A Study of the Robustness of KNN Classifiers Trained Using Soft Labels

Neamat El Gayar[1], Friedhelm Schwenker[2], and Günther Palm[2]

[1] Cairo University
Faculty of Computers and Information
12613 Giza, Egypt
n.elgayar@fci-cu.edu.eg
[2] University of Ulm
Department of Neural Information Processing
D-89069 Ulm, Germany
{friedhelm.schwenker, guenther.palm}@uni-ulm.de

Abstract. Supervised learning models most commonly use crisp labels for classifier training. Crisp labels fail to capture the data characteristics when overlapping classes exist. In this work we attempt to compare between learning using soft and hard labels to train K-nearest neighbor classifiers. We propose a new technique to generate soft labels based on fuzzy-clustering of the data and fuzzy relabelling of cluster prototypes. Experiments were conducted on five data sets to compare between classifiers that learn using different types of soft labels and classifiers that learn with crisp labels. Results reveal that learning with soft labels is more robust against label errors opposed to learning with crisp labels. The proposed technique to find soft labels from the data, was also found to lead to a more robust training in most data sets investigated.

1 Introduction

Dealing with vagueness is a common problem in many pattern recognition problems. This vagueness is sometimes due to the existence of overlapping classes. In supervised learning models (classifier design) crisp labels are mainly used for training. Crisp labels indicate the membership of a training pattern to a single class. Such labels can be hard to obtain in real applications and fail to reflect the natural grouping or uncertainty that is available among classes.

Few attempts have been made in the machine learning community to discuss the necessities, approaches and virtues of using soft labels for classifier training [1,2]. Soft labels allow a pattern to belong to multiple classes with different degrees. A soft label can be considered fuzzy, probabilistic or possibilistic according to what its entries indicate. A review of hard, fuzzy and probabilistic and possibilistic labels can be found in [3].

Using soft labels can be very useful in cases where the feature space has overlapping or ill-defined classes, to accommodate the uncertainty of an external teacher about certain patterns, to model the opinions of several experts, and to

F. Schwenker and S. Marinai (Eds.): ANNPR 2006, LNAI 4087, pp. 67–80, 2006.

deal with linguistic features [1]. In some real world applications like in medicine, a clear (crisp) classification of training data may be difficult or impossible: Assignments of a patient to a certain disorder frequently can be done only in a probabilistic (fuzzy) manner [4].

In most cases however, data sets are most commonly labeled with crisp labels. Nevertheless, soft labels can be generated to provide more realistic memberships of the training patterns to ensure robust training [5]. In [2] Kuncheva reviews various schemes for generating soft labels and discusses whether using soft labels for learning can improve the classifier performance. Results of a detailed experimental investigation for the *K-nearest neighbor classifier* (KNN) indicates that although there is no clear winner amongst the K-nearest neighbor classifier and the *Fuzzy-K-nearest neighbor classifier* (FKNN) using fuzzy labels; the FKNN can be a useful choice for some applications because it provides additional information about the certainty of the classification decision. Keller et al [6] claim that the improvement on the error rate might not be the main benefit from using the FKNN model. More importantly, the model offers a degree of certainty which can be used with a "refuse-to-decide" option. Thus objects with overlapping classes can be detected and processed separately.

Few work has been devoted to study learning under noisy test instances [7,8,9] and was mainly restricted to studying noise imposed on hard labels. In this study we attempt to compare between the robustness of classifiers trained using soft labels opposed to classifiers trained with hard labels against errors in the labels. In our problem we mainly are interested to study cases where the classes are not mutually exclusive and therefore each training sample is allowed several class labels. This is different for the multiple-label problem; where multiple candidate class labels are associated with each training instance and it is assumed that only one of the candidates is the correct label [10]. Our work also introduces a new labeling technique based on the *Generalized Nearest prototype Classifier* (GNPC) proposed in [11]. The GNPC has be shown to unify disparate classification techniques like clustering and relabeling, *Parzen's classifier*, *radial basis function networks* (RBF), *Learning vector quantization* (LVQ) type classifiers; and nearest neighbor rules. In this study we focus on one family of the GNPC to use it for generating soft labels which is based on clustering and relabeling. In particular our approach uses fuzzy clustering of data points and fuzzy relabeling of prototypes to assign soft labels to data vectors.

Experiments were conducted on five data sets to compare the classifier performances that learn using crisp labels and different types of soft labels. Experiments were conducted at different noise levels imposed on the data labels. The classifiers used in this study to learn using crisp/soft labels are KNNs. KNN models are simple, wide applicable models and are usually recognized as good competitors to many neural network models and other classification paradigms [11]. The KNN variations that learn using soft labels (i.e FKNN) have been investigated in detail [2] while, most neural model and other known classification techniques were mainly devised to work with crisp labels. Our study will be extended in future work to other classification paradigms working with soft labels

including MLP and RBF learning with fuzzy labels [5,1] and recently the work of LVQ models learning with soft labels [12].

