Appl Intell (2008) 28: 238–246
DOI 10.1007/s10489-007-0064-0

Prototype-based classification

Petra Perner

Published online: 1 June 2007

Abstract Image-based diagnostic tools are important tools for the determination of diseases in many medical applications. The interpretation of these images is often done manually, based on prototypical images. Consequently, only a few images collected into an image catalogue are initially available as a basis for the development of an automatic image-interpretation system. In this paper we study the question if it is possible to build up an image-interpretation system based on such an image catalogue. We call the system catalogue-based image classifier. The system is provided with feature-subset selection, feature weighting, and prototype selection. The performance of the catalogue-based classifier is assessed by studying the accuracy and the reduction of the prototypes after applying a prototype-selection algorithm. We describe the results that could be achieved and give an outlook for further developments on a catalogue-based classifier.

Keywords Image classification · Case-based reasoning · Feature-weight learning · Feature-subset selection · Prototype selection · Cases · Prototypes

1 Introduction

Although digital image scanners and cameras are quite common in many applications, it is still not standard to have a sufficiently large database as a basis for the development of an image-interpretation system. People rather like to store image catalogues comprised of one prototypical image for each class, instead of constantly collecting images into a database. This is especially true for medical [1] and biological applications [2]. A wide spectrum of image catalogues is available, published as books or in web-catalogues, showing for example brain CT or MRT images of degenerative brain disease, images of different skin diseases, cell-pattern images of malaria-infected blood cells, pap smear cell images, autoantibody-related HEp-2 cell pattern images, images showing different fungi spores, etc. These image catalogues should guide and train a human when interpreting the images. Automatic image-interpretation systems for these tasks are not of up-to-date standard in practice. It is often difficult to develop an automatic image-interpretation system for a specific application, due to the lack of a sufficiently large set of images as a basis for its development and owing to the difficulties arising from the automatic image description and the image-interpretation problem.

The prototypical images stored in these catalogues represent the visual appearance of a pattern, a scene, or an object. In this respect it is summarized information about the topic of interest and does not represent the coding rules for this image [3]. The information about the visual features, their importance and the form in which they are expressed in the image is implicitly represented in the respective image.

The usage of prototypical images for the development of an automatic image-interpretation system is therefore concerned with analyzing the image for visual features, determining their importance, and rejecting unimportant details and background information, as well as with constructing the classifier. It seems to be natural to use case-based reasoning as a problem-solving method for this kind of application, since case-based reasoning can start to reason based on a few samples and can incrementally collect more cases for each class and learn, based on these cases, more precise

P. Perner (✉)
Institute of Computer Vision and Applied Computer Sciences,
PF 3011 14, 04251 Leipzig, Germany
e-mail: pperner@ibai-institut.de

class representations during the usage of the system and the system performance can be improved.

The use of case-based reasoning in applications with prototypical cases has been successfully studied for medical applications by Schmidt and Gierl [4], Belazzi et al. [5] and by Nilsson and Funk [6] on time-series data. The simple nearest-neighbor-approach [7] as well as hierarchical indexing and retrieval methods [4] have been applied to the problem. It has been shown that an initial reasoning system could be built up based on these cases. The systems are useful in practice and can acquire new cases for further reasoning [8] while running. A deep evaluation was often not possible, since the broad variation of the problem could not be captured by the few initial prototypes. Due to the sparse data-set problems, such topics as case editing and condensing have not been studied in these application areas. There are comparative studies of case editing for other applications, such as SPAM filtering by Delany and Cunnigham [9], a case-authoring system by McKenna and Smyth [10] and Ontañón and Plaza for symbolic data [11].

The problem of feature-subset selection [12] and feature weighting [13] is empirically studied on standard machine-learning data bases, as well as on real-world data.

We have developed case-based reasoning methods for all the different stages of an image-interpretation system [14]. Usually the methods were evaluated by a case-base that had been collected by the domain expert, but the cases were not considered as prototypical cases.

In this paper we are studying the following question: is it possible to build an image-interpretation system based on a prototypical image for each class and if so, what necessary functions should such a system have? The basis for our study were four different image catalogues of HEp-2 cell images collected by different manufacturers of HEp-2 cell diagnostica and by a diagnostic laboratory. HEp-2 cell stands for the human epithelial cell line type 2. HEp-2 cells are used for the identification of antinuclear autoantibodies (ANA). ANA testing for the assessment of systemic and organ-specific autoimmune diseases has increased progressively since immunofluorescence techniques were first used to demonstrate antinuclear antibodies in 1957. HEp-2 cells allow for recognition of over 30 different nuclear and cytoplasmic patterns, which are given by upwards of 100 different autoantibodies [15]. The catalogues for the different cell patterns are comprised of one prototypical image for each pattern class [15].

