Feature Selection for Retrieval Purposes

Marco Reisert[1] and Hans Burkhardt[1]

University of Freiburg, Computer Science Department,
79110 Freiburg i.Br., Germany
{reisert,burkhardt}@informatik.uni-freiburg.de

Abstract. The quality of a retrieval system relies to major part on the quality of the used features. The features have to be small and compact, but also discriminative. Feature selection is one way to achieve both goals. We present a new feature selection method with the focus on retrieval purposes. The new method is based on the well known Relief algorithm. The new algorithm is shown to be superior to state-of-the-art methods both on toy problems and real-life 3D-Shape and image retrieval tasks. The algorithm is based on the intuition that distances to false detection has to be enlarged and distances to non-detected positives has to be shortened.

1. Introduction

Feature Extraction and Feature Selection are important tasks in many fields of computer vision. Given a retrieval or classification task, the goal is to find a transform from a possible high dimensional feature space to a low dimensional space, which is better suited for retrieval or learning purposes. While in Feature Extraction the transform may be nearly arbitrarily chosen, in Feature Selection one restricts to simply selecting a subset of features. In literature two types of Feature Selection algorithms are distinguished, the so-called wrapper and filter methods [1]. Wrapper methods estimate the usefulness of a subset of features by a given predictor or learning machine. Filter or Variable Ranking methods compute relevance scores for each single feature and choose the most relevant ones according to those scores. A popular example is Pearson's correlation coefficient expressing the correlation of a certain feature with the corresponding class labels. The main drawback of such simple filter methods is that they are not able to detect inter-feature-dependencies, one important example is the XOR-problem. Neither the first nor the second dimension alone helps to determine from which class an example is stemming, only both dimensions together contain enough information about the class membership. But there are methods, which may be categorized as variable ranking methods and are also able to reveal such feature dependencies, e.g. Relief [3] is one of those.

In this paper we present some fundamental modifications of the basic Relief algorithm, which lead to nice performance improvements and show that also

filter methods are able to cope with non-linear and multi-modal[1] environments. The paper is organized as follows: in Section 2 we uncover some drawbacks of the original Relief algorithm and propose the modifications. In Section 3 we examine various experiments on toy and real-world data, including image and shape retrieval examples. And finally in Section 4 we give a conclusion and outlook.

2. The Method

In this section we first give a introduction to Relief and then propose the modifications. Finally we make some short complexity considerations.

2.1. Relief

Our proposed idea has similarity with a popular feature ranking algorithm called Relief proposed in 1992 by Kira and Rendell [3]. The main idea of Relief is to compute a ranking score for every feature indicating how well this feature separates neighboring samples. The algorithm seeks for every training sample its nearest neighbor from the same class (nearest hit) and from the opposite class (nearest miss). The Relief score for a single feature is then the difference (or ratio) between the distance to the nearest miss and the distance to the nearest hit, projected on the specific feature. Several variants have been proposed. We give here a generalized and improved version of Relief which makes use of the k-nearest hits/misses and is based on the ratio:

Start with zero scores $\mathbf{h} = (0, ..., 0)$ and $\mathbf{m} = (0, ..., 0)^T$.
For every training sample $\mathbf{x} \in X$ do
  Find k-nearest hits $H(\mathbf{x})$ and k-nearest misses $M(\mathbf{x})$ of $\mathbf{x}$.
  Update $\mathbf{h} \mapsto \mathbf{h} + \mathbf{d}(\mathbf{y}, \mathbf{x})$ for all $\mathbf{y} \in H(\mathbf{x})$.
  Update $\mathbf{m} \mapsto \mathbf{m} + \mathbf{d}(\mathbf{y}, \mathbf{x})$ for all $\mathbf{y} \in M(\mathbf{x})$.
Compute the ratios $w_i = \frac{m_i}{h_i}$.

Here $\mathbf{d}(\cdot, \cdot)$ denotes the feature-wise distances, i.e. the i-th component of $\mathbf{d}(\cdot, \cdot)$ is the distance between x_i and y_i with an appropriate metric. The w_i determines the relevance of feature i. If we want to select j features, we take the j features with the highest w_i-scores. The algorithm has received a high popularity in the past, in particular due to its easy implementation, its good performance and the ability to also work in multi-modal environments, where simple ranking criterias like Pearson's correlation coefficients or Fisher's criterion fail. But for retrieval purposes Relief shows some drawbacks. In the following section we want to reveal under what circumstances Relief has its problems and propose modifications, which circumvent those problems.