The paper is organized as follows; Section 2 reviews two methods to generate soft labels based on a KNN classification. Section 3 proposes a new technique based on GNPC using fuzzy clustering and fuzzy relabelling. Section 4, describes the used data sets, and outlines details of the experiments conducted. In Section 5, results are illustrated, summarized and discussed. Finally, the paper is concluded in Section 6.

2 Soft Labels

A crisp label $y(x)$ of a pattern vector $x \in \mathbb{R}^d$ denotes the class to which this pattern belongs. On the other hand, if $l(x)$ is a soft label of x, then $l(x)$ is a M-dimensional vector with entries in $[0,1] \subset \mathbb{R}$ indicating the degree with which pattern x is a member of each class. Here M is the number of classes in the application at hand. The problem of determining a soft label can hence be stated as follows: Given a labeled data set Z of N samples $Z = \{x_1, \ldots, x_i, \ldots, x_N\} \subset \mathbb{R}^d$, where each x_i is associated with a crisp label $y(x_i) \in \{C_1, C_2, \ldots, C_M\}$, where C_i, $i = 1, \ldots, M$ are the available class labels. Calculate for each x_i a soft label, $l(x_i) \in (l_{i1}, l_{i2}, \ldots, l_{iM}) \in [0,1]^M$ representing the degrees of class memberships to the classes $\{C_1, C_2, \ldots, C_M\}$ respectively, i.e. $l_{ij} = l(x_i)_j$ denotes the degree with which pattern x_i belongs to class C_j. As follows, two schemes to assign soft labels using a KNN classifier are briefly described. In the next section we introduce a soft labeling technique based on Generalized nearest prototype classifier (GNPC).

2.1 K-Nearest Neighbor Soft Labels

The KNN soft labels for any $x_i \in Z$ are calculated as follows: First the k points in Z closest to x_i are determined, and then the membership of pattern x_i to class C_j is calculated through the relative frequencies:

$$l(x_i)_j = \frac{k_j}{k} \tag{1}$$

Here k_j is the number of elements $x \in Z$ amongst the k closest neighbors to x_i which are labeled with classes $y(x) = C_j$.

2.2 Keller Soft Labels

The Keller et al. soft labeling scheme [6] is similar to the KNN labeling scheme but guarantees that all objects retain their true class labels if the soft labels are "hardened" by the maximum membership rule. This scheme will affect only those objects which are close to classification boundaries by diminishing the "certainty" for their own class at the expense of increasing the certainty for the bordering class (or classes). The soft labels are computed as follows:

$$l(x_i)_j = \begin{cases} 0.51 + 0.49\frac{k_j}{k} & : \text{ if } C_j \text{ is the crisp class label of } x_i \\ 0.49\frac{k_j}{k} & : \text{ otherwise} \end{cases} \quad (2)$$

Again, k_j is the number of elements amongst the k closest neighbors $x \in Z$ to x_i which are labeled with class C_j.

3 Generating Soft Labels by GNPC

Prototype based classification is perhaps the simplest and most intuitively motivated pattern recognition paradigm. There are many classification techniques that are based implicitly or explicitly on similarity to point prototypes, for example RBF networks, LVQ and some recent extensions of the LVQ with soft assignments to data vectors to prototypes [12,13,14,15]. Like the K-nearest neighbor method Nearest Prototype Classifier (NPC) is a local classification method in the sense that classification boundaries are approximated locally. Instead of making use of all the data points of a training set, however, NPC relies on a set of appropriately chosen prototype vectors. This makes the method computationally more efficient, because the number of items which must be stored and to which a new data point must be compared for classification is considerably less. In [10] an integrated framework for a generalized nearest prototype classifier (GNPC) is proposed. Five large families of classifiers are shown to fit within the GNPC framework. The five families differ most importantly in the way prototypes are obtained and not in their formal GNPC representation. The definition of a GNPC is listed below.

Definition 1 [10]: The generalized Nearest Prototype Classifier (GNPC) is the 5-tuple (V, L_V, s, T, S) where:

a) $V = \{v_1, \ldots, v_p\}$, $v_i \in \mathbb{R}^d$ is the set of p prototypes;
b) L_V is the $M \times p$ label matrix of the p prototypes for M classes ;
c) $s(\cdot, \cdot)$ is a similarity measure defined on $\mathbb{R}^d$ that calculates the similarity between data point x and prototypes v_k.
d) T is any t-norm defined over fuzzy sets, and S is an aggregation operator.

Given an unlabeled vector $x \in \mathbb{R}^d$, the similarity of x to all p prototypes is calculated to produce the similarity vector $s = (s(x, v_1), \ldots, s(x, v_p)) = (s_1, \ldots, s_p)$ The label of a vector x to class C_j is assigned as follows

$$l(x)_j = S_{k=1}^{p} L_V(j, k) T s_k$$

According to the GNPC, x is then assigned to the class which corresponds to the maximum entry in the label vector. The GNPC framework can be categorized into 5 families of different classifiers according to the determination $V, L_V, s(x, v)$ and the operators T and S in the Definition above. It is shown that for the five families of the classifiers studied in [10] the type of operations used for T is the *product*; while the S is either defined as the *maximum* or the *average* operation.