Until a short time ago the identification of the patterns has been done manually by a human inspecting the slides with the help of a microscope. The lacking automation of this technique has resulted in the development of alternative techniques based on chemical reactions, which do not have the discriminating power of ANA testing. An automatic system will pave the way for a wider use of ANA testing. A first commercial system has been available since a short time ago, but the effort for collecting enough high-quality images as a development basis for such a system were big and time-consuming. This fact motivated our research on a catalogue-based classifier.

In Sect. 2 we shortly describe the image catalogues. The image analysis and the features extracted for the objects in the image are described in Sect. 3. The resulting databases are described in Sect. 4. The nearest-neighbor classifier and the feature selection and feature-weighting algorithm are described in Sect. 5. Likewise we describe in Sect. 5 the prototype-selection method that allows us to find out the quality of the image catalogues. Finally, we give results in Sect. 6 and an outlook for further work in Sect. 7.

2 Cases and prototypes

Let X be a set of cases collected in a case base CB. Each case is represented by a tuple $X = \{(F, v_i), (S, s)\}$ where F is the set of descriptive features, v is the set of feature values and S is the solution space with the possible s. The relation between each case in the case base can be expressed by the similarity value *sim*. The case base can be partitioned into n case classes C: $CB = \bigcup_{i=1}^{n} C_i$ such that the intra-case-class similarity is high and the inter-case-class similarity is low. The set of cases in each case class C can be represented by a representative who generally describes the cluster. This representative can be the prototype, the medoid, or an a-priori-selected case. Whereas the prototype implies that the representative is the mean of the cluster which can easily be calculated from numerical data, the medoid is the case whose sum of all distances to all other cases in a cluster is minimal.

A prototypical case must not necessarily be comprised of the whole set of features describing the cases [16]. It can be represented by only a subset of features that represents the most common features. Besides that not all features might have the same importance.

The relation between the different case classes C can be expressed by higher-order constructs expressed e.g. as super classes that give us a hierarchical structure over the case base. The top-level of this structure represents the most common case class, while the leaves are the most specific case classes. This hierarchical structure helps to guide and speed up the retrieval process in a case-based reasoning system and also to control the size of the case base. This aspect will not be considered in this paper. We assume a flat organization of the case base.

Usually, prototypical cases are generalized from a set of single cases. In our case we have prototypical cases represented as images that have been selected by humans. The most common features are not reported and neither is the

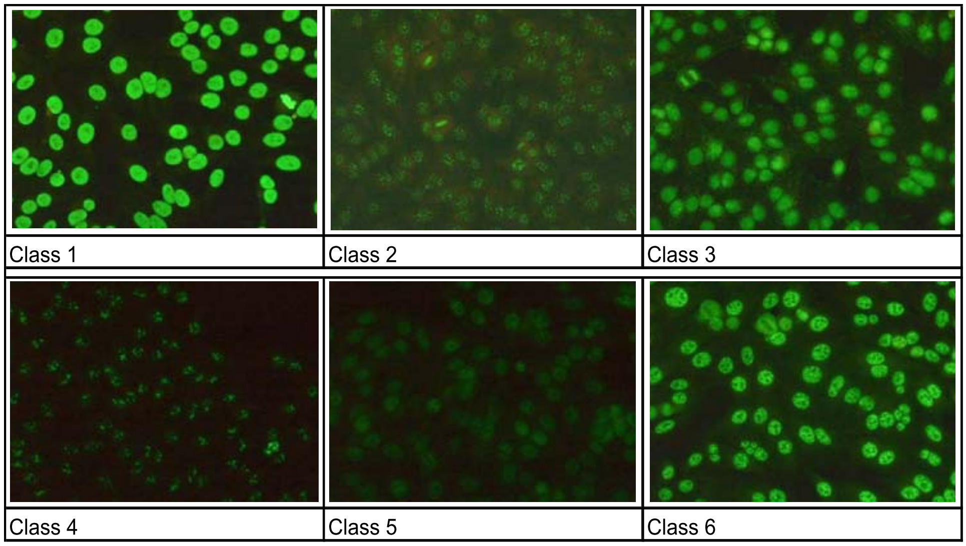


Fig. 1 Prototypical images of six classes

range of values reported. That means, when building our system, we are starting from the top and have to collect more information about the specific class during the usage of the system.

Since a human has selected the prototypical images, his decision on the importance of an image might be biased, and to pick only one image might be difficult for a human. He can have stored more than one image as prototypical images. Therefore, we need to check the redundancy of the many prototypes for one class before taking them all into the case base.

According to this consideration, the minimal functions that our system should carry out are:

- classification based on the nearest neighbor rule,
- prototype selection by a redundancy-reduction algorithm,
- feature weighting to determine the importance of the features for the prototypes and
- feature-subset selection to select the relevant features from the whole set of features for the respective domain.