[1] In the following we call a single feature unimodal if its within-class-distribution is nearly Gaussian and form a cluster. Otherwise, if the distribution consists of two or more clusters the feature is called multi-modal.

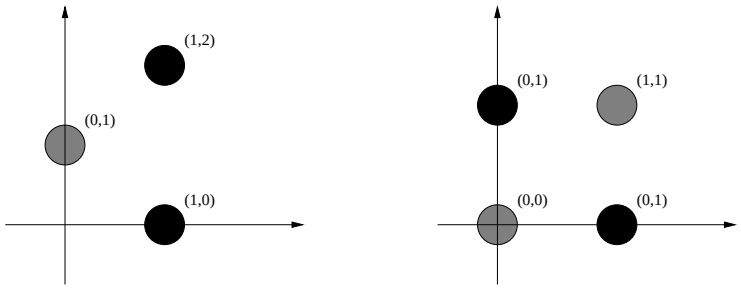

Fig. 1. a) Two class problem, where both dimensions are able to separate the classes, but the x-axis is better suited for retrieval purposes. b) The XOR problem. Only both dimensions together separate the classes.

2.2. The Method

To show this let us consider the two classes in figure 1a). Class A has two centers located at $(1, 2)$ and $(1, 0)$ and class B is located at $(0, 1)$ in the Euclidean plane. Obviously, the first dimension (or feature) easily separates the two classes. And, if we restrict to the first feature, a query for $(1, 2)$ actually gives a result from cluster $(1, 2)$ or cluster $(1, 0)$ of class A, this is the desired result. Though the second feature is also able to separate A and B (of course not linearly), a query for $(1, 2)$ retrieves members of B much earlier than objects around the second cluster $(1, 0)$ of class A. Such a behavior is clearly not desired for retrieval purposes. Although both features keep information about the class membership, one would favor the first dimension. But which dimension does Relief predict as more relevant? Since Relief only considers nearest hits, the algorithm is not able to establish the connection between the two clusters of class A, because cluster $(1, 0)$ is farther to cluster $(1, 2)$ than to the center of class B. In this case Relief selects the second dimension as more relevant. Hence we have to modify the algorithm such that also objects far away (in terms of the original distance measure) belonging to the query's class are taken into account. Instead of taking the nearest hits we should take the "farthest" hits, meaning those results belonging to the same class as the query but not found by the search, i.e. the false negative answers. We define $N(\mathbf{x})$ by the set of all objects, which are in the same class as $\mathbf{x}$ but are not found under the first c objects in the results list for query $\mathbf{x}$, where c denotes the number of class members in the queryclass. Similar we can define the set of all false positives $P(\mathbf{x})$ by all those objects, which do not belong to the same class as $\mathbf{x}$ but are found under the first c search results. We modify the original algorithm by using the sets $M(\mathbf{x})$ by $P(\mathbf{x})$ instead of $H(\mathbf{x})$ by $N(\mathbf{x})$:

Start with zero scores $\mathbf{n} = (0, ..., 0)$ and $\mathbf{p} = (0, ..., 0)^T$.
For every training sample $\mathbf{x} \in X$ do
  Find false negatives $N(\mathbf{x})$ and false positives $P(\mathbf{x})$ for query $\mathbf{x}$.
  Update $\mathbf{n} \mapsto \mathbf{n} + \mathbf{d}(\mathbf{y}, \mathbf{x})/||\mathbf{d}(\mathbf{y}, \mathbf{x})||$ for all $\mathbf{y} \in N(\mathbf{x})$.
  Update $\mathbf{p} \mapsto \mathbf{p} + \mathbf{d}(\mathbf{y}, \mathbf{x})/||\mathbf{d}(\mathbf{y}, \mathbf{x})||$ for all $\mathbf{y} \in P(\mathbf{x})$.
Compute the ratios $w_i = \frac{p_i}{\alpha + n_i}$.

In contrast to the original Relief algorithm we introduced a normalizing step in order to not overweight the distances to the false negatives $N(\mathbf{x})$, whose distances are naturally larger than the distances to the false positives. We also introduce a balancing parameter α to adapt the influence of the false negatives $\mathbf{N}$.