As follows we introduce an approach to generate soft labels based on Generalized nearest prototype classifier (GNPC). In particular our proposed method is linked to the first family of classifiers which is based on clustering and re-labeling.

3.1 Soft Labels with GNPC Using Fuzzy Clustering and Fuzzy Re-labeling

Our proposed method to assign soft labels to a previously labeled data set is summarized in Table 1.

Table 1. The GNPC Algorithm

Input

- Z a set of N crisp labeled training examples x
- M classes $C_1, C_2, \ldots, C_M$.
- Each x is associated with a crisp label $y(x_i) \in \{C_1, C_2, \ldots, C_M\}$

1. Fuzzy Clustering FCM step

1. Initialize p, the number of clusters of Z
2. Cluster analysis of Z using the FCM algorithm
3. For each $x_i \in Z$, obtain its cluster membership $\mu(x_i)$

2. Calculating soft labels for the p clusters

1. Initialize the $M \times p$ matrix $L = 0$
2. For each point $x_i \in Z$ and it's label $y(x_i) = C_q$: $L_q = L_q + \mu(x_i)$ here L_q is the q-th row of L.
3. Normalize the columns of L to sum to 1
4. Normalize the rows of L to sum to 1

3. Infer a soft label of a pattern from the soft labels of the clusters

For data point x_i, calculate its fuzzy label as follows:

1. Find $\mu(x_i)$ from FCM in previous step
2. Calculate soft label using fuzzy composition: $l(x_i) = L \circ \mu(x_i)$.

As follows we describe the algorithm in details linking it to the GNPC described above. In Step 1 of the algorithm, the prototypes are defined in the data using clustering. Each cluster is represented by a single prototype. Data vectors in the same cluster are supposed to possess much more similarity to each other than to the patterns in other groups. Data vectors are clustered disregarding their labels. Here we choose to employs fuzzy C-means clustering (FCM) [16] which is probably the most popular fuzzy clustering method that attempts to cluster feature vectors by iteratively minimizing an objective function. The fuzzy *c-means* generates for each element x_i in the data a membership vector

$$\mu(x_i) = (\mu_{i1}, \ldots, \mu_{ip})$$

that describes how strong it belongs to each cluster. Here μ_{ik} denotes the similarity of pattern x_i to the cluster prototype v_k and is calculated using the FCM as follows:

$$\mu_{ik} = \frac{1}{\sum_{q=1}^{p} \left(\frac{d_{ik}}{d_{iq}}\right)^{\frac{1}{\beta-1}}}$$

$\beta > 1$ is the fuzzifier, d_{iq} can be calculated by any distance measure; here we use the Euclidean distance.

The membership values of the patterns generated using the FCM method are distributed over the clusters in a normalized fashion The number of clusters for a given dataset is computed using cluster validity measures [17]; where different numbers of clusters are experimented with and the clustering with the maximum cluster validity is selected [18].

Step 2, of the algorithm corresponds to finding soft labels for cluster prototypes denoted by the $M \times p$ prototype label matrix L_V. This is often referred to in the literature as relabelling. Crisp relabelling schemes minimizes overall number of resubstituition misclassifications to label each cluster prototype with the class held by the majority of the vectors within that cluster [10]. An advantage can be gained making soft relabeling [10,19]; thereby providing soft connections between prototypes and classes; value $L_V(j,k)$ can hence be regarded as the strength of the association of v_k with class C_j.

In step 2.2 of Table1 the fuzzy labeling procedure for the prototypes is described which uses the membership functions of the patterns generated from the FCM step and the crisp labels information to derive soft labels for the cluster prototypes. Here restrict the sum of label vector for the prototypes v_k to one and therefore we add the normalization step in 2.3-4. The resulting label matrix L_V can be regarded as a "fuzzy relation" on $C \times V$ [20] expressing the association between the classes and the prototypes.

Finally, in step 3 for a given data point its soft label can be inferred from the prototype label matrix L_V and the similarity of x to the given prototypes denoted by the FCM membership function vector $\mu(x)$ using a "composition" operation $\circ$ used for fuzzy-logic inference mechanism [20] The soft label of the vector x is hence calculated as follows: $l(x) = L \circ \mu(x)$ or $l(x)_j = S_{k=1}^{p} L_V(j,k) T \mu_k$. Typically T is an intersection operation (a t-norm) and S is either a union or a mean n-place operation. In our implementation we use T as product and the S as maximum operation.

4 Data and Experiments

Experiments were performed on five different data sets. All data sets share the characteristics that there exist some classes overlapping to some extent. The first data set is the Iris data set containing 4 features and 3 classes. We also used two benchmark synthetic data sets [2] which are two dimensional. The first syntactic data set, the Normal-mixtures data, consists of two classes and is generated from a mixture of two normal distributions with the same covariance matrices. The class distribution has been chosen to allow a best possible error rate of 8%. The second syntactic data set, the Cone-torus data set consist of 3 classes and is generated from three differently shaped distributions; where patterns from each class is not equal but are rather distributed with a frequency of 0.25, 0.25 and 0.5. We also use a set of object images, COIL data, obtained from Columbia Object Image Library [21]. The dataset contains 8 features of the images of 20 different objects, for each object 72 training samples are available.

