Diederich, J., Fortuner, R. 1998. Classification Using Small Fuzzy Biological Data Sets, 1998. IEEE International Conference on Fuzzy Systems, May 4-9, 1998, Anchorage, Alaska.

Classification Using Small Fuzzy Biological Data Sets

Jim Diederich
University of California
Department of Mathematics
Davis, CA 95616
dieder@math.ucdavis.edu

Renaud Fortuner
Scientific consultant
Correspondant du Muséum National d'Histoire
Naturelle, Paris
17130 Montendre, France
fortuner@wanadoo.fr

Abstract

In this paper we examine fuzzy methods used to identify populations from well-described closely related species. Real nematological data is used to assess the potential and limitations of several methods. A method is introduced to handle large numbers of attributes. Both crisp and fuzzy representations of the data are investigated.

1. Introduction

In [1] a classification method is presented using local representation by fuzzy rules, where a data set is used for training and learning methods are provided to adjust the grades of certainty factors in the rules. Related results are found in [2] using multiple partitions but not learning. Both derive fuzzy rules using crisp data for the training set. Other methods using distance metrics may be found in [3,4].

In this paper we investigate the case where the data sets are treated as fuzzy values. The methods of [1,2] seem more natural to explore given the nature of our data. In particular, in descriptions of nematode species, it is most common to have a mean and range given for each measurement; less often a mean and standard deviation are given. Each mean and range is derived from a sample of ten to twenty specimens from a single population. This specimen data is usually not published due to volume of data. Consequently, for a well-described species, means and ranges for a dozen or so populations are published. For a classification problem this size training data set would be considered rather small. (Note that this would be called an identification problem in biology since the purpose is to use known classes, the training data, to develop rules that can then be used to assign unknowns, to one of the classes.)

For two closely related, well described species, distinguishing between unknowns can be very difficult and may ultimately rely on quantitative data, typically measurements. In [5], Fortuner studied two such species, Helicotylenchus dihystera, H.d., and Helicotylenchus pseudorobustus, H.p., using descriminant functional analysis, DFA. However, this method is more data intensive since it relies on data for individual specimen, which as mentioned is not generally available in the literature. Similarity methods have also been used, which are less data intensive as only means are required, [6], but they have the drawback of not assigning an unknown to a known class.

Our objective here is to investigate the feasibility of using the fuzzy rule-based methods of [1] for classification of well described species using the limited available data consisting of means and ranges. In doing this we present an algorithm that can extend the methods of [1], and [2], to identify unknowns in some cases when the number of attributes is large.

2. Fuzzy representation and rules.

We will use the data of from [5] for 11 populations of H.d. , denoted Da, ..., Dk, and 11 populations of H.p., denoted, Pa, ..., Pl for the training data. except for Pe, which has been reclassified as H.d. For the unknowns, we will use the 10 populations of H.d., denoted Dha, ..., Dhj, and 5 other populations H.flatus, H.nannus, H.microlobus, H.bradys, and H.phalerus. The first two are synonymous with H.d. and the last three with H. p.

We do not use artificially-generated data to test our methods. Therefore, the conclusions we draw are based on real data and this gives a good perspective on the possibilities and limitations of the methods.

Measurements for eight characters (attributes) are considered for each of the populations. They are: the length of the body, LON; the length of the oesophagus, OES; the length of the stylet, STY; the position of the phasmides, PHAS; the length of the process terminal, PRO; the length of the tail, QUE, the diameter of the body, DVU; and the number of tail annuli, TAN. The data for the first seven of these can be found in [5], pp. 194-222; that for PRO is unpublished. The domains for each character are shown in Table 1.

Index	Character	Domain
1.	LON	500 - 900
2.	OES	90 - 150
3.	STY	22 - 30
4.	PHAS	19 - 39
5.	PRO	0 - 5
6.	QUE	9 - 27
7.	DVU	17 - 38
8.	TAN	4 - 17

Table 1. Characters and domains

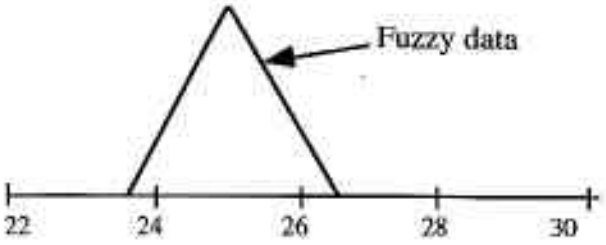
For a given population and each character the mean, $\bar{x}$, and the range will be represented by a triangular membership function, with the vertex at the mean, with membership value 1.0, and range for its support. An example is shown in Figure 1(a) where $\bar{x} = 24.88$, with range 23.5 - 26.5 for the character STY. Later we will briefly mention the effects of using trapezoidal functions to represent the fuzzy data.