Considering the XOR problem from figure 1b, we have a closer look at the interplay of the counterparts $\mathbf{n}$ and $\mathbf{p}$. Suppose a feature vector with plenty of dimensions where the information containing features are feature one and two and the other features contain uncorrelated noise. One can imagine that both score vectors are of the same form $\mathbf{p} \approx (1, 1, ...)$ and $\mathbf{n} \approx (1, 1, ...)$, where the additional dimension contain mainly small noisy values. If we pay more attention to the false positive vector $\mathbf{p}$, i.e. we suppose large α, the relevance scores w_1 and w_2 are lifted and one can determine the first two features as the relevant ones. In the opposite case for small α we have a problem. Since $\mathbf{n}$ stands somehow for the non-relevance of the particular feature, the first two weight scores are damped and the algorithm has difficulties to determine the first two dimensions as the most relevant. We have to make a tradeoff between two demands. The false negatives scores $\mathbf{n}$ help us to merge different clusters like in the case in figure 1a. In other words, it detects the unimodal distributed features like Fisher's or Pearson's ranking criterias. But it also hinders the algorithm to find the relevant features in the XOR problem.

3. Experiments

In this section we present various experiments on toy and real-world data. For comparison we use three other ranking methods. The most simple method is Pearson's correlation coefficient[2], which is a very simple idea. We also used Fisher's coefficient, but due to the very similar behavior of both we restricted to Pearson's. As a second method we choose feature selection by Maximum Marginal Diversity (MMD) introduced by Vasconcelos [9]. The MMD idea is closely related to the infomax principle, which maximizes the mutual information between features and class labels. It was shown by Vasconcelos that for a special class of classification problems both principles are identical and in [10] he showed that it seems that vision related problems are actually a instance of those. Since the proposed feature selection method is based on Relief , of course, we also compare our experimental results to that. We use the version of Relief proposed above, where we adapted k, the number of neighbors, to the training set size. As Relief is not straightforward adaptable to multi-class problems, we used a version proposed by Kononenko. In [4] he compared two versions and showed that the so called Relief-F algorithm is superior. Instead of finding the k-nearest misses from the opposite classes, the algorithm finds the k-nearest misses for each different class and averages their contribution.

[2] Pearson's correlation coefficient for a feature x is given by $PCC = \frac{\sum(x_i-\bar{x})(c_i-\bar{c})}{\sqrt{\sum(x_i-\bar{x})^2 \sum(c_i-\bar{c})^2}}$, where the index i ranges over the whole training set and c_i are the class labels.

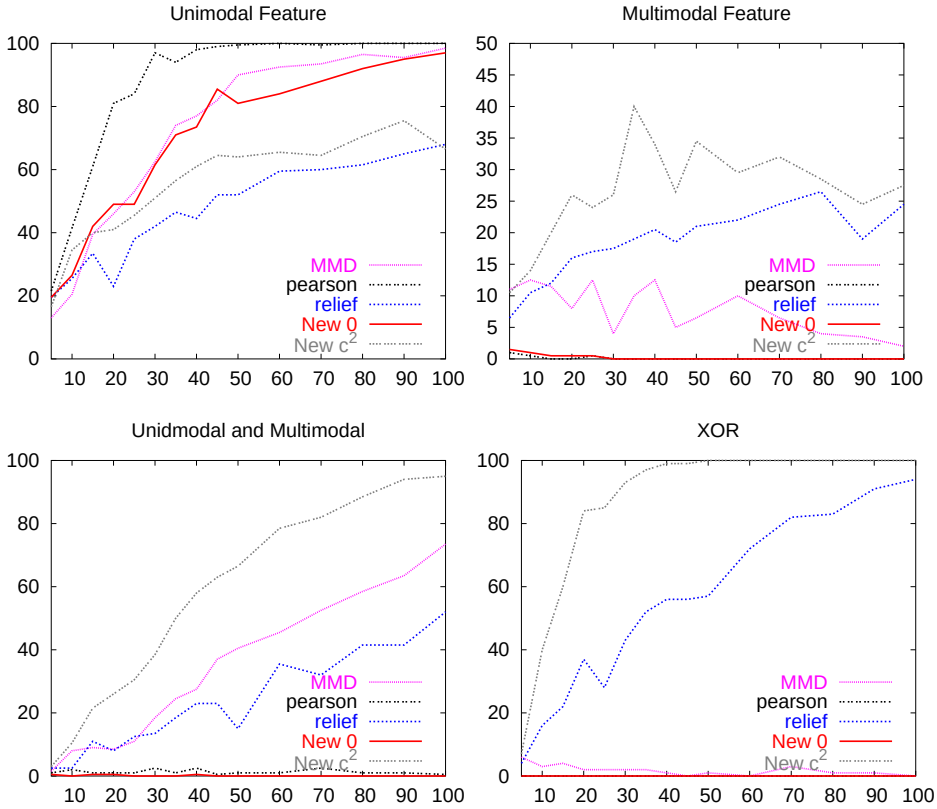

Fig. 2. Evaluation Plots for 2-class problem from figure 1a and the XOR problem from figure 1b. On the x-axis the number of training samples per class is given. The y-axis gives the percentage of correct detected features.