3 Image catalogues

Image Catalogues usually show one prototypical image for one class. These catalogues represent the visual knowledge of an image-inspection domain. They are available as hardcopies or as a collection of digital images. They are usually used to train novices for the specific image-inspection task. Together with the prototypical images a verbal description of the appearance of the pattern is sometimes represented. Unlike in other tasks, such as industrial inspection, these verbal descriptions are no standardized visual image descriptions allowing a human to build up conceptual knowledge. Therefore, as long as no automatic image-interpretation system is available, a lot of work is invested in order to build up ontology to improve the overall quality of image-inspection tasks.

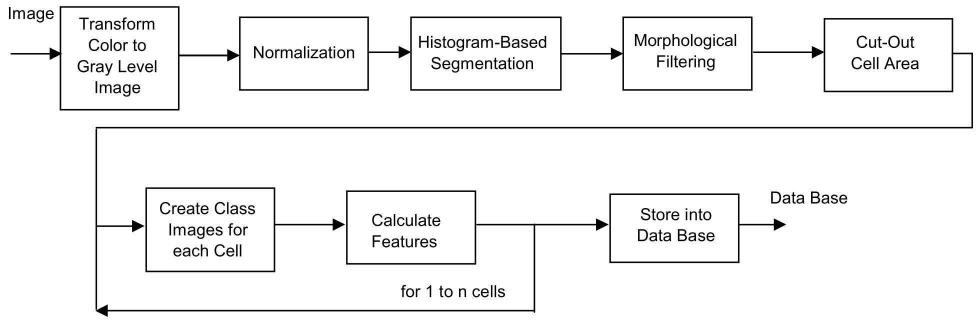
Prototypical images of HEp-2 cell patterns for six different classes are shown in Fig. 1. We should note that for each class we have only one prototypical image. The image shows the visual appearance of a particular pattern on the cells. Since in each image numerous cells are contained, we are getting many cell patterns of the same kind. To select the pattern on a single cell as a prototypical pattern might be difficult for a human, since he is not trained based on a single cell pattern. He rather looks for the overall visual appearance of a pattern on the cell rage in the image. Therefore we have an unknown number of single cells in each image showing the pattern of interest for each class. Each cell could be considered as a prototype itself. That will result in a data base of prototypes with an unequal number of samples for each class.

4 Creation of the database

Each image is processed by the image-analysis procedure described in [17]. The flowchart of the overall algorithm is given in Fig. 2. The color image (see Fig. 3) is transformed into a gray-level image. The image is normalized to the mean and standard gray level calculated from all images to avoid invariance caused by the inter-slice staining variations.

Automatic thresholding has been performed by the algorithm of Otsu [18]. The algorithm can localize the cells with their cytoplasmatic structure very well, but not the nuclear envelope itself. We then apply morphological filters like dilation and erosion to the image, in order to get a binary mask for cutting out the cells from the image (see Fig. 4). The au-

Fig. 2 Creation of data base



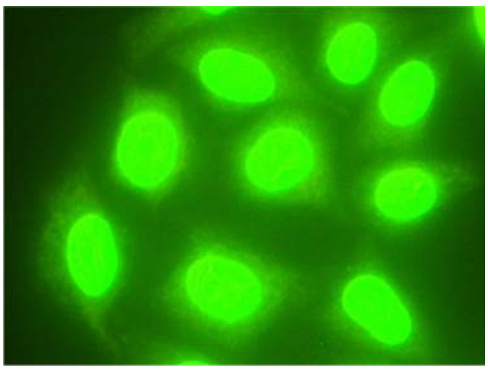


Fig. 3 Original image

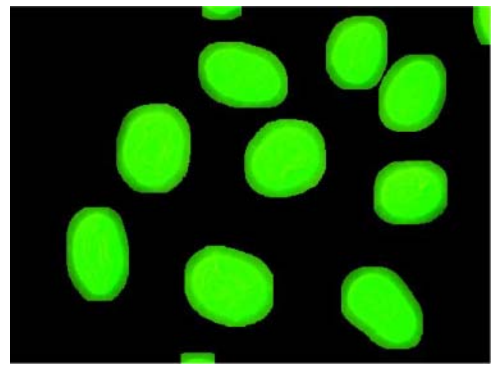


Fig. 5 Cut-out of cell area

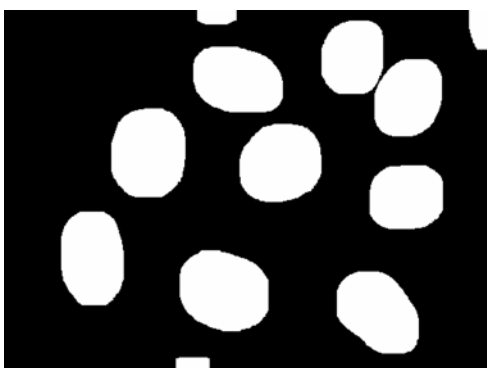


Fig. 4 Image after filtering

tomatically found cell areas after applying the binary mask are shown in Fig. 5.

