Hubness-Based Fuzzy Measures for High-Dimensional k-Nearest Neighbor Classification

Nenad Tomašev[1], Miloš Radovanović[2], Dunja Mladenić[1], and Mirjana Ivanović[2]

[1] Institute Jožef Stefan
Artificial Intelligence Laboratory
Jamova 39, 1000 Ljubljana, Slovenia
nenad.tomasev@ijs.si, dunja.mladenic@ijs.si
[2] University of Novi Sad
Department of Mathematics and Informatics
Trg D. Obradovića 4, 21000 Novi Sad, Serbia
radacha@dmi.uns.ac.rs, mira@dmi.uns.ac.rs

Abstract. High-dimensional data are by their very nature often difficult to handle by conventional machine-learning algorithms, which is usually characterized as an aspect of the *curse of dimensionality*. However, it was shown that some of the arising high-dimensional phenomena can be exploited to increase algorithm accuracy. One such phenomenon is *hubness*, which refers to the emergence of hubs in high-dimensional spaces, where hubs are influential points included in many k-neighbor sets of other points in the data. This phenomenon was previously used to devise a crisp weighted voting scheme for the k-nearest neighbor classifier. In this paper we go a step further by embracing the *soft* approach, and propose several fuzzy measures for k-nearest neighbor classification, all based on hubness, which express fuzziness of elements appearing in k-neighborhoods of other points. Experimental evaluation on real data from the UCI repository and the image domain suggests that the fuzzy approach provides a useful measure of confidence in the predicted labels, resulting in improvement over the crisp weighted method, as well the standard kNN classifier.

1 Introduction

High dimensional data are ubiquitous in modern applications. They arise naturally when dealing with text, images, audio, data streams, medical records, etc. The impact of this high dimensionality is manyfold. It is a well known fact that many machine-learning algorithms are plagued by what is usually termed the *curse of dimensionality*. This comprises a set of properties which tend to become more pronounced as the dimensionality of the data increases. First and foremost is – the unavoidable sparsity of data. In higher-dimensional spaces all data is sparse, meaning that there is not enough data to make reliable density estimates. Another mitigating influence comes from the concentration of distances, which has been thoroughly explored in the past [1, 2]. Namely, all the distances between data points generated from the same distribution tend to become increasingly more similar to one another as new dimensions are added. Luckily, this does not affect multiple-distribution data as much, as was shown in [3]. The question of

whether the very concept of nearest neighbors is meaningful in high-dimensional data sets was addressed in [4]. Admittedly, there are some difficulties, but nearest neighbor methods remain popular, both for classification and clustering.

Hubness is a high-dimensional phenomenon which concerns k-nearest-neighbor sets [5–7]. Denote by $N_k(x)$ the number of k-occurrences of x, i.e., the number of times x appears in k-nearest-neighbor lists of other points in the data. The distribution of $N_k(x)$ exhibits significant skew in high-dimensional cases, skew which increases with *intrinsic* dimensionality of the data. This leads to the emergence of *hubs*, influential points which affect the reasoning procedure of nearest-neighbor-based methods for many data points. Hubs appear as a consequence of the geometry of high-dimensional space, and the behavior of data distributions within them. Most data sets (approximately) appear as hyperspheres or unions of hyperspheres centered around some distribution means. This positioning renders points closer to the data centers more likely to be included in k-nearest-neighbor lists of other data points. This tendency increases with dimensionality. The hubness phenomenon was successfully applied to the k-nearest neighbor (kNN) algorithm, yielding observable improvement in many cases [5, 8]. This weighting scheme will be addressed in Section 2.1 as we start to explain the motivation for our subsequent approach.

Our goal is to extend the class-nonspecific crisp kNN weighting scheme described in [5] to class-specific soft voting in the spirit of the fuzzy k-nearest neighbor (FNN) algorithm [9]. The rest of the paper is structured as follows. In Section 2 we present the related work, focused around two major points – the hubness-weighted kNN algorithm, and the FNN algorithm. While observing the former, we outline its weak points and aim our proposed improvements in their direction. The respective hubness-based fuzzy membership functions are presented in Section 3. We go on to evaluate the proposed approach in Section 4. Finally, we give our final remarks as well as future research directions in Section 5.

2 Related Work

2.1 Hubness-Weighted *k*NN

Weighted voting in nearest-neighbor classifiers has become something of a common practice. Weights are usually either based on element position in the k-neighbor list or its distance to the observed data point. Some more robust approaches taking into account also the correlation between these differences have also been recently developed [10]. The hubness weighting scheme which was first proposed in [6] is a bit more flexible, in a way that the weight associated to x_i is $w(x_i, k)$, meaning that each point in the training set has a unique associated weight, with which it votes whenever it appears in some k-neighbor list, regardless of its position in the list. This weighting is based on the interpretation of how the hubness phenomenon affects kNN performance. As was mentioned before, hubness of an element x_i is the number of its k-occurrences in neighbor lists of other elements, and is denoted by $N_k(x_i)$. This can be decomposed into two parts: $N_k(x_i) = GN_k(x_i) + BN_k(x_i)$, where $GN_k(x_i)$ is the number of *good* k-occurrences and $BN_k(x_i)$ is the number of *bad* k-occurrences. Good occurrences are those when the label of x_i matches the label of the element in whose k-neighbor list