3.1. Toy Data

For the following three experiments we always use the Euclidean distance as the distance metric. First we consider the problem from figure 1a. Both classes occur with same probability. The feature vectors have length 20, where the first 2 features contain the information and the other 18 features are Gaussian noise with $\sigma = 1.0$. The clusters are also Gaussian distributed with the same standard deviation. We did three runs. For increasing number of training samples, we measured the percentage how often the algorithms detect the unimodal feature (x-axis in figure 1a) as most important or the multimodal feature (y-axis in figure 1a). Finally we measure how often both together are reported as the two most relevant features. To estimate the percentage each experiment is performed two hundred times. For the Relief algorithm and our new algorithm we choose $\alpha = 0$. But we also examined our algorithm with $\alpha = c^2$, where c is the number of training samples per class. It is obvious that the scaling behavior of α with respect to the number of training samples is quadratic. As already mentioned

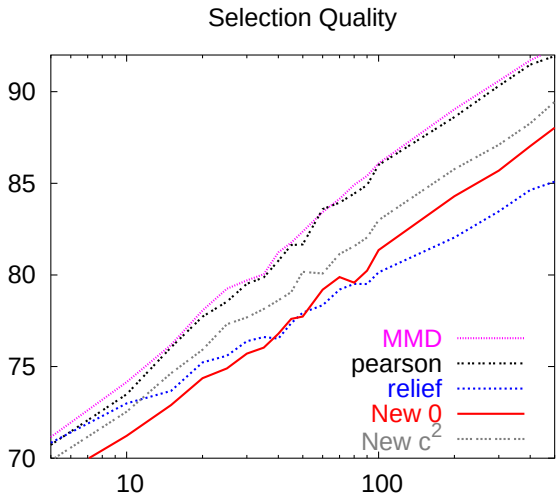

Fig. 3. Evaluation Plots for the problem introduced by Trunk. On the logarithmic x-axis the number of training samples per class is given. The y-axis gives the quality of the feature subset.

large α-values lead to a down weighting of false negatives. The results are given in figure 2. Pearson's correlation coefficient easily detects the unimodal feature and outperforms all other methods in this case. MMD performs very similar to the new method with $\alpha = 0$. Like expected for $\alpha = c^2$ our method is not able to detect the unimodal feature with the same reliability as for $\alpha = 0$. But for $\alpha = 0$ it slightly outperforms the Relief algorithm, which approves our modifications. For the multimodal feature the situation is inverted. Pearson's coefficient and our method for $\alpha = 0$ are totally unable to find the feature. Also MMD does not perform very well. Our method for $\alpha = c^2$ beats Relief and all other methods. In the last graph we see that its overall performance is also the best. Our method actually finds both features most. Only MMD can compete a little bit, further below Relief is still not too bad, but our method for $\alpha = 0$ and Pearson's totally fail.

As a second experiment we consider the XOR problem. The training data is created like above by a Gaussian distributed process with means (0,1), (1,0) for class A and (1,1), (0,0) for class B and with standard deviation $\sigma = 1$. Again we measure the percentage that both features are detected as most relevant. In figure 2 the results are presented. Only Relief and our method can cope with this difficult case, but our method is much more reliable in detecting the relevant features. Only 50 training samples per class are enough to get 100% detection ratio.

As a last experiment we adopt a evaluation method introduced by Trunk [8] which was also used by Jain and Zonkger [2] and Vasconcelos [9]. To evaluate the reliability of feature selection methods a 20 dimensional feature vector is

created, every feature is Gaussian distributed with $\sigma = 1$ and means $\mu_i = \frac{1}{\sqrt{i}}$ for class A and $\mu_i = -\frac{1}{\sqrt{i}}$ for class B, where the i stands for the i-th feature. The optimal j-subset is obviously the set of the first j features. The quality of the feature subset is computed by taking the number of commonalities in the optimal feature subset with the subset generated by the feature selection algorithm. This number is divided by the number of dimensions and averaged over the subset size j from 1 to 19. In figure 3 the results are given. Pearson's coefficient and MMD outperform the three other methods, both methods have also the highest slope, followed by our methods, which have both nearly the same slope. Relief seems to have the lowest slope, but for very small training set sizes it can compete with the others.