The gray levels ranging from 0 to 255 are quantized into 16 intervals t (see Fig. 6). Each subimage $f(x, y)$ containing only one cell gets classified according to the gray level into t slices, with $t = \{0, 1, 2, \ldots, 15\}$. For each slice a binary image is calculated, containing the value "1" for pixels with a gray-level value falling into the gray-level interval of slice t and value "0" for all other pixels. We call the image $f(x, y, t)$ in the following slice image. Object labeling is done in the slice images with the contour-following method.

For the objects in each slice features are calculated for classification. The first one is a simple Boolean feature which expresses the occurrence or non-occurrence of objects in the slice image. Then the number of objects in the slice image is calculated. From the objects the area, a shape factor, and the length of the contour are calculated (see Table 1). The mean value for each feature is calculated over all the objects in the slice image. This is done in order to reduce the dimension of the feature vector. Since the quantization of the gray level was done in equal steps and without considering the real nature, we also calculated for each class the mean value of the gray level and the variance of the gray level. A total of 192 features were calculated that make up a very intelligent structure and texture descriptor for cells.

It has been shown in [17] that not all features of the complex structure and texture descriptor are relevant for a particular texture description. The feature selection procedure of the decision-tree induction process used in the study described in [17] for building the classifier selects only a subset of features from the whole set of features. However, using decision-tree induction requires a sufficiently large sample set which is not the assumption in this study. When producing as many features as possible for an object without being informed about the relevance of the features, a good feature-subset selection strategy is required, no matter which classifier is used.

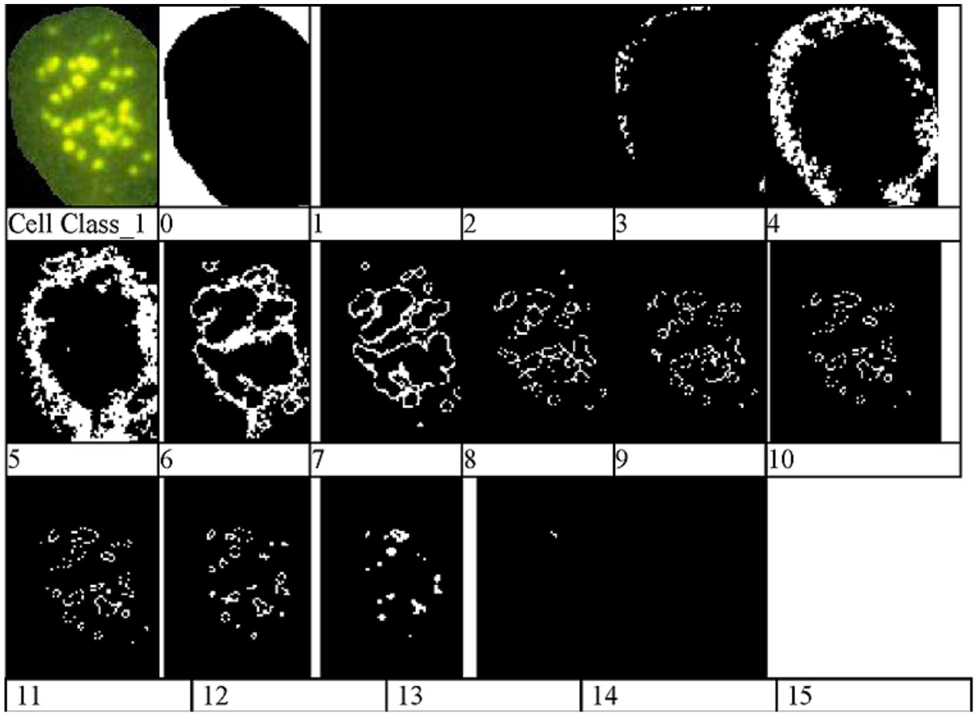


Fig. 6 Original cell and class images

5 The databases

For our investigation we used four different image catalogues that came from four different manufacturers of HEp-2 cells. The databases differ in the number of classes and in the number of samples per class, see Table 2. It cannot be expected that the distribution of the samples or the classes, respectively, is the same in each database. It is more likely that quite the contrary can happen. Although most of the databases were obtained from the manufacturers, they could not provide samples for all classes.

These image data bases were processed according to the image-analysis procedure described in Sect. 4. For each of the image data bases we obtained a feature-value data base that was the basis for the following study.