Table 2 summarizes the characteristics of the data sets used. In addition, we use the Satimage data from the ELENA database [22]; which represents Landsat Multi-Spectral Scanner image data, consisting 6435 patterns each of 36 attributes representing 6 different classes (red soil, cotton crop, grey soil , damp grey soil, soil with vegetation stubble and very damp grey soil). Table 2 summarizes some of the characteristics of the data sets used.

Table 2. Summary of data sets used in the numerical experiments

Data set	No of features	No of classes	No of data points	pattern/class
Iris	4	3	150	equal
Normal-mixtures	2	2	1000	equal
Cone-torus	2	3	400	different
COIL	8	20	1440	equal
Satimage	36	6	1000	different

We mainly use the K-nearest neighbor classifier (KNN) in the numerical experiments. The KNN classifier is popular for its simplicity to use and implementation, robustness to noisy data and its wide applicability in a lot of appealing applications [23]. We used a simple version of the FKNN [2] that is trained using soft labels. The KNN trained with crisp labels was compared to the FKNN trained with the different soft labels described in Section 2. Experiments were repeated for the five data sets described above. To evaluate the robustness of the different labels we forced errors on the training sets with different percentages and examined the performance of the classifiers trained with crisp labels and the three variations of the soft labels. The errors were introduced to the crisp training data set and then mapped by the corresponding labeling scheme into soft labels. The accuracy of the compared techniques was calculated using a 10-fold cross validation [24]. Note the K-nearest neighbor value, k used in equations 1 and 2 to generate the KNN soft labels and the Keller soft labels, respectively are not necessary the same as the K-nearest neighbor value, used for the KNN and the FKNN classifiers used for the final classification; we will therefore refer to the latter value as K to avoid confusion. We investigated the affect of the choice of k (for soft label generation) and K (for final classifiers) in our experimentation. In particular we generate the soft labels using KNN soft labels and the Keller soft labels for $k = 3, 5, 7, 9, 11, 13, 15$. For the KNN and FKNN classifiers we repeat the experiments for $K = 3, 5, 7, 9$ for all data sets. In the following section experimental results are presented and discussed.

5 Results and Discussion

Figures 1, 2, 3, 4 and 5 illustrate the results on the Isis data set, the Normal mixtures data set, the Cone Torus data set, the Coil data set and the Satimage data set, respectively. The figures compare the accuracy of the KNN classifier trained with crisp labels and the accuracy of the FKNN classifier trained with the