3.2. Image and Shape Retrieval

We consider three databases, two image databases and one 3D-Shape database. All three are divided in a test and training set. For Relief and our method we optimize the parameter K, and the parameter α by computing both, the ranking criteria and the performance results, on the training set. The feature rankings obtained with optimal parameters are then used for evaluation on the test set. As a performance measure we mainly use the First-Tier (1T) measure introduced in [7], since this measure focuses on the demands a user has on a retrieval system. The 1T-rate measures the percentage of objects in the query's class that appear within the top C matches, where C is the number of members in the query's class minus one. These statistics is similar to the "Bulls Eye Percentage Score".

ETH 80 database The database was introduced by Leibe and Schiele in [5]. It consists of eight categories of high resolution color images. Each object is represented by 41 images from viewpoints spaced equally over the upper viewing hemisphere. We decided to consider the four most difficult categories, namely dogs, horses, cows and cars. The training set is of size 660, the test set is of size 1010. Sometimes it is even difficult for a human to discriminate if the viewpoint is unfortunately chosen. Since we want to evaluate feature selection algorithms, a large set of features is computed for each image. Each feature set is a four dimensional histogram of the distance between two randomly chosen points in the image, the relative orientation of the gradients at those points (the normalized dot-product, i.e. the cosine), the relative orientation between the gradient and the vector connecting both points and the absolute difference of the intensity values at the chosen points. The random points are chosen by the probability proportional to the gradient magnitude. As we deal with color images we apply this procedure for every color channel (RGB) and also crosswise, meaning that one random point is chosen from e.g. the R-channel and one from the G-channel. This results in 6 histograms for every combination (RR,GG,BB,RG,RB,GB). The overall feature size is then $\#f = \#bins * 6$, where in the experiments we chose $\#bins = 8 * 4 * 4 * 4 = 512$, resulting in $\#f = 3072$ features. The resulting histograms are all invariant to rotation and translation of the image plane

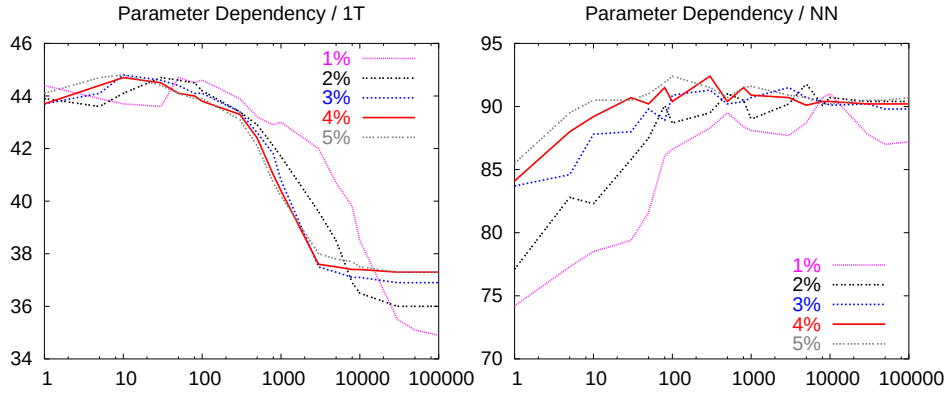

Fig. 4. Parameter Dependency on the ETH80 database with invariant features. On the x-axis the parameter α is drawn in logarithmic scale. On y-axis left the 1T-measure is given and on the right the Nearest-Neighbor Classification Accuracy (NN). The plot is evaluated for several fixed feature subset sizes given in percentage of the original size.

and illumination changes. Since rotation invariance is not really necessary, we additionally computed two histograms, where the drawn samples are weighted by the cosine/sine of the absolute angle of the vector connecting the randomly chosen points, resulting in a feature size of 9216. We always compare the histograms using the L_1-norm. The L_1-norm is closely related to the Histogram-Intersection-Kernel which predestines it for our purposes. To gain an intuition how the parameter α of our method has to be chosen, we recorded the 1T-rate for different fixed feature sizes. Additionally we also consider the Nearest-Neighbor Classification Accuracy (NN), meaning the rate that an object of the query's class appears as first. In figure 4 the corresponding plots are shown. The 1T- and NN-measure evolve in a opposite manner. While 1T has a diffuse maximum for small α and decrease with growing α, NN shows higher values for large α, i.e. for large α the precision of detecting first an object from the same class rises, but there are still lots of objects which cannot be found. We know that for large α the algorithm only considers negative samples nearby the query, positive samples far away are not considered which explains that in this case the algorithm is not able to merge different clusters belonging to the same class and hence shows a bad 1T-rate. We have to find a tradeoff between merging the intra-class clusters and obtaining a high NN-rate.