6 The classification method

The classification method is based on the nearest-neighbor rule. Since the prototypes are available at the same time, we choose a decremental redundancy-reduction algorithm proposed by Chang [19] that deletes prototypes as long as the classification accuracy does not decrease. The feature-subset selection is based on the wrapper approach [20] and an empirical feature-weight learning method [13] is used. Cross validation is used to estimate the classification accuracy. The prototype selection, the feature selection, and the feature weighting steps are performed during each run of the cross-validation process.

6.1 Nearest-neighbor rule

This nearest-neighbor rule [7] classifies x in the category of its nearest neighbor. More precisely, we call $x'_n \in \{x_1, x_2, \ldots, x_i, \ldots, x_n\}$ a nearest-neighbor to x if $\min d(x_i, x) = d(x'_n, x)$, where $i = 1, 2, \ldots, n$.

The nearest-neighbor rule chooses to classify x into category C_n, where x'_n is the nearest neighbor to x and x'_n belongs to class C_n.

In the case of the k-nearest neighbor we require k-samples of the same class to fulfill the decision rule. As a distance measure we use the Euclidean distance.

6.2 Prototype selection by Chang's algorithm

For the selection of the right number of prototypes we used Chang's algorithm [19]. The outline of the algorithm can be described as follows: Suppose the set T is given as $T = \{t^1, \ldots, t^i, \ldots, t^m\}$ with t^i as the i-th initial prototype. The idea of the algorithm is as follows: We start with every point in T as a prototype. We then successively merge any two closest prototypes t^1 and t^2 of the same class by a new prototype t, if the merging will not downgrade the classification of the patterns in T. The new prototype t may simply be the average vector of t^1 and t^2. We continue the merging process until the number of incorrect classifications of the pattern in T starts to increase.

Roughly, the algorithm can be stated as follows: Given a training set T, the initial prototypes are just the points of T. At any stage the prototypes belong to one of two sets—set

Table 1 List of features and their calculation

Description	Name	Type	Formula
Area of the single cell	A_{cell}	numerical	$A_{\text{cell}} = \begin{cases} f(x,y,t) = 1 \text{ and object then } A_{\text{cell}} = A_{\text{cell}} + 1 \\ f(x,y,t) = 0 \text{ then } A_{\text{cell}} = A_{\text{cell}} \end{cases}$
Density in class image t	*Dens_t*	numerical	$Dens\_t = \begin{cases} f(x,y,t) = 1 \text{then } Dens\_t = Dens\_t + 1/A_{\text{cell}} \\ f(x,y,t) = 0 \text{ then } Dens\_t = Dens\_t \end{cases}$
Number of objects	*Count_t*	numerical	$n(t)$
Mean area of objects in class image t	*Marea_t*	numerical	$\overline{A(t)} = \frac{1}{n(t)} \sum_{i=1}^{n(t)} A_i(t)$
Standard deviation of the area of the objects in class image t	*Staarea_t*	numerical	$S(t) = \sqrt{\frac{1}{n(t)} \sum_{i=1}^{n(t)} (A_i(t) - \overline{A(t)})^2}$
Relative mean area of objects in class image t to area of cell	*Rmarea_t*	numerical	$RA(t) = \frac{\bar{A}(t)}{A_{\text{cell}}}$
Relative standard deviation of the area of the objects in class image t	*Rstaarea_t*	numerical	$RS(t) = \sqrt{\frac{S(t)^2}{A_{\text{cell}}}}$
Mean shape factor for objects in class image t	*Form_t*	numerical	$\bar{F}(t) = \frac{1}{n(t)} \sum_{i=1}^{n(t)} 10 \cdot \frac{A_i(t)}{u_i(t)}$ with $u_i(t)_{\text{contour}}$ being the length of the i-th object in class image t
The contour length of a single object is $u = l + \sqrt{2} \cdot m$ with l being the number of contour pixels having odd chain coding numbers and m being the number of contour pixels having even chain coding numbers			
Mean contour length of objects in class image t	*Mlength_t*	numerical	$\bar{u}(t) = \frac{1}{n(t)} \sum_{i=1}^{n(t)} u_i(t)$
Standard deviation of the contour length of objects in class image t	*Stalength_t*	numerical	$S_u(t) = \sqrt{\frac{1}{n(t)} \sum_{i=1}^{n(t)} (u_i(t) - \overline{u(t)})^2}$
Relative mean contour length of objects in class image t	*Rmlength_t*	numerical	$\overline{ru(t)} = \frac{\overline{u(t)}}{A_{\text{cell}}}$
Relative standard deviation of the contour length of objects in class image t	*Rstalength_t*	numerical	$RS_u(t) = \sqrt{\frac{S(t)^2}{A_{\text{cell}}}}$

