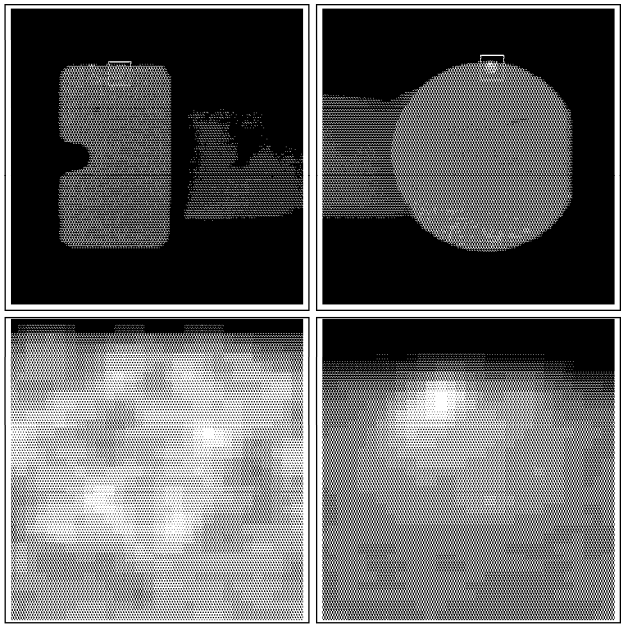
The same number of objects were misclassified for the membership matrices W_0 and W_1 when the "leave one out" method was applied. In the both cases these were exactly the same 5 objects. They are presented in the figure by medium size symbols. The matrix W_1 is preferred since it requires the smaller number k of *NN* (2 instead of 7). The matrix W_2, with *11-NN* rule offers the best result. The value of the estimated error rate equals to 0.

It is worth checking what the result would be if the fuzzy *k-NN* rule without the preclassifier were applied to the same data. It appeared that more learning steps were required to reach to ideal result with the error rate equal to 0. This means that the trail (W_3, k_3, er_3) needed to be calculated. The value of k_3 equaled 11. After generating (W_2, k_2, er_2) there was one incorrectly classified object. It is marked in the figure by the largest circle. It is interesting that this object comes outside of the set A.

5. Example 2

The example analyzed in the previous section was very favourable since not all the classes were represented in the overlap area. Now, we shall describe a more practical pattern recognition problem concerning quality control in production of ferrite cores.

Ferrites are ceramic magnetic materials made of oxide powders. Ferrite cores are manufactured by compacting and sintering these powders, and by grinding some surfaces to required dimensions. Certain surface irregularities emerge in the manufacturing process: chippings, ragged edges, pull-outs, cracks, scratches and discolourations [5]. Two ground surfaces of one of the core types are shown in the upper figures ($512 * 512$ pixels). A pull-out is visible in a marked window of the left one, and a chipping in the window of the right one. Magnified views of the windows (32*32 pixels, which correspond to approx. $1.5 * 1.5$ mm) are shown in the lower figures.



In visual inspection of core surfaces one of the approaches is to recognize whether a given pixel of an image of a core belong to a "good" surface, or to one of the classes corresponding to above listed types of irregularities. Basing on the result of such classification carried out for all pixels, a global quality measure can be assessed. In this example, an object is a pixel.

Various features can be used in such classification. It occurred that good results can be obtained with raw image brightnesses in the neighbourhood (square mask centred at the pixel) used as features. We have extended that idea by taking the following features: (a) raw brightnesses in a mask; (b) brightnesses in the mask rotated according to a locally dominating direction of image texture [8]; (c) statistical moments of brightnesses in a mask, up to a prescribed order; (d) same as (c), but in a rotated mask; (e) modulus of gradient. Further textural features will be applied in future.

The mask sizes used were: $5 * 5$ for features (a) and (b), and $3 * 3$ and $5 * 5$ for features (c), (d) and (e). Moments of order 2 and 3 were taken. A total of 60 features were obtained: 1 to 25 – brightnesses in the mask, by rows; 26 to 50 – brightnesses in the rotated mask; 51 to 54 – moments: mask $3 * 3$, order 2 and 3; mask $5 * 5$, order 2 and 3; 55 to 58 – moments as above, but in the rotated masks; 59 and 60 – gradients in masks $3 * 3$ and $5 * 5$. From these features, which were obviously too many, the most significant ones were found with the *k-NN* methodology.

From other considerations it results that a technically acceptable training set should contain at least about $10,000$ objects (pixels). Here, we present a simplified, pilot example with only three classes: chipping, pull-out, no defect, and with only 312 patterns in each class (total 936).

As we have recommended in the end of the section 2 the feature selection was performed for the preclassifier to minimize the number of objects in the class overlap area, i.e. the size of the set A. The number of features was too large to apply the review of all their possible combinations. We decided to use separately forward and backward feature selection strategies and to take the best result. It was obtained for 5 features: $(15, 26, 30, 35, 45)$ which were found by forward feature selection strategy. There were 580 objects out of 936 in the set A. When all 60 features were used then the set A contained 932 objects.