As in [1], each domain will be partitioned into K-1 equal intervals to create K membership functions representing a fuzzy grid for the domain. Each boundary point of each interval will serve as the vertex of a triangular membership function with support between its neighboring boundary points, except for the membership functions at the ends. Clearly the scales are not the same on each axis, but as in [1], the size of the grid, i.e., K, for each axis will be the same. A member of the grid is denoted A^K_{ji} where j is the index of the character, which for our application will be the index shown in Table 1, and i is the i^{th} member of the grid. An example for K = 5 and the domain of the character LON is shown in Figure 1(b).

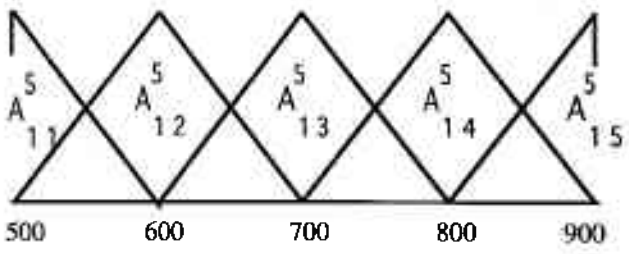
The membership function for A^K_{ji} will be designated μ^K_{ji}. If x is a crisp scalar then its value for the grid function A^K_{ji} is given by the membership function $\mu^K_{ji}(x)$. When x is a fuzzy data point, the value is

$$(1) \qquad a^K_{ji}(x) = \max_y \{\mu^K_{ji}((y), x(y)\}.$$

This is simply the greatest common value for x and A^K_{ji} [7]. For example, in Figure 2(a) the third grid function for the character LON, which has index 1, and K = 5, is A^5_{13} evaluated at the (crisp) scalar x = 675.5 to yield

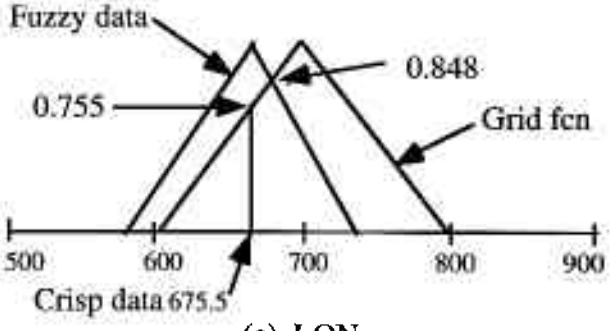


(a) Fuzzy data for STY

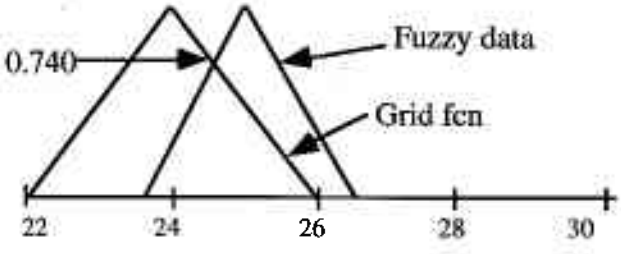


(b) A fuzzy grid for LON

Figure 1. Fuzzy functions.



(a) LON



(b) STY

Figure 2. Evaluating crisp and fuzzy data

$\mu^5_{1\,3}(675.5) = 0.755$. When x is a fuzzy scalar given by the mean, 675.5, and range, 584.0 - 737.0, shown also in Figure 2(a), $a^5_{1\,3}(x) = 0.848$.