x_i is observed. Bad occurrences are characterized by a mismatch of labels. Elements with high bad hubness are often found in neighbor lists of elements belonging to other categories in the data. This means that bad hubs exhibit a detrimental influence on k-nearest-neighbor classification, because their vote often gives misleading information. The aforementioned weighting scheme reduces these bad influences directly. Standardized bad hubness is defined as $h_b(x_i, k) = (BN_k(x_i) - \mu_{BN_k})/\sigma_{BN_k}$, and the weight associated to x_i is then $w(x_i, k) = e^{-h_b(x_i,k)}$. It was shown that this often leads to significant improvement in high-dimensional settings where hubs naturally appear as an artefact of dimensionality. The amount of improvement depends on the distribution of bad hubness within the data.

What the described approach disregards completely is the structure of bad hubness. In non-binary classification, when a label mismatch occurs, it can occur for any of the classes. Instead of observing $N_k(x_i)$ as a sum of good and bad hubness, we could decompose it into $N_k(x_i) = \sum_{c=1}^{n_c} N_{k,c}(x_i)$, where each $N_{k,c}(x_i)$ is the number of k-occurrences of x_i in neighborhoods of elements of class c, and n_c is the total number of classes. Good hubness is just the special case when $c = y_i$, y_i being the label of x_i in the data set. Therefore, instead of using the hubness information only to reduce the votes of bad hubs, it is possible to take into account the structure of bad hubness, which can be used to decompose the crisp vote given by x_i into a fuzzy vote relying on all $N_{k,c}(x_i)$. There already exists a framework that can assist in achieving this goal, referred to as the fuzzy nearest neighbor classifier.

2.2 Fuzzy Nearest Neighbor Algorithm

Fuzzy sets are based on a notion of inherent ambiguity in the data, meaning that a single element can be viewed as partially belonging to several different categories at the same time [11]. This ambiguity is often problem-specific and the set membership function is then provided by the domain experts. However, there are also ways of deducing some sort of fuzziness automatically from the data. Denote by $u_{ci} = u_c(x_i)$ the degree of membership of x_i in class c. The following properties must hold in order for u_c to define a fuzzy split on the data set:

$$\begin{aligned} &\sum_{c=1}^{n_c} u_{ci} = 1, \\ &0 < \sum_{i=1}^{n} u_{ci} < n, \\ &u_{ci} \in [0, 1]. \end{aligned} \tag{1}$$

If a fuzzy measure u_c is given, it is possible to perform k-nearest neighbor classification in a fuzzy manner, as was first proposed by [9]. Let x be a newly observed data instance for which we wish to perform classification. The degree of membership of x in each of the classes is then defined as

$$u_c(x) = \frac{\sum_{i=1}^{k} u_{ci}(\|x - x_i\|^{-(2/(m-1))})}{\sum_{i=1}^{k} (\|x - x_i\|^{-(2/(m-1))})}, \tag{2}$$

where $\|\cdot\|$ denotes the Euclidean norm. The parameter m in Eq. 2 determines how heavily the distance is weighted when calculating contributions from each neighbor. For large values of m, neighbors are weighted more equally, while low values of m favor closer neighbors. The most commonly used default value for this parameter is $m = 2$, so that fuzzy votes are weighted by the reciprocal of the distance.

There exist many ways for automatically generating suitable fuzzy measures from the data. This is not only used for class membership fuzziness, but also for fuzzifying attributes. A range of techniques can be used, including genetic algorithms, clustering, neural networks, entropy, and others [12]. In the original fuzzy-nearest-neighbor article [9], some simple ways to achieve this were also proposed, one of which was to observe k nearest neighbors of x_i and count the percentages of them coming from any particular class. The final measure was a linear combination of the element's label and these percentages, normalized so as to fall in the desired $[0, 1]$ range.

Apart from applying the fuzzy approach to specific domains, most attention has been given lately to the issues of scalability in terms of achieving speedup in fuzzy nearest neighbor search [13, 14], as well as improving the weighting scheme [15].

3 Proposed Hubness-Based Fuzzy Measures

The basis of our motivation was already mentioned in Section 2.1 while discussing the properties of hubness-weighted kNN. Instead of using *good* and *bad* hubness, we propose to use *class hubness* $N_{k,c}(x_i)$ defined uniquely for each element in the training set. It is immediately apparent that this measure can be fit into the fuzzy nearest-neighbor framework. Contrary to the more usual fuzzy measures, it does not represent inherent fuzziness of an element's label, but instead measures the fuzziness of an *appearance* of elements in k-neighbor sets, based on the training data. Regardless of the semantic difference between the two, their form remains the same. There are, however, some difficulties with using hubness as a fuzzy measure. For small values of k, there are many elements in the data which have zero hubness. This becomes even more pronounced in high dimensions due to the mentioned skew of the distribution of k-occurrences. Also, in non-binary classification problems, we need even more hubness data in order to be able to properly estimate the partial memberships for all the existing categories. This poses a serious limit on using class hubness for calculating fuzziness. We would be forced to use very high k values, which could be detrimental in cases when best kNN classification is achieved for smaller neighborhood sizes, as is often the case for non-noisy small or medium-sized data sets.