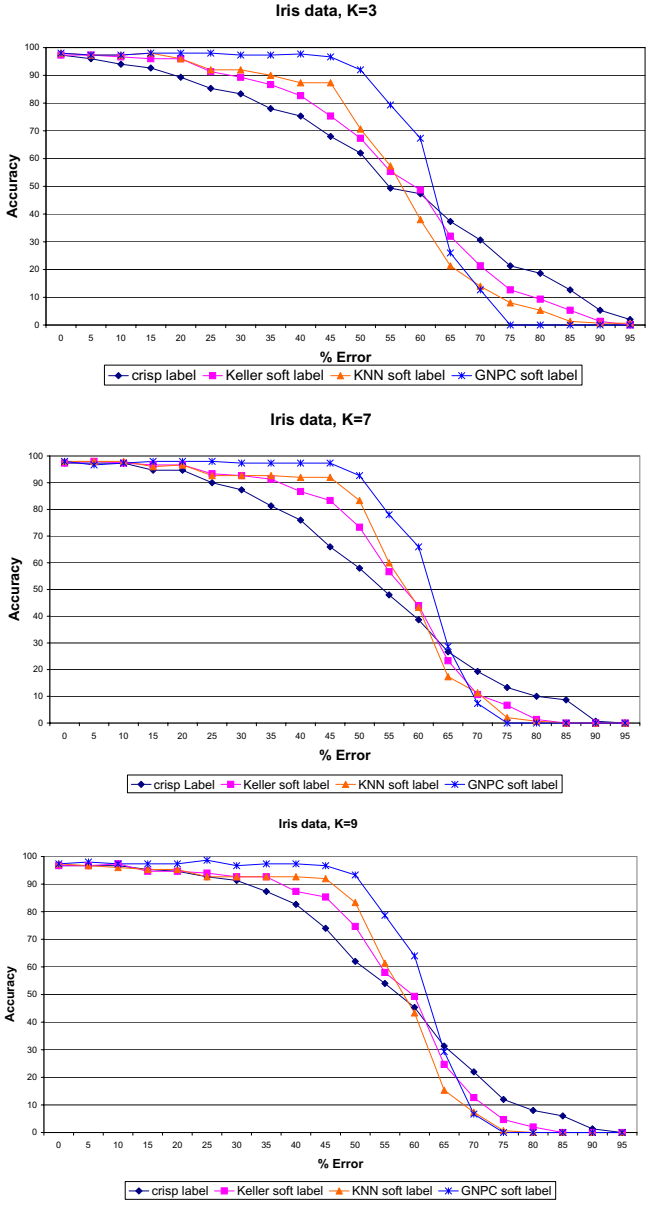

Fig. 1. Results of the IRIS data

KNN soft labels, Keller et al. soft labels and the proposed GNPC soft labels. The accuracies for the different techniques are calculated when noise was introduced to the class labels. In the experiments, from 5-95% of the class labels have been flipped at random using the uniform distribution. Results for KNN and FKNN classifier at $K = 3$ are presented for all data sets as the results on other values

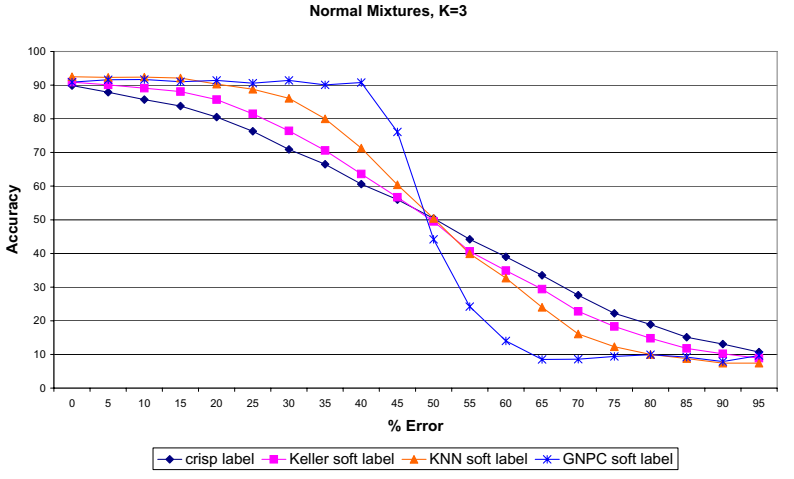

Fig. 2. Results of the Mixtures data set

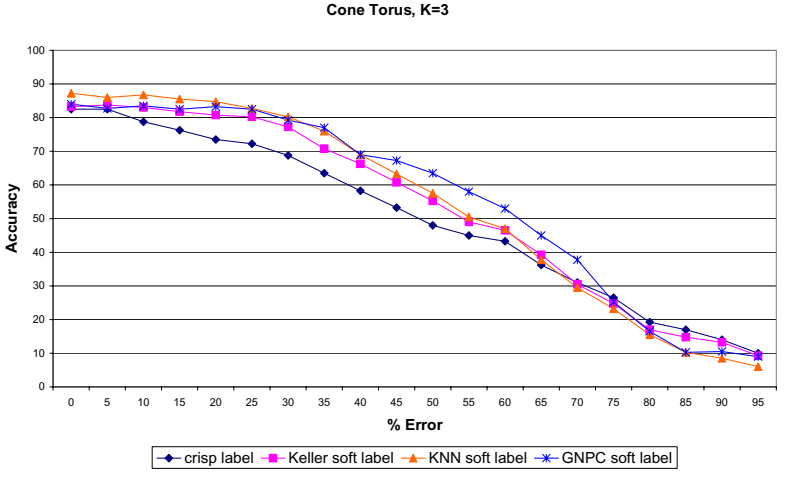

Fig. 3. Results of the Cone-torus data set

of K resulted in a more or less similar relative behavior for all techniques when different labeling techniques are compared. Figure 1 outlines the results for the Iris data set for $K = 3, 7, 9$. The performance of the classifiers trained with Keller soft labels and KNN soft labels showed sensitivity for the choice of k ; where the results of the classifiers trained with KNN soft labels were generally more sensitive to the choice of k. Different data sets behaved differently under the choice of k (and the choice of the K for the classifiers) but generally experiments have shown that a reasonable choice of k would be 7 or 9 for all data and classifier models under investigation. In figures 1-5 we fix k to 7. Figures 6 and 7