In the experiments we focus on the 1T measure and tune α to get a high 1T-rate, but still keep α as large as possible to obtain a high NN-rate as well. In other words we try to find the turning point of the 1T-rate for increasing α. All the pre-tuning is done manually for a fixed feature size of 5% from the original size.

In figure 5 the results for the invariant and variant features are shown. For the NN-rate our new algorithm is able to beat all the other three methods. For the 1T-measure our method obtains the highest scores for very small feature

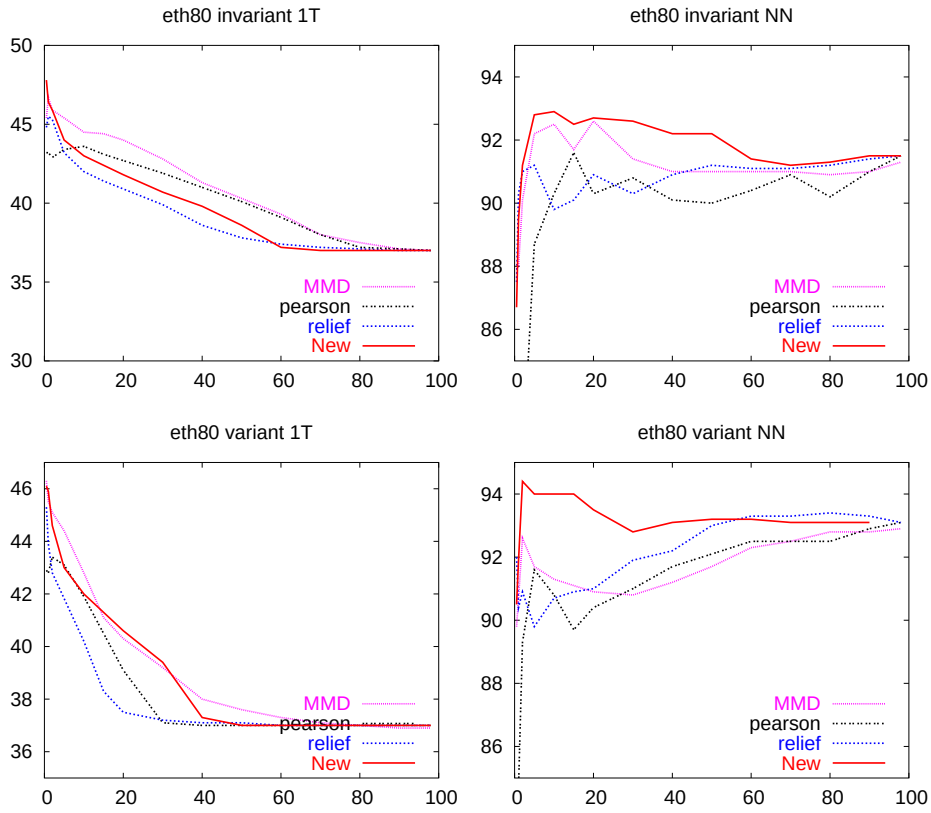

Fig. 5. Results for the ETH80 database. On the x-axis the feature size is given (in percentage of the original size). On the y-axis the 1T-rate and NN-rate respectively.

sizes (the first plot point is at 0.5%) and also the highest overall 1T-rates. But for growing feature sizes the other methods, in particular MMD outperforms our algorithm with respect to the 1T-rate. It is astonishing that very small feature sizes (15 invariant features, 45 invariant features) lead to the highest 1T-rates. It seems that there is only a very small set of features which show a low intra-class variability. One can also notice that for both measures it is always worse to use all features instead of a cleverly selected subset.

Caltech database As a second problem we consider a 2-class problem. Caltech (http://www.robots.ox.ac.uk/ vgg/data3.html) provides several image databases. In our experiments we tried to discriminate between a set of airplanes and background (training set 687, testing set 987), and secondly between a set motorbikes and background (training set 750, testing set 1005). We used the same variant features as above. One may argue that it is a little bit naive to use such global features for a dataset with a complex background, but since we are interested in examination of feature selection algorithms the use of such 'dirty' features is jus-