We propose to handle the problems outlined above by only using hubness of the elements which exhibit hubness greater than some predefined threshold. This in fact separates the data for which it is possible to make reliable fuzzy estimates from those which exhibit too low a hubness to be of any use in such a way. For the data below the threshold, we propose to use a different fuzzy estimate. We explore four such approaches and discuss the pros and cons of their use in the rest of this section, as well as analyzing the fruitfulness of their application in Section 4 when presenting the results of experimental evaluation. Let X be the training set and Y the set of corresponding

labels. The hybrid fuzzy measure which we will be considering in the rest of the paper takes the following form:

$$u_c(x_i) = \begin{cases} p_k(y = c|x_i) \approx \frac{N_{k,c}(x_i)+\lambda}{N_k(x_i)+n_c\lambda}, & \text{if } N_k(x) > \theta, \\ f_k(c, x_i), & \text{if } N_k(x) < \theta. \end{cases}$$

The term $p_k(y = c|x_i)$ denotes the conditional probability of element x being of class c if element x_i appears in its k-neighbor set. For elements which exhibit hubness above a certain threshold, this can be estimated by dividing the class hubness by total hubness. The λ factor is a Laplace estimator, which is used for smoothing to prevent any probability from being estimated as zero. By observing the formula for the conditional probability, one can notice that the label y_i of x_i is not used at all when casting the vote of x_i! This is indeed a very peculiar property. Even though it is possible to work with fuzziness defined in such a way, we wanted to make the fuzziness also dependent on the element's label, so we included each x_i in its own neighbor list at the 0th position. For high hubness elements, this does not make a large difference, but by doing so we implicitly express a certain degree of confidence in label y_i. The value of $f_k(c, x_i)$ for low-hubness elements should, ideally, represent a kind of estimate of the actual conditional probability. Since this is not easy to achieve, alternative nearest neighbor-based fuzzy estimates pose themselves as viable alternatives. We focused on four different ways of dealing with low hubness: a crisp estimate method, a global estimate method, as well as two different local estimates.

- What we refer to as the *crisp estimate* (CE) is the simplest and least flexible way of handling low hubness, which is not in itself necessarily bad – to use the element's own label. In this scenario, low-hubness elements vote the same way they would vote in kNN, with no attached fuzziness. Smoothing is performed by using the same λ value as before.
- *Global estimate* (GE) is more flexible, but introduces the risk of adding more fuzziness than necessary. We compute the GE of the conditional probability as defined in Eq. 3. The denominator is in fact what $\sum_{(x,y)\in(X,Y)|y=y_i} \sum_{c=1}^{n_c} N_{k,c}(x)$ sums up to. This is a sensible approach, but it remains questionable just how much is lost and how much is gained by employing it. Even though it does give a global conditional probability of elements from a particular class being included in neighbor sets of another class, there is no guarantee that locally, in the observed part of the data set, this estimate holds.

$$f_k(c, x_i) = \frac{\lambda + \sum_{(x,y)\in(X,Y)|y=y_i} N_{k,c}(x)}{n_c\lambda + \sum_{(x,y)\in(X,Y)|y=y_i} N_k(x)} \tag{3}$$

- If the global estimate fails to capture the proportions contained in the underlying conditional probability for a specific data instance, using a local fuzziness estimate is a possible alternative. Since we already have the k-neighbor lists, it seems natural to take advantage of this when trying to estimate an element's fuzziness. Here we depart from trying to estimate the actual conditional probability and experiment with a more usual approach. Let $\{x_{i1} \dots x_{ik}\}$ be the k nearest neighbors of x_i and for convenience denote x_i also as x_{i0}, since we insert each element into its neighbor

list at the 0th position. The *local estimate* (LE_1) is then given by Eq. 4, where $\delta_{cy_{ij}}$ is Kronecker's delta function. It is not entirely clear which value of k would lead to a good estimate, therefore in our experiments we used $k = 10$ by default.

$$f_k(c, x_i) = \frac{\lambda + \sum_{j=0}^{k} \delta_{cy_{ij}}}{n_c\lambda + k + 1} \tag{4}$$

– There is an alternative way to define local fuzziness based on nearest neighbors and this was in fact one of the methods from the original FNN paper [9]. It is based on LE_1, but made so as to emphasize the label of an element, as in the CE method. In fact, it represents a linear combination of the two approaches. We will denote it LE_2, as defined in the following equation:

$$f_k(c, x_i) = \begin{cases} 0.51 + 0.49 \cdot \frac{\lambda + \sum_{j=1}^{k} \delta_{cy_{ij}}}{n_c\lambda + k + 1}, & \text{if } c = y_i, \\ 0.49 \cdot \frac{\lambda + \sum_{j=1}^{k} \delta_{cy_{ij}}}{n_c\lambda + k + 1}, & \text{if } c \neq y_i. \end{cases}$$

Apart from testing these fuzzy measures separately, we have also merged them into a single hybrid hubness-based fuzzy nearest neighbor algorithm which we present in Algorithm 1. Given the training data set, we use the leave-one-out procedure to try classifying each point x from the training data by observing the remaining $n - 1$ elements. Such a classification is attempted for each element and for all the k values in a given range, as well as different threshold values and different $f_k(c, x_i)$. The configuration leading to the highest accuracy on the training data is then selected for use on the test set.