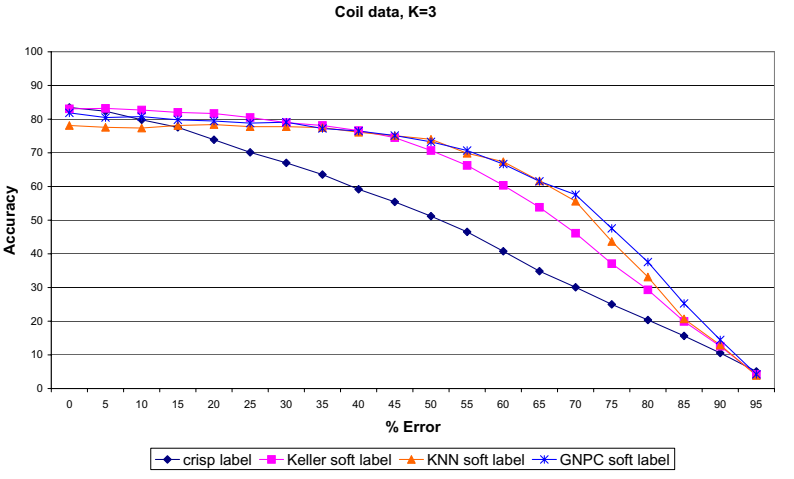

Fig. 4. Results of the COIL data set

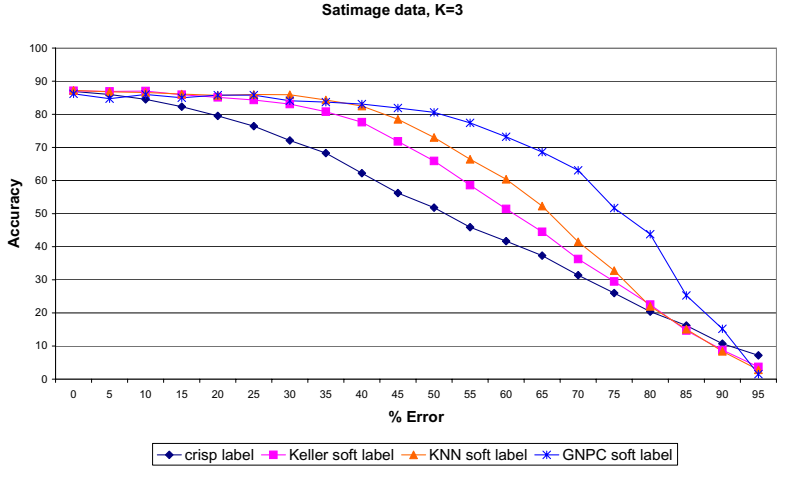

Fig. 5. Results of the Satimage data set

investigate the performance of the classifiers trained with Keller soft labels and KNN soft labels with $k = 3, 5, 7, 9, 11, 13, 15$ for the Iris and the Satimage data sets, respectively. For the GNPC soft labels we used an adaptive technique to find the optimal number of clusters "p" for each data set as mentioned before. For the Iris data set we used 10 clusters, the Normal Mixtures data with 20 clusters, the Cone torus data with 45 clusters, the Coil data with 300 clusters and the Satimage data with 120 clusters.

The performance the classifiers trained with the FCM soft labels were affected by the number of clusters "p" chosen and there is a relation between the number

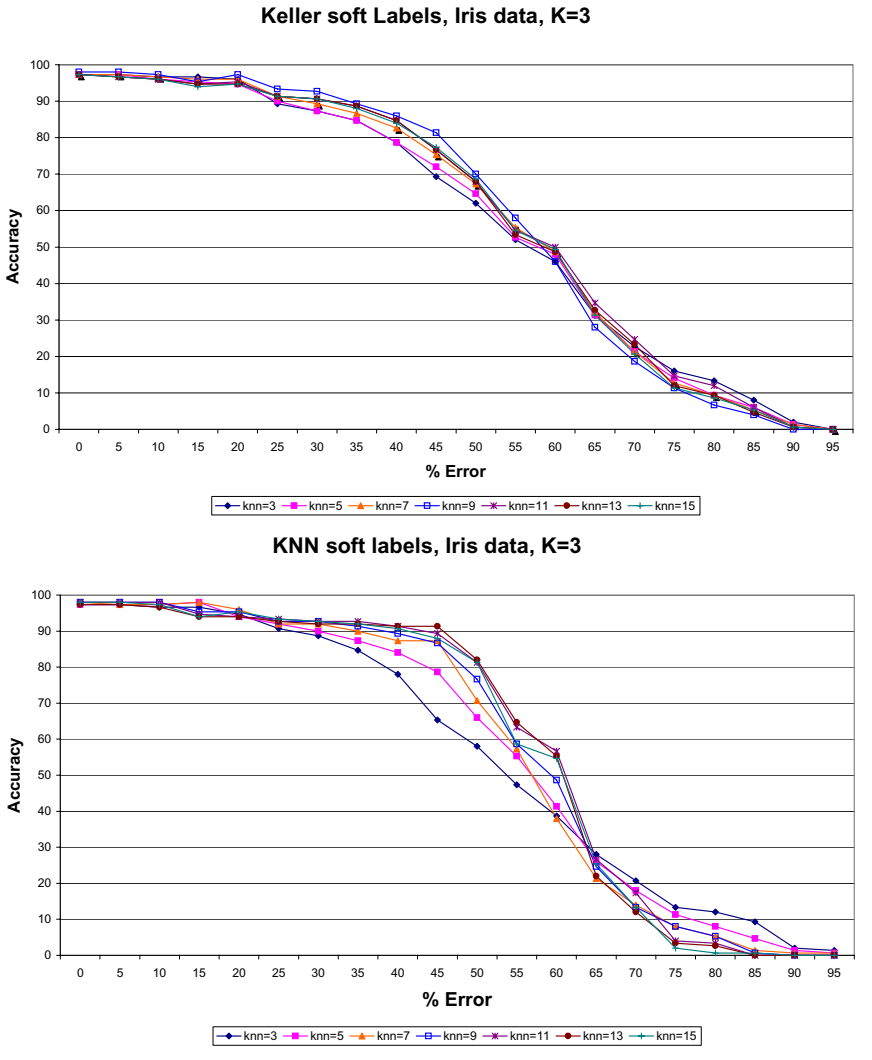

Fig. 6. Effect of the parameter k for the Iris data when using Keller soft labels and KNN soft labels