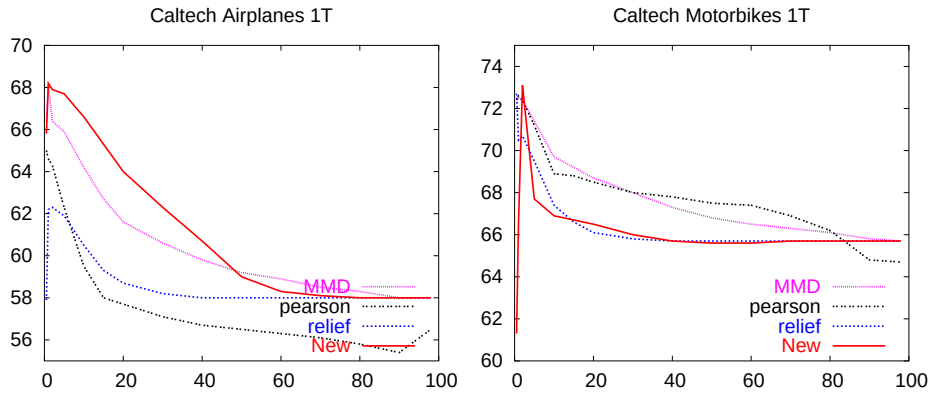

Fig. 6. Results for the Caltech database. On the x-axis the feature size is given (in percentage of the original size), on the y-axis the 1T-rate.

tified. Since the provided background image set is only in gray-value format, we only used the luminance information and neglected color. In figure 6 the results are presented. This time we only present the results for the 1T-measure. For the airplanes our method again outperforms the others. But for the motorbikes our method shows a little strange behavior. For the very small sizes the performance is very bad, for 2% it shows a very high peak and then again falls down to very low performance. Relief also performs very bad on this dataset.

Princeton Shape Benchmark (PSB) Finally we deal with the Princeton Shape Benchmark provided in [7]. The PSB consists of 1814 triangulated 3D-surface models. The database is divided into a test and training set, both of size 907. The authors also provide a hierarchical classification. We work with the three finest granularities, level 0: about 90 classes, level 1: about 40 classes, level 2: 7 classes. The used features are proposed in [6]. This time three dimensional histograms are used: the joint distribution of the distance between two equidistributed random points on the surface of the object, the relative orientation of the surface normals at those points and the relative orientation of one normal with the vector connecting both points. To keep more information we compute for each bin the angular distribution of the connecting vector falling into this bin. Since invariance to 3D-rotation is necessary a Spherical Harmonic Transform of the angular distribution is computed and the norms of the coefficient vectors are stored, which are known to be invariant to rotation. The resulting feature (SD) is of size 768. But it is also possible to preserve much more information. Instead of taking the norm for each SH-coefficient vector alone, crosswise dot-products of coefficient vectors from different bins are also invariant to rotation. Applying this procedure the feature size grows quadratically in the number of bins of the underlying histogram. The examined features (SDF) are in this case of size 6240. Again the same optimization procedure was applied like in the image re-

trieval case. The results for the 1T-rate are given in figure 7. Again our proposed method shows very nice results and seems to be superior in nearly all situations. Only for very small feature sizes the MMD method sometimes outperforms our algorithm, but in all cases our method achieves the highest overall 1T-rate.

4. Conclusion and Future Work

We have introduced a new feature selection algorithm, which is based on the well known Relief algorithm and is very well suited for retrieval purposes. We well motivated our modification and demonstrated that on both toy and real-world data our algorithm shows a good performance in comparison to other methods. It works quite well on a great diversity of databases.

A nice property of our algorithm is a parameter which helps us to find a tradeoff between the accuracy finding nearest neighbors from the query's class and the amount of false negatives within the first few result objects. But this parameter is also the main drawback of our algorithm, an automated estimation of this parameter with respect to the user's demands would be very helpful.

References

1. I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.
2. A. Jain and D. Zongker. Feature selection: Evaluation, application and small sample performance. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 21(3):153–158, 1997.
3. K. Kira and L. Rendell. A pratical approach to feature selection. In *Proceedings of the International Conference on Machine Learning. (Aberdeen 1992) Editors: D.Sleeman and P.Edwards, Morgan Kaufmann*, pages 249–256.
4. I. Kononenko. Estimating attributes: Analysis and extensions of relief. In *Proceedings of ECML-94, Catania, Sicily, April 1994 (F.Bergadano, L.de Readt (eds.)),Springer Verlag*, pages 171–182.
5. B. Leibe and B. Schiele. Analyzing appearance and contour based methods for object categorization. In *Proceedings of CVPR 2003, Madison, Wisconsin*, 2003.
6. M. Reisert and H.Burkhardt. Second order 3d shape features: An exhaustive study. *accepted for publication in Computers and Graphics, Special Issue on Shape Reasoning and Understanding (publication date: April 2006)*, 30(2), 2006.
7. P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser. The princeton shape benchmark. In *Shape Modeling International, Genova, Italy*, 2004.
8. G. Trunk. A problem of dimensionality. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 1(3):306–307, 1979.
9. N. Vasconcelos. Feature selection by maxiumum marginal diversity. In *Proceedings of NIPS 2002*, pages 1351–1358.
10. N. Vasconcelos. Feature selection by maxiumum marginal diversity: optimality and implications for visual recognition. In *Proceedings of CVPR 2003*, volume 1, pages 762–772.