Table 2 Name of database and number of classes and samples per class

Name	Class number																										Number of classes	Number of classes
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26		
DB_1	105	96	63	83																							4	347
DB_2	8	2	2	14	7	5	15	9	5	4	14	9	8	10	48	23	17	31	3	3	3	7	5	2	13	5	26	298
DB_3	7	30	29	28	11	7	13	5	13	13																	10	156
DB_4	25	12	18	21	5	16	22	24	21	20	5	14															12	203

A or set B. Initially, A is empty and B is equal to T. We start with an arbitrary point in B and initially assign it to A. Find a point p in A and a point q in B, such that the distance between p and q is the shortest among all distances between points of A and B. Try to merge p and q. That is, if p and q are of the same class, compute a vector p^* in terms of p and q. If replacing p and q by p^* does not decrease the recognition rate for T, merging is successful. In this case, delete p and q from A and B, respectively, and put p^* into A, and the procedure is repeated once again. In the case that p and q cannot be merged, i.e. if either p and q are not of the same class or merging is unsuccessful, move

Fig. 7 Prototype selection algorithm

$T = \{t^1, \ldots, t^m\}$
$B \leftarrow T$
Define ∇
I: Arbitrarily select a point v in B
$A \leftarrow \{v\}$
$B \leftarrow B - \{v\}$
$MERGE \leftarrow 0$
II: **IF** B is empty **THEN**
IF $MERGE = 0$ **THEN** A is a set of prototypes for a nearest neighbour classifier **STOP**
ELSE $B \leftarrow A$ **GO TO I**
ELSE
Find a point p in A and a point q in B such that the distance between p and q is the shortest among all distances between points of A and B
IF class(p) = class(q) **THEN** calculate median p^* of p and q ,and associate class to p^*
IF incorrect classification increases by ∇ **THEN** do not replace p and q by p^*
$A \leftarrow A \cup \{q\}$
$B \leftarrow B - \{q\}$
ELSE
$A \leftarrow (A \cdot \{p\}) \cup \{p^*\}$
$B \leftarrow B \cdot \{q\}$
$MERGE \leftarrow MERGE + 1$
REPEAT with II

q from B to A, and the procedure is repeated. When B becomes empty, recycle the whole procedure by letting B be the final A obtained from the previous cycle, and by resetting A to be the empty set. This process stops when no new merged prototypes are obtained. The final prototypes in A are then used in a nearest-neighbor classifier (see Fig. 7).

6.3 Feature-subset selection and feature weighting

The wrapper approach [20] is used for selecting a feature subset from the whole set of features and for feature weighting. This approach conducts a search for a good feature subset by using the k-NN classifier itself as an evaluation function. By doing so the specific behavior of the classification methods is taken into account. The leave-one-out cross-validation method is used for estimating the classification accuracy. Cross-validation is especially suitable for a small data set. The best-first search strategy is used for the search over the state space of possible feature combination. The algorithm terminates if we have not found an improved accuracy over the last k search states.

The feature combination that gave the best classification accuracy is the remaining feature subset. We then try to further improve our classifier by applying a feature-weighting tuning-technique in order to get real weights for the binary weights.

The weights of each feature w_i are changed by a constant value δ: $w_i := w_i \pm \delta$. If the new weight causes an improvement of the classification accuracy, then the weight will be updated accordingly; otherwise, the weight will remain as it is. After the last weight has been tested, the constant δ will be divided into half and the procedure repeats. The process terminates if the difference between the classification accuracy of two interactions is less than a predefined threshold.

Some researchers [22] have noted overfitting to the data when using small sample sets and the cross-validated accuracy estimates for guiding the search in the wrapper approach. In their experiments they use decision tree induction for the classifier that naturally tends to overfit to the data. Overfitting means that the specialization of the learnt model is too high and the generalization capability of the model is not as good as it should be [23]. We like to remind the reader that our strategy is to start with a model for classification based on few samples and then incrementally improve the model based on the new acquired cases and case classes. That might also require that we have to reconsider the chosen prototypes, feature subset as well as the feature weights. A control strategy that tells us when to reconsider the prototypes, the feature subset and the feature weight has to be developed in further work.

7 Results

We calculated the classification accuracy for the simple nearest-neighbor classifier and the k-nearest neighbor with $k = 3$ based on the leave-one-out cross-validation. The results for the different data bases are shown in Table 3. The experiments differ in the feature-weight setting (all weights are one or learnt based on the procedure described in Sect. 5.3) and in the setting for the number of neighbors k, see Table 4. For the experiment P1K1 (feature weights are for all features one, $k = 1$), the best results could be achieved for the data base DB_1. This is not surprising, since this data base has enough prototypes for each class and a low number of classes. The result for the k-nearest-neighbor classifier was slightly better for the data base DB_1. A poor result was achieved for the data base DB_2, i.e. the database with the highest number of classes and the smallest number of prototypes per class. Contrary to our expectation, the result for the database DB_4 was the poorest, although the number of samples in each class seems to be moderate.