of classes available in the data and the number of adequate clusters. We recommend starting with number of clusters that are a multiple of 10 of the number of classes. For the GNPC soft labels we also used for the FCM algorithm a fuzzification constant of 2, except for the Coil and the Satimage data sets where we reduced the fuzzification constant to 1.1 to prevent soft labels produced by the FCM clustering to be to diffuse among multiple classes. To infer the soft labels of a given data point we used a max-product [20] composition operator.

Examining the results illustrated in figures 1 through 5, it is obvious that the accuracy of the KNN classifier trained with crisp labels is generally less

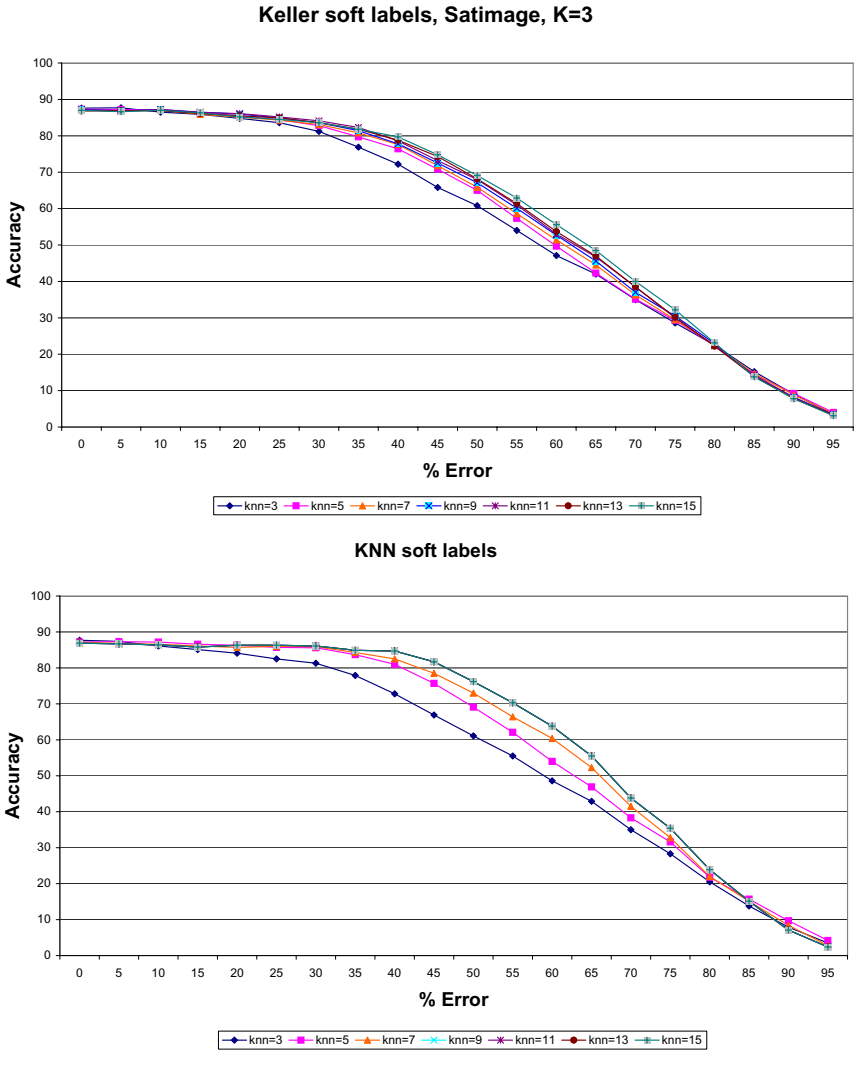

Fig. 7. Effect of the parameter k for the Satimage data when using Keller soft labels and KNN soft labels

robust to errors on the labels compared to the FKNN classifier trained with different soft labels. The FKNN trained with soft labels seems to be able to sustain a robust performance up to 65% error rate for the Iris data, 45 % error rate for the Normal Mixtures data, 75% error for the Cone torus data and finally until 95% for both the Coil data and the Satimage data; compared to the KNN trained with crisp labels. In the same time, the FKNN classifier trained with the GNPC soft label is able in most cases to maintain a more stable accuracy compared to the other KNN and the Keller soft label for most data sets under investigation.

6 Conclusion and Future Work

In this work we propose a new technique to obtain soft labels from available crisp labels that is based on fuzzy clustering of the data and fuzzy relabeling of the cluster prototypes. Fuzzy labels are obtained through fuzzy logic based inference. The proposed method is extended from a GNPC family which unifies many disparate classification techniques.