When $x_p = (x_1, ..., x_n)$ is a multidimensional data point, where each component x_j is a fuzzy scalar representing a mean and range for character j, and $A^K_{j\,i_j}$ is the i^{th}_j grid function for the j^{th} character, j = 1, ..., n, we designate the product operator

$$(2) \qquad \prod_j a^K_{j\,i_j}(x_j)$$

where $x_j(y)$ is used in (1) in place of x(y). Instead of using all of the characters given in Table 1 will often use just a subset of these characters to form our rules. For those characters not selected, their factors will not appear in (2). For example, if we select just two characters, LON, index 1, and STY, index 3, then using the fuzzy data for LON and the grid function from Figure 2(a), represented by x_1 and $A^5_{1\,3}$, and if the data for STY is given by x_3 with mean 24.88 and range 23.50 - 26.50, and taking the grid function $A^5_{3\,2}$, shown in Figure 2(b), then the value of (2) will be 0.848 * 0.740 = 0.628. Continuing this example, a rule using just these two characters and the given grid functions in Figure 2 would have the form:

> If x_1 is $A^5_{1\,3}$ and x_3 is $A^5_{3\,2}$
>
> then x_p belongs to H.d. with CF = 0.463.

CF is the certainty factor of the rule, which indicates the strength of the conclusion that x_p is in the class H.d. (x_1 and x_3 are the component values of x_p.) The degree that the data point x_p satisfies this rule is the product of (2) and the CF, i.e., 0.628 * 0.463 = 0.291.

The creation of a rule for a subset of characters consists of taking a grid function from each domain for these characters and then determining the class, H.d. or H.p., and the CF. To do this, each data point in the training set that belongs to H.d. is selected to compute (2). These results are summed to form $\beta 1$. This is repeated for points in H.p. to produce $\beta 2$. If $\beta 1 > \beta 2$, then the class in the rule is set to be H.d.; if $\beta 2 > \beta 1$, the class is set to be H.p.; and if $\beta 1 = \beta 2$, then no class is specified by the rule. The CF for the rule is

$$(3) \qquad |\beta 1 - \beta 2| / |\beta 1 + \beta 2|.$$

In [1,2] once a rule-base is constructed, an unknown data point x_p can be identified in the following way: For each rule in the rule-base, compute (2) for x_p and multiply by the CF for the rule. Take the maximum value m_1 for rules where the class is H.d., and the maximum m_2 for rules where the class is H.p., and if the $m1 > m2$ the unknown is determined to belong to H.d., otherwise it is determined to belong to H.p., except in ties where the class is not determined. The confidence level is |m1 - m2|. Unless stated otherwise, when we construct a rule base the same subset of characters are used for each rule in the rule-base. When a subset of characters is used, the interpretation of class membership is relative to the characters used.

3. 3-way ordering of characters.

For eight characters, even a grid with K = 5 would be far too big since it would potentially generate 5^8 = 390,625 rules. As an alternative we start by ordering the characters by measuring their efficacy in identifying the training set after training is completed.

Using only one character at a time for the training set, without learning [1], and K large since only one character is involved, each character is assessed. The results are shown in Table 2(a). There the number of incorrectly identified training patterns is given, and the total confidence level is computed summing each correctly identified training point minus the confidence level for each improperly identified point. This is then repeated for two characters at a time, using the best two characters, PRO and STY, in combination with each of the next two best, QUE and LON, with the results shown in Table 2(b). Since two groups (PRO,QUE) and (PRO,STY) in Table 2(b) have the lowest score in terms of the number incorrectly identified training points, i.e., all are correctly identified, and since there is a common character, PRO, we take (PRO, STY, QUE) to be the best three-character combination. This combination could also be viewed as having incorrectly identified 3 since it can be obtained via the second and last entries in Table 2(b). One could therefore argue in favor of other heuristics such as the highest total score, which would give (PRO, STY, LON), or testing for the best triple among the best two.

This can be repeated with all remaining characters, to get the next best three-character set. For our set, the next best was (LON, DVU, PHAS), with the last set (OES, TAN). We will refer to this as 3-way ordering of the characters. If four characters remain at the end, they can be decomposed into two sets of two characters. Based on this heuristic the characters were given the descending order PRO, STY, QUE, LON, DVU, PHAS, TAN, and OES.