It is clear that we cannot achieve a significant improvement of the accuracy for the database DB_2 when applying the k-NN classifier, since there are often only two prototypes of the same class.

Table 3 Classification accuracy for the different databases

DataBase	TestID			
	P1K1	P1K3	P3k1	P3K3
DB_1	80.12	80.20	80.71	78.04
DB_2	48.31	45.69	50.94	45.32
DB_3	69.44	69.44	69.87	69.78
DB_4	44.22	44.22	45.99	56.28

Table 4 Experimental set-up

Experiment	Feature Weights	K-NN
P1K1	1	1
P1K3	1	3
P3K1	learnt	1
P3K3	learnt	3

If we use the feature-subset selection and feature-weighting procedure described in Sect. 5.3, we can improve the classification accuracy for DB_1,DB_2,DB_3, and DB_4 in the case of the simple nearest-neighbor classifier and for DB_4 for the k-nearest-neighbor classifier. The highest improvement (5%) can be achieved for the database DB_2. In [12] we have shown that different feature-subset selection strategies select different feature subsets from a set of features and that there is often only a small intersection between the selected subsets. Therefore, we were not concerned about the kind of selected features, we rather looked for the improvement in accuracy. We could show by our results that feature-subset selection is an essential feature of a catalogue-based classifier.

In an early, experiment [17] we achieved an accuracy of 75% for six classes and based on decision trees. For each of the six classes we had 53 images. If we set our actual results into this context, we can conclude that every result that is higher than 60% accuracy is a good result.

If we apply Chang's prototype-selection strategy to our data base DB_1, we can reduce the number of prototypes by 75.5% for the simple nearest-neighbor approach, while preserving the accuracy, see Table 5. It is interesting to note that for this data base only 15 samples are adopted; whereas 70 samples are generated. There is enough redundancy among the samples.

In the case of the database DB_2, that is the database with the largest number of classes, but only a few samples per class, the reduction of samples is only 21.32%. The majority of samples are adopted, whereas only a few new prototypes are generated. The prototype generation is mostly performed for the classes with more than two samples. As a result we have to conclude that only two samples in some of the classes do not cover well the solution space for these classes.

From this we can conclude that the new function needed is prototype enrichment for that case where only a small number of prototypes is available. A method of Bayesian Case Reconstruction was developed by Hennessy et al. [21] to broaden the coverage of a case library by sampling and recombining pieces of existing cases to construct a large set of "plausible" cases. However, this method is not applicable in our case. We need to develop a method for our problem.

Table 5 Reduction of samples after application of Chang's algorithm

Name	Number of classes	Number of cases	Generated	Overtaken	Total number	Number of loops	Reduction in %
DB_1	4	347	70	15	85	192	75.50
DB_2	26	272	43	171	214	15	21.32
DB_3	10	156	33	57	90	33	42.31
DB_4	12	203	39	110	149	15	26.60

We observe the same situation for the database DB_4 as for database DB_2, whereas for the database DB_3 we see a reduction by 42.31%. That implies that the database DB_4 has much class overlap in the samples and is not a true gold standard. It is the only database that comes from a diagnostic lab and not from a manufacturer.

Like in the work of McKenna and Smyth [10] we need to develop a case-authoring system that allows us to study the competence of prototypes and the class-specific accuracy of a prototype with respect to other prototypes. This is left to further work.

8 Conclusions

We have studied the use of prototypical images for the development of an automatic image-interpretation system based on the nearest-neighbor classification. The results are promising. They show that it is possible to build an image-interpretation system with sufficient classification accuracy based on a small number of prototypical images. Feature-subset selection and feature weighting are essential functions of such a system. These functions can significantly improve the classification accuracy, especially in the case of small samples. Prototype generation has been applied in the sense that samples are generalized to one prototype. This function can be used in order to judge the quality of the prototypical case-base. The higher the reduction rate, the higher the redundancy of the prototypes. The results for the different prototypical databases used for our study show that two new functions are necessary for the proposed prototype classification-system. One function is prototype enrichment for that case where only a small number of prototypes is available. We also need to develop a case-authoring system that allows us to study the competence of prototypes and the class-specific accuracy of a prototype with respect to other prototypes. This is left to further work.

Acknowledgements This project is sponsored by the European Commission within the project "Multimedia Understanding through Semantics, Computation, and Learning" No. 507752. The author wish to thank the three referees for their suggestions that helped to improve the final version of this paper.