The robustness of the classifiers trained with crisp and soft labels is tested. Experiments were conducted on five data sets to compare the behavior of the classifiers when errors are available in the data labels. Results reveal the robustness of the KNN classifier trained with soft labels opposed to classifiers trained with crisp labels. The proposed soft labeling scheme based of fuzzy clustering and fuzzy relabelling was found in general be most robust to error on labels and less sensitive to the choice of parameters (like the k parameter in the KNN and Keller soft labeling techniques) and the KNN classifier model. In our future work we intend to examine more alternatives to generate soft labels by exploring and extending other families of the GNPC framework and to study the effectiveness of other classifier models (MLP, RBF, LVQ and SVM) that learn with soft labels.

We particularly also intend to investigate other alternatives for soft labels in cases where the data set is incompletely labeled or is labeled using different sources of information. We also aim towards using soft labels in the framework of multiple classifier systems and similarly test their usefulness in the context of accuracy and robustness.

Acknowledgement

This research has been partially supported by the DFG (German Research Society) grant SCHW 623/3-2 and SCHW 623/4-2.

References

1. El Gayar, N.F.: Fuzzy Neural Network Models for Unsupervised and Confidence-Based Learning. PhD thesis, University of Alexandria, Dept. of Comp. Sc. (1999)
2. Kuncheva, L.: Fuzzy classifier design. Physica-Verlag (2000)
3. Bezdek, J., Pal, N.: Two soft relatives of learning vector quantization. Neural Networks **8**(5) (1995) 729–743
4. Villmann, T., Hammer, B., Schleif, F.M., Geweniger, T.: Fuzzy labeled neural gas for fuzzy classification. In: Proceedings of WSOM 2005. (2005) 283–290
5. Pal, S., Mitra, S.: Multilayer perceptron, fuzzy sets and classification. IEEE Transactions on Neural Networks **3** (1992) 683–697
6. Keller, J., Gray, M., Givens, J.: A fuzzy k-nearest algorithm. IEEE Transactions on Pattern Analysis and Machine Intelligence **7** (1985) 693–699
7. Angluin, D., Laird, D.: Learning from noisy examples. Machine Learning **2**(4) (1987) 343–370
8. Lam, P., Stork, D.: Evaluating classifiers by means of test data with noisy labels. In: Proceedings of the International Joint Conference on Artificial Intelligence. (2003) 513–518

9. Bernadt, J., Soh, L.: Authoritative citation knn learning with noisy training datasets. In: Proceedings of the International Conference on Artificial Intelligence. (2004) 916–931
10. Jin, R., Ghahramani, Z.: Learning with multiple labels. In: Advances in Neural Information Processing Systems. Volume 15. (2003)
11. Kuncheva, L., Bezdek, J.: An integrated frame work for generalized nearest prototype classifier design. International Journal of Uncertainty, Fuzziness and Knowledge-based System **6**(5) (1998) 437–457
12. Villmann, T., Schleif, F.M., Hammer, B.: Fuzzy labeled soft nearest neighbor classification with relevance learning. In: Proceedings of ICMLA 2005, IEEE Press (2005) 11–15
13. Karayiannis, N.: Repairs to GLVQ: A new family of competitive learning schemes. IEEE Transaction on Neural Networks **7** (1996) 1062–1071
14. Seo, S., Bode, M. Obermayer, K.: Soft nearest prototype classification. IEEE Transaction on Neural Networks **14** (2003) 390398
15. Seo, S., Obermayer, K.: Soft learning vector quantization. **15** (2003) 15891604
16. Melin, P., Castillo, O.: Hybrid Intelligent Systems for Pattern Recognition Using Soft Computing. Springer-Verlag (2005)
17. Xie, X., Beni, G.: A validity measure for fuzzy clustering. IEEE Trans. on Pattern Analysis and Machine Intelligence **13**(8) (1991) 841–847
18. Czogala, E., Leski, J.: Fuzzy And Neuro-Fuzzy Intelligent Systems. Physica-Verlag (2000)
19. Van de Wouwer, P., Scheunders, P., van Dyck, M., De Bodt, M., Wuyts, F., Van de Heyning, P.: Wavelet-filvq classifier for speech analysis. In: Proc. of the Int. Conf. Pattern Recognition, IEEE Press (1996) 214218
20. Klir, G., Yuan, B.: Fuzzy Sets and Fuzzy Logic, Theory and Applications. Prentice-Hall of India (2002)
21. Nene, S.A., Nayar, S.K., Murase, H.: Columbia object image library (coil-20). Technical report, Technical Report CUCS-005-96 (1996)
22. ELENA: (ftp.dice.ucl.ac.be, directory: pub/neural-nets/elena/databases)
23. Duda, R., Hart, P., Stork, D.: Pattern Classification. John Wiley and Sons (2004)
24. Kuncheva, L.: Combining pattern classifiers. John Wiley and Sons (2004)