In the next step we applied the "leave one out" method for *k-NN* rule and forward and backward feature selection strategies to minimize the number of misclassified objects from the set A. As a reference set still the whole set of 936 objects was taken. For each of the reviewed feature subset always the optimum number k of *NN*s was considered. It appeared that with the use of 8 features $(2, 5, 18, 19, 21, 26, 43, 60)$ and *1-NN* rule only 5 objects were misclassified in the "leave one out" method. So, the final misclassification rate was 0.534%. Without feature selection it would be 1.82%. What is interesting, the same results could be reached directly by use of the same 8 features omitting the application of the preclassifier. The usefulness of the preclassifier is in such an approach rather questionable.

The second stage can be realized in another manner. We can establish which of the above mentioned 580 objects lie in the overlap area A_{123} of all 3 classes, which in an intersection of areas A_{12}, A_{13} and A_{23} of

all possible pairs of the three classes. Thus, 4 separate classification problems were obtained. The area A_{123} contained 100 objects and all of them were perfectly recognized in the "leave one out" method by use of only 2 features $(3, 59)$ and *14-NN* rule. For the area A_{12} containing 252 objects, the same above mentioned 8 features were selected and the *1-NN* rule chosen as the optimum. Only 4 objects were misclassified, 2 from the class 1 and 2 from the class 2. The area A_{13} contained 6 objects and one feature (15) with *20-NN* rule offered the perfect classification. The last area A_{23} with 222 objects required 3 features $(21, 40, 45)$ with a *2-NN* rule for perfect classification. Thus, the only misclassified objects were those 4 above mentioned. This means that this time the final misclassification rate was 0.427%. The improvements are slight. However, this is not an only advantage of applying the preclassifier.

Let us calculate the average number of features required in our problem for the classification of the one object, with and without preclassifier. The "leave one out" method will form a basis for these considerations.

The 5 features $(15, 26, 30, 35, 45)$ selected for the preclassifier were sufficient to classify 356 objects out of 936. So, $5 * 356$ feature measurements are needed. The 100 objects from the area A_{123} required additional 2 features $(3, 59)$, i.e. $2 * 100$ features, 252 objects from the area A_{12} were classified by use of 8 features $(2, 5, 18, 19, 21, 26, 43, 60)$, but the feature 26 was already measured for the preclassifier, and therefore we obtain $7 * 252$ additional feature measurements. The objects from the A_{13} do not require new features since the only needed feature (15) was already used by the preclassifier. The 222 objects from the last considered area A_{23} that should be classified by use of 3 features $(21, 40, 45)$ need additional $2*222$ feature measurements. One of this features: (45), was already required by the preclassifier. Finally, the average number of the feature measurements equals: $(5 * 356 + 2 * 100 + 7 * 252 + 2 * 222)/936 = 5$. Without the preclassifier the number of required feature measurements is 8.

It is worth noting that raw brightnesses and brightness gradients appeared to be more informative in this example than statistical moments, which were not selected as features for recognition.

6. Concluding remarks

The two presented examples show that various kinds of advantages can be obtained by the use of the *1-NN* preclassifier. In the first, illustrating example the features selected for the preclassifier and the main classifier were the same. We saved the number of computations required for learning of the fuzzy *k-NN* rule and the time required for final object classification.

Both these advantages were reached by the application of different classification rules for "easy" and "difficult" objects, where an "easy" object denotes an object which belongs to only one class area A_i, and a "difficult" object is that lying in the class overlap area.

The industrial example have shown that we can reduce the number of computations required for the learning stage as well as for the classification stage by decreasing the average number of measured features. In this case learning comprised the determination of the feature set and the optimum number k of *NN*s. No fuzzy membership matrices were necessary for *k-NN* rules in this case. The preclassifier made it possible to split the global classification problem into tasks of smaller sizes.

Acknowledgement This research has been supported by COPERNICUS programme within the project *CRAck and SHape Defect Detection in Ferrite Cores (CRASH)*, No. COP-94 00717. The presented part of results has been realized by the Institute of Biocybernetics and Biomedical Engineering, Polish Academy of Sciences, and the Association for Image Processing, Inc., Poland.

References

[1] J. C. Bezdek, S. K. Chuan, and D. Leep. Generalized k-NN rules. *Fuzzy Sets and Systems*, 18:237–256, 1986.

[2] B. V. Dasarathy. Nearest Neighbor (NN) norms: NN pattern classification techniques. *IEEE Computer Society Press*, 1995.

[3] P. A. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach.* Prentice-Hall, 1982.

[4] R. A. Fischer. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7, 1936.

[5] F. Fontana, M. Mari, D. Chetverikov, M. Lugg, T. Postupolski, A. Jóźwik, M. Skłodowski, M. Nieniewski, W. Cudny, and L. Chmielewski. CRAck and SHape defect detection in ferrite cores. 2nd Technical Progress Report of the Project CRASH, CRASH Consortium, Feb 1996.

[6] K. Jajuga. *The Statistical Pattern Recognition Theory (in Polish).* PWN, Warsaw, 1990.

[7] A. Jóźwik. A learning scheme for a fuzzy k-NN rule. *Pattern Recognition Letters*, 1:287–289, 1983.

[8] G. Z. Yang, P. Burger, D. N. Firmin, and S. R. Underwood. Structure adaptive anisotropic filtering for magnetic resonance image enhancement. In V. Hlaváč and R. Šára, editors, *Proc. 6th Int. Conf. Computer Analysis of Images and Patterns (CAIP'95)*, volume 970 of *Lecture Notes on Computer Science*, pages 384–391, Prague, Czech Republic, Sep 6-8, 1995. Springer Verlag, 1995.