Time complexity of this approach is in fact completely comparable to the one of hubness-weighted kNN, with the bottleneck being the computation of k-neighbor sets. Fast approximate algorithms for calculating all k-neighbor sets do exist, with one of the most recent presented by Chen et al. [16]. This approximate algorithm runs in $\Theta(dn^{(1+\alpha)})$ time, where $\alpha \in (0, 1]$ is a parameter used to set a trade-off between speed and accuracy. This makes hubness-based algorithms potentially feasible for use on large-scale data sets.

We tested two versions of the algorithm presented in Algorithm 1. The first version uses the distance-based fuzzy vote weighting described in Eq. 2, which we denote by *dwh-FNN*. As an alternative we also tested a version of the algorithm where no distance-based weighting is performed, and fuzzy voting was performed simply by summing all the respective u_{ci} for every class. This will be referred to as *h-FNN* in the rest of the text. The parameter m from Eq. 2 was set to 2 by default, this being the value which is most frequently used.

4 Experimental Evaluation

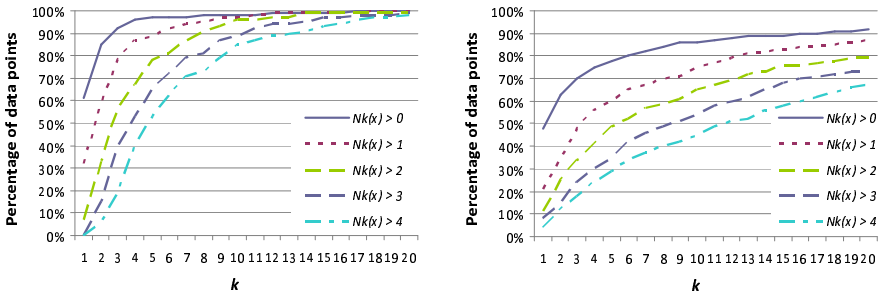
This section presents the results of experiments that compare the standard k-nearest neighbor classifier and hubness-weighted kNN, with the two proposed hubness-based fuzzy approaches h-FNN and dwh-FNN. Section 4.1 employs data sets from the established UCI repository of various dimensionalities, while Section 4.2 focuses on high-dimensional data from the image domain.

Algorithm 1 Hubness-based Fuzzy Nearest Neighbor: Training

```
int[][] nearestNeighbors = calculateNearestNeighborLists(k_min, k_max);
float[][][] classHubnessAllK = calculateElementToClassHubness(nearestNeighbors);
float[][][] GEAllK = calculateGlobalEstimates(nearestNeighbors);
float[][] LE_1 = calculateLE1(nearestNeighbors);
float[][] LE_2 = calculateLE2(nearestNeighbors);
float[][] CE = calculateCE(nearestNeighbors);
float maxAcc = 0;
int bestK, bestTheta;
for all θ = θ_min; θ ≤ θ_max; θ++ do
  for all k = k_min; k ≤ k_max; k++ do
    float GEAcc, LE1Acc, LE2Acc, CEAcc = 0;
    for all i = 0; i < trainingData.length; i++ do
      if votebyGE(x_i, GEAllK, ClassHubnessAllK, nearestNeighbors) == x_i.label then
        GEAcc++;
      end if
      if votebyLE1(x_i, LE_1, ClassHubnessAllK, nearestNeighbors) == x_i.label then
        LE1Acc++;
      end if
      if votebyLE2(x_i, LE_2, ClassHubnessAllK, nearestNeighbors) == x_i.label then
        LE2Acc++;
      end if
      if votebyCE(x_i, CE, ClassHubnessAllK, nearestNeighbors) == x_i.label then
        CEAcc++;
      end if
    end for
    updateMaxAccAndBestConfiguration(GEAcc, LE1Acc, LE2Acc, CEAcc);
  end for
end for
```

4.1 UCI Data Sets

Hubness-based fuzzy measures that we proposed are of a hybrid nature, since in each case they combine two different estimates. In order to see how different estimates might be applied, we calculated on each data set, for a range of neighborhood sizes, the percentage of data points which have hubness below/above a given threshold. For two of the used data sets, the plots for several lower thresholds for hubness can be seen in Fig. 1. Naturally, great variation of behavior can be observed across different data sets, since it is related to the aforementioned skew of the hubness distribution in high dimensions. In other words, we expect for highly skewed data sets the term $f_k(c, x_i)$ to play a more important role than in the case of low to medium-skewed data with respect to hubness. It is precisely for these data sets that the mentioned estimates of actual hubness may become as important as hubness itself. From Fig. 1, however, the difference becomes quite clear. For less skewed data sets, if good classification can be achieved for neighborhood size $k \in [10, 20]$ or above, then there is probably enough hubness information to allow for its use as a fuzzy measure. If, on the other hand, the nature of the data is such that best results are obtained for low k values, ranging maybe from 1



(a) Iris data set (b) Dexter data set

Fig. 1. Percentage of elements with hubness over a certain threshold, for neighborhood sizes 1–20

to 5, the situation is reversed. However, when dealing with highly skewed data, such as in case of the Dexter data set, influence of $f_k(c, x_i)$ is non-negligible even when considering higher k values.