4. Fuzzy vs. crisp training data.

In addition to examining the feasibility of using fuzzy data to distinguish closely related species, we also examine whether there is any advantage in using fuzzy data instead of crisp data for the training sets. In what follows fuzzy data will always be used for testing unknowns as it is generally better and more reliable to test a sample of a dozen specimen from an unknown rather than a single specimen. When testing the training data we shall use fuzzy data for testing when fuzzy data is used for training and crisp for testing when crisp data is used for training. The assumption in the latter case being that only crisp data is available.

	# Incorrect	Total
LON	6	5.1
OES	10	1.1
STY	2	11.7
PHAS	8	2.0
PRO	2	13.1
QUE	4	4.8
DVU	7	2.4
TAN	9	0.3

(a) K = 20

PRO,LON	1	14.1
PRO,QUE	0	12.9
PRO,STY	0	16.5
STY,LON	3	11.7
STY,QUE	3	11.9

(b) K = 20

Table 2. Ranking the characters

Using just 23 crisp data points (means only for our 23 training populations Da - Pl) for each character it would appear to be too little data resulting in too few rules for the corresponding fuzzy subspaces. On the other hand, using fuzzy data might implicitly expand the data set resulting in fewer missing rules for the fuzzy subspaces.

Indeed, when the best five characters (1,3,5,6,7 in Table 1) with K = 5 were used the crisp data set produced 259 rules, which is one tenth the size of the rule base, 2,287 rules, using the fuzzy data. In both cases, rules with CF's less than 0.1 were eliminated and not counted. Somewhat unexpectedly the majority of rules in each case, 220 for the crisp training data, and 1,457 for the fuzzy training data, had CF's = 1.0. This is due to the fact that (3) will always produce the value 1.0 whenever one of the β's is zero, even if the other one is very small. Clearly then from the point of view of the rule base size, the crisp approach is preferred, and from the completeness of the rule base, the fuzzy approach would seem better. While we didn't test this here, one would expect that the forgetting algorithm of [1] could be used to reduce the number of rules.

The best five characters were also used for comparing crisp and fuzzy data for training, since using all eight characters presents problems with the rule-base size. Four rule bases each for the fuzzy data and the crisp data were created. One of the rule bases was created using the best two characters, PRO and STY, one using the best three characters, one the best four, and one the best five. The values of K were 20, 10, 5, and 5, respectively, with K decreasing as necessitated by the combinatorics. Table 3(a) shows the results of testing the training populations where the training and testing data are fuzzy. For example, for the first column the confidence level is shown for each population where the best 2 characters, PRO and STY were used to create the rule-base with K = 20.

Noteworthy is that the confidence levels for each population tends to decrease considerably as the number of characters increases from two to five. This is due in part to the coarser grid as K necessarily decreases, in part to the additional number of factors in computing (2), and in part to the descending order of the characters. The same holds true for crisp data as seen in Table 3(b).

While the fuzzy data has some of the highest confidence levels and the highest average, it also has the lowest confidence levels in comparison to crisp data. One population mis-identified. No training, [1], was used to compare fuzzy and crisp data. Using trapezoidal functions to represent the training data tends to further increase the confidence levels of the stronger populations and decrease those of the weaker ones.

5. The 3-way ordering algorithm.

In order to handle eight characters we present a method based on the descending order of characters using the best characters' confidence levels to offset those for characters where they are lower. In addition, grid sizes do not need to be reduced as the number of characters increases, as one could argue that a finer grid is also necessary for the more difficult characters.

As suggested by these considerations we propose the following algorithm as a means to boost the confidence levels of the training set and to handle larger sets of characters:

The 3-way ordering algorithm:

0. Set a value of K for the partition as in the adaptive method [1]. The value of K should be selected to create as fine a partition as desired without creating an excessively large rule base assuming that three characters will be used at a time.