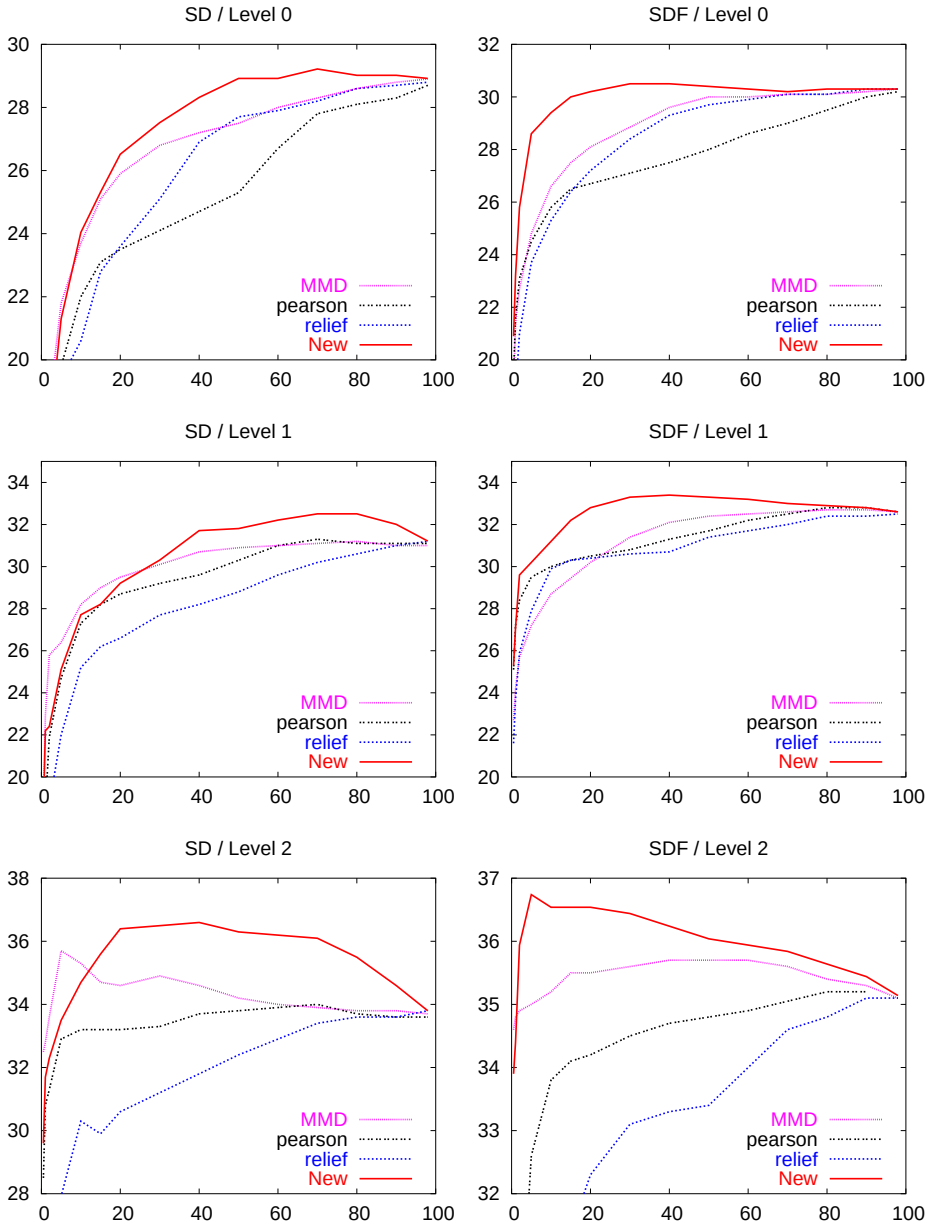

Fig. 7. Results for the Princeton Shape Benchmark. On the x-axis the feature size is given (in percentage of the original size), on the y-axis the 1T-rate.