The first round of testing was performed on 15 data sets taken from the UCI data repository. The used data sets are of various sizes and dimensionalities, and are summarized in Table 1, with the first six columns denoting data-set name, size, dimensionality (d), number of classes (n_c), and the observed skewness of the distributions of N_1 and N_{10} (S_{N_1}, $S_{N_{10}}$).[3] For each data set, the skew of the distribution of k-occurrences was calculated for various k values, to indicate the degree of hubness of the data. Euclidean distance was used in all the experiments.

On the described UCI data sets, kNN, hubness-weighted kNN, h-FNN and dwh-FNN were tested. In all the algorithm tests, 10 runs of 10-fold cross-validation were performed. All algorithm parameters were set automatically, separately on each fold during the training phase, based on the training set. Neighborhood sizes were tested in the range $k \in [1, 20]$ and thresholds $\theta \in [0, 10]$. Classification accuracies achieved by the classifiers are given in Table 2. The corrected resampled t-test [17] was used to test for statistical significance of differences in accuracy for each data set. Differences which were found to be significant with $p < 0.01$ compared to dwh-FNN are denoted by symbols ○/● in the table.

The dwh-FNN classifier was selected as the baseline for statistical comparison in Table 2 since we determined that it generally outperformed all other classifiers. To provide a more detailed pairwise classifier comparison, Table 3 shows the number of wins of classifiers signified by the column label, over classifiers denoted by the row labels, with statistically significant wins given in parenthesis.

Overall improvement over kNN is apparent already from the shown average scores over all data sets in Table 2, as well as Table 3. Particular improvements vary and there do exist data sets for which none can be observed, as well as some where performance degradation is present. Hubness-weighted kNN, h-FNN and dwh-FNN exhibit similar

[3] Skewness, the standardized 3rd moment of a distribution, is 0 if the distribution is symmetrical, while positive (negative) values indicate skew to the right (left).

Table 1. Summary of UCI datasets

Data set	size	d	n_c	S_{N_1}	$S_{N_{10}}$
colonTumor	62	2000	2	1.04	1.06
dexter	300	20000	2	2.95	3.33
diabetes	768	8	2	0.73	0.15
ecoli	336	7	8	0.62	0.37
glass	214	9	6	0.58	0.23
ionosphere	351	34	2	2.17	1.71
iris	150	4	3	0.46	0.03
isolet-1	1560	617	26	1.30	1.20
mfeat-fourrier	2000	76	10	1.20	0.75
ozone-eighthr	2534	72	2	1.31	0.70
page-blocks	5473	10	5	0.79	0.11
parkinsons	195	22	2	0.39	−0.19
segment	2310	19	7	0.70	0.16
vehicle	846	18	4	0.92	0.44
yeast	1484	8	10	0.78	0.27

Table 2. Classification accuracy of kNN, hubness-weighted kNN (hw-kNN), h-FNN and dwh-FNN on UCI data sets. The symbols ○/● denote statistically significant better/worse performance compared to dwh-FNN

Data set	kNN	hw-kNN	h-FNN	dwh-FNN
colonTumor	65.1±19.6 ●	72.5±20.6	74.9±20.0	74.5±20.0
dexter	60.1±18.2 ●	72.5± 7.9 ○	68.6± 8.3	68.5± 8.3
diabetes	76.5± 4.1 ○	72.0± 4.6 ●	74.2± 4.9	74.2± 4.9
ecoli	85.4± 6.0	84.5± 6.4	83.6± 6.4	84.3± 6.3
glass	70.5± 9.3 ○	67.6±10.0 ○	65.4± 9.9 ○	63.8±10.0
ionosphere	89.7± 5.2	87.5± 5.7 ●	89.9± 5.5	90.0± 5.6
iris	96.9± 4.0 ○	95.3± 4.8	95.1± 4.7	94.7± 4.8
isolet-1	90.0± 2.6 ○	81.3± 3.4 ●	81.2± 3.8 ●	82.3± 3.6
mfeat-fourier	77.5± 2.9 ●	80.3± 2.6 ●	81.0± 2.6 ●	81.9± 2.6
ozone-eighthr	76.8± 2.5 ●	93.4± 1.8	93.4± 1.3	93.6± 1.3
page-blocks	93.5± 1.0 ●	96.0± 0.8	96.1± 0.8	96.2± 0.8
parkinsons	82.7± 7.7 ●	92.1± 5.8	92.5± 5.2	92.7± 5.2
segment	89.9± 1.7 ●	91.2± 1.7	90.8± 1.8 ●	91.2± 1.8
vehicle	60.7± 5.7 ●	66.6± 5.1	64.4± 4.9	65.2± 5.6
yeast	59.0± 4.1 ○	52.3± 4.1 ●	55.1± 3.8	55.5± 3.8
Average	78.29	80.34	80.41	80.57