References

1. Smith MM, Smith TL (2002) A pocket atlas of normal CT anatomy of the head and brain, 2 edn. Radiology pocket atlas series. Williams & Wilkins, Baltimore
2. Samson RA, Hoekstra ES, Frisvad JC, Filtenborg O (eds) (2002) Introduction to food- and ariborne fungi, 6th edn. Centraalbureau voor Schimmelcultures, Utrecht
3. Sachs-Hombach Kl (2002) Bildbegriff und Bildwissenschaft. In: Gerhardus D, Rompza S (eds) Kunst–Gestaltung–Design, Heft 8. St. Johann, Saarbrücken, pp 1–38
4. Schmidt R, Gierl L (2001) Temporal abstractions and case-based reasoning for medical course data: two prognostic applications. In: Perner P (ed) Machine learning and data mining in pattern recognition, MLDM2001. Lecture notes in artificial intelligence, vol 2123. Springer, Berlin, pp 23–34
5. Nilsson M, Funk P (2004) A case-based classification of respiratory sinus arrhythmia. In: Funk P, Gonzalez PA (eds) Advances in case-based reasoning, ECCBR 2004. Lecture notes in artificial intelligence, vol 3155. Springer, Berlin, pp 673–685
6. Belazzi R, Montani St, Portinale L (1998) Retrieval in a prototype-based case-library: a case study in diabetes therapy revision. In: Smyth B, Cunningham P (eds) Advances in case-based reasoning, ECCBR 1998. Lecture notes in artificial intelligence, vol 1488. Springer, Berlin, pp 64–75
7. Aha DW, Kibler D, Albert MK (1991) Instance-based learning algorithm. Mach Learn 6(1):37–66
8. Bichindaritz I, Kansu E, Sullivan KM (1998) Case-based reasoning in CARE-PARTNER: gathering evidence for evidence-based medical practice. In: Smyth B, Cunningham P (eds) Advances in case-based reasoning, ECCBR 1998. Lecture notes in artificial intelligence, vol 1488. Springer, Berlin, pp 334–345
9. Delany SJ, Cunningham P (2004) An analysis of case-base editing in a spam filtering system. In: Funk P, Gonzalez-Calero PA (eds) Advances in case-based reasoning, ECCBR 2004. Lecture notes in artificial intelligence, vol 3155. Springer, Berlin, pp 128–141
10. McKenna E, Smyth B (2000) A competence-guided case-base editing technique. In: Blanzeri E, Portinale L (eds) Advances in case-based reasoning, EWCBR 2000. Lecture notes in computer science, vol 1898. Springer, Berlin, pp 186–197
11. Ontañón S, Plaza E (2004) Justification-based selection of training examples for case base reduction. ECML 2004:310–321
12. Perner P (2001) Improving the accuracy of decision tree induction by feature pre-selection. Appl Artif Intell 15(8):747–760
13. Wettschereck D, Aha DW (1995) Weighting features. In: Veloso MM, Aamodt A (eds) Case-based reasoning research and development. Lecture notes in computer science, vol 1010. Springer, Berlin, pp 347–358
14. Perner P (2000) CBR-based ultra sonic image interpretation. In: Blanzieri E, Portinale L (eds) Advances in case-based reasoning. Lecture notes in artificial intelligence, vol 1898. Springer, Berlin, pp 479–490
15. Bradwell IAR, Stokes RP, Johnson GD (1995) Atlas of HEp-2 patterns. AR Bradwell, ISBN 0704416220
16. Eysenck MW, Keane MT (2000) Cognitive psychology: a student's handbook, 4th edn. Psychology, Hove
17. Perner P, Perner H, Müller B (2002) Mining knowledge for Hep-2 cell image classification. Artif Intell Med 26:161–173
18. Otsu N (1979) A threshold selection method from gray-level histograms. IEEE Trans Syst Man Cybern 9:38–52
19. Chang C-L (1974) Finding prototypes for nearest neighbor classifiers. IEEE Trans Comput C 23(11):1179–1184
20. Perner P (2002) Data mining on multimedia data, Lecture notes in computer science, vol 2558. Springer, Berlin
21. Hennessy DN, Buchanan BG, Rosenberg JM (2002) Bayesian case reconstruction. In: Craw S, Preece A (eds) Advances in case-based reasoning, ECCBR 2002. Lecture notes in artificial intelligence, vol 2416. Springer, Berlin, pp 148–158
22. Kohavi R, Sommerfield D (1995) Feature subset selection using the wrapper method: overfitting and dynamic search space topology. In: Proceedings of the first international conference on knowledge discovery and data mining
23. Perner P, Zscherpel U, Jacobsen C (2001) A comparison between neural networks and decision trees based on data from industrial radiographic testing. Pattern Recognit Lett 22:47–54