#chars	2	3	4	5
K	20	15	10	5
Da	0.846	0.770	0.538	0.429
Db	0.832	0.704	0.433	0.378
Dc	0.802	0.596	0.694	0.621
Dd	0.188	0.103	-0.037	-0.050
De	0.507	0.330	0.140	0.107
Df	0.828	0.483	0.503	0.398
Dg	0.532	0.372	0.137	0.096
Dh	0.527	0.519	0.306	0.238
Di	0.607	0.394	0.255	0.195
Dj	0.842	0.615	0.379	0.385
Dk	0.784	0.522	0.266	0.186
Pa	0.840	0.813	0.433	0.332
Pb	0.751	0.658	0.346	0.229
Pc	0.814	0.648	0.325	0.233
Pd	0.622	0.389	0.233	0.250
Pe	0.364	0.110	0.010	0.088
Pf	0.764	0.506	0.369	0.377
Pg	0.773	0.599	0.571	0.593
Ph	0.866	0.528	0.292	0.220
Pi	0.802	0.703	0.480	0.480
Pj	0.872	0.602	0.535	0.375
Pk	0.822	0.714	0.355	0.283
Pl	0.867	0.767	0.499	0.437

Total (5 Chars): 6.88 Average: 0.299

(a) Fuzzy training

Da	0.498	0.468	0.357	0.203
Db	0.550	0.544	0.264	0.204
Dc	0.471	0.292	0.731	0.619
Dd	0.574	0.351	0.151	0.117
De	0.495	0.428	0.216	0.116
Df	0.800	0.308	0.406	0.230
Dg	0.495	0.385	0.176	0.100
Dh	0.476	0.605	0.181	0.127
Di	0.558	0.672	0.235	0.162
Dj	0.443	0.411	0.257	0.206
Dk	0.597	0.539	0.280	0.159
Pa	0.527	0.675	0.256	0.155
Pb	0.596	0.549	0.155	0.075
Pc	0.325	0.326	0.136	0.084
Pd	0.586	0.508	0.284	0.258
Pe	0.270	0.313	0.201	0.178
Pf	0.770	0.352	0.399	0.396
Pg	0.461	0.290	0.354	0.346
Ph	0.667	0.216	0.112	0.072
Pi	0.500	0.338	0.588	0.584
Pj	0.412	0.174	0.306	0.177
Pk	0.347	0.279	0.126	0.086
Pl	0.632	0.543	0.282	0.238

Total (5 Chars): 4.892 Average: 0.213

(b) Crisp training

Table 3. Testing knowns

1. Determine a descending order for the characters. This can be done determining the best characters, three at a time, as presented in Section 3. For each three characters create the resulting rule-base R_i. For each R_i compute the total confidence level C_i for the training set.

2. Test the training populations by computing for each population x_p its confidence c_i for each rule-base R_i. Compute the weighted average of the confidence levels, $\Sigma\ c_i*w_i$, where $w_i = C_i/\Sigma\ C_i$

3. If any of the training populations are mis-identified in step 2, then use the learning procedure as in [1] for each R_i where populations were mis-identified and recompute C_i. (An alternative is to train an R_i whenever any population is mis-identified by it even if none are mis-identified in Step 2.)

To clarify our method consider the characters we have presented. In step 0, we set K = 10, and determine PRO, STY, and QUE to be the first best set of characters and create the rule-base R_1. Repeat for LON, DVU, and PHAS to produce R_2, then for TAN and OES for to obtain R_3. The C_i were computed to be $C_1 = 12.443$, $C_2 = 5.650$, and $C_3 = 1.082$ for the fuzzy training data and $C_1 = 9.567$, $C_2 = 6.905$, and $C_3 = 3.239$ for the crisp.

In Step 2 all in the training sets are correctly identified, thus Step 3 is not used in this example. The alternative training will be considered when the unknowns are considered. The results for the fuzzy and crisp training sets are shown in Table 4.

For the fuzzy data and crisp data the average confidence level was higher using all eight characters with 3-way ordering over using the best five characters. Again, the fuzzy has a higher average confidence level than the crisp but is lower on the weaker populations than the crisp.

While the decoupling of the characters using 3-way ordering may not always achieve separation of the training sets, it can always be considered as an adjunct to the methods of [1,2], and be used whenever the training set is better with 3-way ordering than it is without it, and when the number of characters is large.