Table 3. Pairwise comparison of classifiers on UCI data: number of wins (with statistically significant ones in parenthesis)

	kNN	hw-kNN	h-FNN	dwh-FNN
kNN	–	8 (8)	9 (8)	9 (8)
hw-kNN	7 (6)	–	9 (4)	10 (5)
h-FNN	6 (6)	6 (2)	–	11 (3)
dwh-FNN	6 (5)	5 (2)	4 (1)	–

improvement patterns, which makes sense given that they aim at exploiting the same phenomenon. Improvement over the standard kNN classifier signifies that there is a lot of usable bad-hubness information in the data. Fuzzy approaches appear to offer additional improvement over hw-kNN, justifying our approach and the need to differentiate between classes when employing bad hubness for nearest-neighbor classification. The cases where standard kNN is significantly better than hubness-based approaches most probably stem from the difficulties of estimating $p_k(y = c|x_i)$, which requires more data in the case of non-binary classification, as well as $f_k(c, x_i)$ occasionally being an inappropriate substitute in cases of low hubness.

It appears that distance-based weighting described in Eq. 2 does not bring drastic overall improvement to hubness-based fuzzy membership functions that are used in the FNN algorithm, at least not for the default value of the m parameter. This is not all that surprising, though. As was stated in previous discussion, the semantics of hubness-based fuzziness differs slightly from that of more usual fuzzy measures. This is due to the fact that class hubness marks the fuzziness of the elementary event that point x_i appears in a k-neighbor set of an element of some specific category. This hubness is estimated by previous appearances of that element in k-neighbor sets of various other elements in the training data. Among these occurrences, x_i may be located at either place within each observed k-neighbor set. In other words, hubness is a measure which is for a fixed k independent of which positions in k-neighbor sets an element takes. If these lists were to undergo a random permutation, the hubness for that fixed neighborhood size would have remained unchanged.

Let us assume that we wish to determine the label of a new example x by using h-FNN. The contribution of those x_i closer to x stems not only from previous events when they were also close to the observed element, but also from previous events when they were much further away. The same holds for farther elements in the k-neighbor set. This is why a linear combination of class hubness contributions is sufficient and any additional distance-based weighting seems superfluous. On the other hand, due to the fact that we can not calculate proper class hubness probabilities for low hubness elements, this is only partially true. In those cases when fuzziness is estimated for low hubness x_i, distance-based weighting might bring some improvement, by emphasizing more important votes. In practice, most k-neighbor sets will probably contain a mixture of these cases.

Comparisons between different hubness-based fuzzy membership functions proposed in Section 3 were also performed. Experiments were rerun without automatic

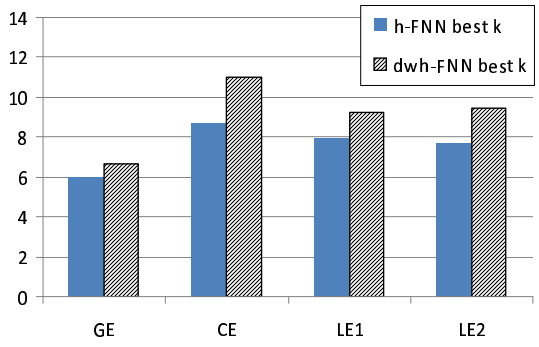


Fig. 2. Average best k values for different hubness-based fuzzy approaches, according to the results from tests on UCI data

parameter selection on the folds, so that the algorithms were trained once for every combination of $k \in [1, 20]$ and $\theta \in [0, 4]$, for every proposed fuzzy scheme. We extracted the parameter values from the range where the algorithms achieved highest accuracy scores, based again on the 10 times 10-fold cross-validation procedure, for every data set. Averages of k values for which the best results were obtained are shown for every used fuzzy scheme in Fig. 2. For each fuzzy approach, lower k values were selected on average if no distance-based vote weighting was performed. This suggests that is distance weighting is performed, more neighbors are required to convey the same amount of information, due to some votes being downgraded. Also, global hubness-based fuzziness (GE) finds its maximum at lower k-values than other measures. This suggests that it might indeed be the most appropriate among the observed approaches, since at lower k-values all $f_k(c, x_i)$ exhibit greater influence on classification, as was discussed previously. However, average best accuracies for all the approaches were basically the same. This means that hubness itself is still the most important part of the hybrid fuzziness. By following the same logic, we conclude that CE is indeed the least flexible way to replace unknown low class hubness probabilities, being the crisp approach. There seems to be no significant difference between the two local fuzzy measures. Average θ value for which best accuracy was achieved was around 1.5 for all approaches. This means that more often than not, class hubness was to be preferred to any of the $f_k(c, x_i)$ terms, even when based only on 3 or 4 k-occurrences.