6. Testing the unknowns.

In testing the unknowns, represented as fuzzy data, the differences between using crisp or fuzzy training data leads to the same conclusions as it did with the training data, that the fuzzy has a higher average confidence level, but has lower confidence levels for the weaker populations. Both correctly identified 13 of the 15 unknowns, though a few only marginally that had very low confidence levels. Training the individual fuzzy-data-generated rule-bases R_i in the alternative in Step 3, did not alter these results very much. The mis-

identification of Dhf, was likely due to the small size of the training data set. However, for the population H.flatus, even when it was added to the training set was still mis-identified. In addition, when five characters were used that were best for H.flatus, its confidence level was very low, and this came at the expense of other

Da	0.669	0.435
Db	0.474	0.461
Dc	0.510	0.204
Dd	0.070	0.209
De	0.264	0.348
Df	0.418	0.228
Dg	0.293	0.306
Dh	0.495	0.488
Di	0.450	0.583
Dj	0.554	0.302
Dk	0.391	0.350
Pa	0.630	0.536
Pb	0.507	0.397
Pc	0.460	0.299
Pd	0.356	0.494
Pe	0.063	0.296
Pf	0.412	0.359
Pg	0.525	0.272
Ph	0.394	0.206
Pi	0.564	0.288
Pj	0.445	0.152
Pk	0.539	0.196
Pl	0.616	0.440
Total:	10.081	7.849
Ave.:	0.438	0.341

(a) Fuzzy (b) Crisp

Table 4. Testing knowns: 3-way ordering

populations that were then mis-identified. It should be pointed out that, H.flatus, in [5] appears to be an outlier.

Some additional tests were conducted. In one the 3-way grouping was taken to be (PRO, LON, TAN), the 1st, 4th, and 7th characters, (STY,DVU, OES), the 2nd, 5th, and 8th characters, , and (QUE, PHAS), the 3rd and 6th to see if the descending ordering had any real benefits. Indeed it did as in the latter case the average confidence levels were lower and three unknowns rather than two were mis-identified. In another test using the multipartition approach of [2], with K = 3 to 8, and 3-way ordering, there were no significant differences with the results shown above.

7. Conclusions.

We have examined fuzzy methods to create rule-bases using small data sets to identify biological populations that belong to one of two closely related species. Our results show that fuzzy methods can produce results that are promising, 13 of 15 unknowns correctly identified. Furthermore, crisp data may be sufficient for the training data as the rule-bases are considerably smaller, with only a small loss in average confidence levels, but having higher confidence levels for the weaker populations. The 3-way ordering method presented may help improve confidence levels and may be useful when the character set is large.

Dha	0.536	0.412
Dhb	0.399	0.244
Dhc	0.503	0.417
Dhd	0.384	0.216
Dhe	0.010	0.011
Dhf	-0.070	-0.009
Dhg	0.648	0.537
Dhh	0.678	0.492
Dhi	0.508	0.394
Dhj	0.564	0.518
H.microlobus	0.478	0.450
H.bradys	0.398	0.276
H.phalerus	0.279	0.240
H.flatus	-0.456	-0.176
H.nannus	0.039	0.107
Total:	4.898	4.129
Average:	0.327	0.275

(a) Fuzzy (b) Crisp

Table 5. Testing unknowns: 3-way ordering

References

[1] Nozaki, K., Ishibuchi, H., & Tanaka, H. 1996. *Adaptive Fuzzy Rule-Based Classification Systems.* IEEE Trans. Fuzzy Systems, 4 (3): 238-249.

[2] Ishibuchi, H., Nozaki, K., & Tanaka, H. 1992. *Distributed Representation of Fuzzy Rules and Its Application to Pattern Classification.* Fuzzy Sets Sys., 52:21-32.

[3] Yager, R. & Filev, D. 1994. *Essentials of Fuzzy Modeling and Control.* John Wiley & Sons, Inc., New York.

[4] Chiu, S., 1997. *Extracting Fuzzy Rules from Data for Function Approximation and Pattern Classification* in Fuzzy Information Engineering, eds., Dubois, Prade, & Yager. John Wiley & Sons, Inc., New York.

[5] Fortuner, R., 1987. *Variabilité et Identification des Espèces chez les Nématodes du Genre Helicotylenchus.* Éditions de l'ORSTOM, Paris.

[6] Fortuner, R., & Wong, Y., 1984. *Review of the Genus Helicotylenchus Steiner, 1945: I: A Computer Program for Identification of the Species.* Rev. Nematol., 7(4): 385-392.

[7] Klir, G. & Yuan, B., 1995. *Fuzzy Sets and Fuzzy Logic.* Prentice Hall, New Jersey.