Frequencies of selected neighborhood size falling in one of the four ranges: $[1, 5]$, $[6, 10]$, $[11, 15]$, $[16, 20]$, are shown in Fig. 3. Two ranges are preferred more often, namely $k \in [1, 5]$ and $k \in [11, 15]$. By examining all the results, we found that in cases of the more tangible accuracy improvements, larger k values ($k > 10$) were selected, while lower k values usually signify equal or only slightly better performance. This can be seen as natural, since larger k values provide the algorithm with more hubness information and hence better probability estimates, on which the used fuzziness was based. However, not all data sets are such that high k values make sense, since in some it may induce a larger breach of locality. This is why hubness-based approaches are not

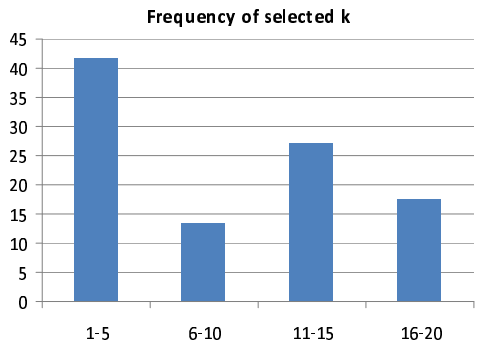


Fig. 3. Frequency of selected best k values, based on the results from tests on UCI data

expected to lead to an improvement over all data sets. This is their inherent limitation. Of course, this also depends heavily on the size of a particular data set. With more data, higher k values can be observed more safely. In high-dimensional spaces this is also affected by the curse of dimensionality because the data is always sparse.

4.2 ImageNet Data

ImageNet database is a large online repository (http://www.image-net.org/) containing over 12 million images grouped in more than 17000 synsets (classes). Images are inherently high-dimensional data, and are therefore quite suitable for testing hubness-based approaches. Out of synsets from the ImageNet hierarchy we constructed five image data sets for testing, with the used classes summarized in Table 4. Some of them combine completely different images, as *subs-3*, while some are made more difficult by containing several different plant types in different categories, as in *subs-6*. SIFT features and color histograms were extracted for each image [18]. A codebook of 400 most representative SIFT features was obtained by clustering from a large sample. Each image was thus represented by a 400-dimensional array of codebook frequencies, as well as a 16-dimensional color histogram. We used the Manhattan distance on this group of data sets. No feature weighting was performed, meaning that color and texture information was given equal significance. This may not be optimal, but we were not interested in performing optimal image classification, since our goal was only to compare the approaches under consideration on high-dimensional data. As in the previous section, Table 5 gives an overview of the obtained data sets. Note that this data exhibits a much higher skew of the distribution of k-occurrences than most UCI data sets from Table 1.

On each of the subsamples we performed 10 times 10-fold cross-validation. The value of k was chosen automatically from the range $k \in [1..10]$ on each fold. Average accuracies of the classifiers are given in Table 6. Statistically significant differences ($p < 0.05$) compared to dwh-FNN are denoted by symbols ∘/•. Pairwise classifier comparison is shown in Table 7.

Table 4. Class structure of the used ImageNet data subsamples

Data set	Classes
subs-3	sea moss, fire, industrial plant
subs-4	cloud, butterfly orchid, herbaceous plant, bird
subs-5	bird, fire, tracked vehicle, people, compass flower
subs-6	fish, industrial plant, wind turbine, compass flower, butterfly orchid, evergreen plant
subs-7	football, worm, sea star, night club, cloud, orchidaceous plant, mountain range

Table 5. Summary of ImageNet data sets

Data set	size	d	n_c	S_{N_1}	$S_{N_{10}}$
subs-3	2731	416	3	15.85	6.19
subs-4	6054	416	4	8.87	6.32
subs-5	6555	416	5	26.08	11.88
subs-6	6010	416	6	13.19	6.23
subs-7	8524	416	7	5.62	4.60

Table 6. Classification accuracy of kNN, hubness-weighted kNN (hw-kNN), h-FNN and dwh-FNN on ImageNet data sets for $k \in [1..10]$. The symbol • denotes statistically significant worse performance compared to dwh-FNN

Data set	kNN	hw-kNN	h-FNN	dwh-FNN
subs-3	78.29±2.38 •	81.51±3.34	82.16±2.26	82.34±2.23
subs-4	54.68±2.02 •	65.91±1.82	64.83±1.62	64.87±1.61
subs-5	50.80±2.08 •	58.06±3.80 •	61.54±1.93	61.81±1.95
subs-6	63.09±1.81 •	70.10±1.68	68.84±1.58	69.04±1.64
subs-7	46.71±1.63 •	51.99±4.68 •	58.85±1.60	59.04±1.59
Average	54.71	65.51	67.24	67.42

Table 7. Pairwise comparison of classifiers on ImageNet data: number of wins (with statistically significant ones in parenthesis)

	kNN	hw-kNN	h-FNN	dwh-FNN
kNN	–	5 (5)	5 (5)	5 (5)
hw-kNN	0 (0)	–	3 (2)	3 (2)
h-FNN	0 (0)	2 (0)	–	5 (0)
dwh-FNN	0 (0)	2 (0)	0 (0)	–

Hubness-based algorithms show obvious improvement on all subsets over the standard kNN classifier. As the number of categories increases, improvement of h-FNN and dwh-FNN over hubness-weighted kNN becomes more pronounced, which is consistent with observations on UCI data.

5 Conclusions and Future Work

We have proposed several ways of incorporating hubness into fuzzy membership functions for data elements. This was meant as a generalization of the previous hubness-weighted kNN approach. Fuzzy nearest-neighbor classification offers better confidence measures of the proposed labels, which leads to potentially easier interpretability of the results by experts working on the problem – and this is the reason we decided to extend the previous crisp hubness-based approach into a fuzzy counterpart. Several hybrid fuzzy membership functions were tested and evaluated. Global class-to-class neighborhood probabilities appear to be the most reliable way to deal with low-hubness elements. The fuzzy nearest-neighbor classifier employing these fuzzy measures outperforms the basic kNN classifier and also offers improvement over the hubness-weighted kNN. The accuracy improvement thus achieved may not be large on average, but the main advantage of the fuzzy approach lies in the mentioned interpretability of the results, and the fact that the approach takes advantage of high intrinsic dimensionality of data instead of being hampered by it, taking a step closer to mitigating the curse of dimensionality.

There is still room for improvement of the proposed methods, with several issues which we plan to address in our future work. Alternative local fuzzy estimates for low-hubness elements need to be explored, since there is no clear reason why the global estimate should lead to better performance as it currently does. Also, an option of using a linear combination of hubness and these estimates for low-hubness elements appears promising, since that way the little hubness information that these elements have would not be simply discarded, but rather extended by this additional information.

Acknowledgments

This work was supported by the bilateral project between Slovenia and Serbia "Correlating Images and Words: Enhancing Image Analysis Through Machine Learning and Semantic Technologies," the Slovenian Research Agency, the Serbian Ministry of Education and Science through project no. OI174023, "Intelligent techniques and their integration into wide-spectrum decision support," and the ICT Programme of the EC under PASCAL2 (ICT-NoE-216886) and PlanetData (ICT-NoE-257641).

References

1. François, D., Wertz, V., Verleysen, M.: The concentration of fractional distances. IEEE Transactions on Knowledge and Data Engineering **19**(7) (2007) 873–886

2. Aggarwal, C.C., Hinneburg, A., Keim, D.A.: On the surprising behavior of distance metrics in high dimensional spaces. In: Proc. 8th Int. Conf. on Database Theory (ICDT). Volume 1973 of Lecture Notes in Computer Science., Springer (2001) 420–434
3. Houle, M.E., Kriegel, H.P., Kröger, P., Schubert, E., Zimek, A.: Can shared-neighbor distances defeat the curse of dimensionality? In: Proc. 22nd Int. Conf. on Scientific and Statistical Database Management (SSDBM). Volume 6187 of Lecture Notes in Computer Science., Springer (2010) 482–500
4. Durrant, R.J., Kabán, A.: When is 'nearest neighbour' meaningful: A converse theorem and implications. Journal of Complexity **25**(4) (2009) 385–397
5. Radovanović, M., Nanopoulos, A., Ivanović, M.: Hubs in space: Popular nearest neighbors in high-dimensional data. Journal of Machine Learning Research **11** (2010) 2487–2531
6. Radovanović, M., Nanopoulos, A., Ivanović, M.: Nearest neighbors in high-dimensional data: The emergence and influence of hubs. In: Proc. 26th Int. Conf. on Machine Learning (ICML). (2009) 865–872
7. Radovanović, M., Nanopoulos, A., Ivanović, M.: On the existence of obstinate results in vector space models. In: Proc. 33rd Annual Int. ACM SIGIR Conf. on Research and Development in Information Retrieval. (2010) 186–193
8. Radovanović, M., Nanopoulos, A., Ivanović, M.: Time-series classification in many intrinsic dimensions. In: Proc. 10th SIAM Int. Conf. on Data Mining (SDM). (2010) 677–688
9. Keller, J.E., Gray, M.R., Givens, J.A.: A fuzzy k-nearest neighbor algorithm. IEEE Transactions on Systems, Man and Cybernetics **15**(4) (1985) 580–585
10. Zuo, W., Zhang, D., Wang, K.: On kernel difference-weighted k-nearest neighbor classification. Pattern Analysis and Applications **11** (2008) 247–257
11. Zadeh, L.A.: Fuzzy sets. Information and Control **8**(3) (1965) 338–353
12. Cintra, M.E., Camargo, H.A., Monard, M.C.: A study on techniques for the automatic generation of membership functions for pattern recognition. In: Congresso da Academia Trinacional de Ciências (C3N). Volume 1. (2008) 1–10
13. Zheng, K., Fung, P.C., Zhou, X.: K-nearest neighbor search for fuzzy objects. In: Proc. 36th ACM SIGMOD Int. Conf. on Management of Data. (2010) 699–710
14. Babu, V.S., Viswanath, P.: Rough-fuzzy weighted k-nearest leader classifier for large data sets. Pattern Recognition **42**(9) (2009) 1719–1731
15. Pham, T.D.: An optimally weighted fuzzy k-NN algorithm. In Singh, S., Singh, M., Apte, C., Perner, P., eds.: Pattern Recognition and Data Mining. Volume 3686 of Lecture Notes in Computer Science. Springer (2005) 239–247
16. Chen, J., Fang, H., Saad, Y.: Fast approximate kNN graph construction for high dimensional data via recursive Lanczos bisection. Journal of Machine Learning Research **10** (2009) 1989–2012
17. Nadeau, C., Bengio, Y.: Inference for the generalization error. Machine Learning **52**(3) (2003) 239–281
18. Zhang, Z., Zhang, R.: Multimedia Data Mining. 1st edn. Chapman and Hall (2009)