A Comparison of Classifiers for Prescreening of Honeybee Brood Cells

Uwe Knauer[1], Fred Zautke[2], Kaspar Bienefeld[2], and Beate Meffert[1]

[1] Humboldt-Universität zu Berlin, Institut für Informatik
10099 Berlin, Germany
{knauer,meffert}@informatik.hu-berlin.de

[2] Länderinstitut für Bienenkunde, Hohen-Neuendorf, Germany
{fred.zautke,kaspar.bienefeld}@cms.hu-berlin.de

Abstract. We report on an image classification task originated from the video observation of beehives. Biologists desire to have an automatic support to identify so called hygienic bees. For this it is important to know which brood cells are in a stadium of initial opening. To find these cells a prescreening process is necessary which distinguishes between three types of cells. To solve this decision problem a number of classification techniques were evaluated. ROC analysis for the given problem shows that the SVM classifier with RBF kernel outperforms linear discrimance analysis, decision trees, boosted classifiers, and other kernel functions.

1 Introduction

One of the biggest threats for the native honeybee *Apis melifera* is the mite *Varroa destructor* [6]. The varroa mites are external honeybee parasites and the infestation of a colony is a serious problem because it is not possible to cure an afflicted colony without the risk of side effects[1]. One of the most promising approaches to block the mites is the rearing of resistant bees. Therefore, current research in the field of apiculture focuses on the genetic selection of hygienic bees [1]. Hygienic behaviour is characterized by three components: finding dead or damaged brood quickly, uncapping these brood cells, and removing dead or damaged brood from the cells. The selection of hygienic bees requires a time consuming observation of the combs. Processing all the material that is typically recorded for a period of one week (24 hours a day) requires at least twice the time for analysis by a human expert. Therefore, it would be helpful to develop algorithms for an automated observation of the combs.
Previously published work has focused on the detection of completely uncapped cells [7]. But if openings can be detected as early as possible (see Fig. 1) the identification of hygienic bees will be improved.
If a way is found to decide whether the surface of a cell is visible or not, the current image can be compared with previously recorded images of the same cell

[1] The treatment with acaricide agents may lead to unwanted residues in wax and honey.

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

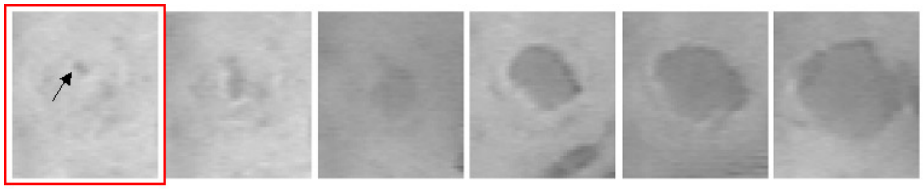

Fig. 1. Different states of uncapping a brood cell. The objectives are (1) to obtain such images in the presence of bees and (2) an early detection which corresponds to the left image of the sequence.

to detect changes. However, this is not as easy as it may sound, because the bees cast shadows on the cells and body parts of the bees look similar to the small changes we are looking for. Hence, early localisation of openings in these images is still under investigation and will be published later. In this paper we focus on the problem to select only relevant images in the presence of bees.
The paper is structured as follows. First, the reader is introduced to the experimental setting. Next, the different classifiers and results of classification are presented. Finally, the integration of these classifiers into a computer vision system is described.

2 Experimental setup

The experimental setting is shown in Fig. 2. The left picture shows a top view of the test beehive. It consists of a camera (1), near infrared illumination (2) and 2000 bees (3). The second picture gives an overview of the system. The recordings are stored on a digital Panasonic HDD video recorder (4). A monitor (5) allows the tuning and control of the camera system. The computer system for image analysis (7) is connected via a USB video capture device (6). To ensure that the experimental conditions for the bee population do not differ too much from the normal environment, the container is placed outside.
Image processing is done with the Open Computer Vision Library. For statistical analysis SAS, Matlab, Spider [10], and SVMlight [5] are used.

Fig. 2. Experimental setup from left to right: test beehive, system chart, still image

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

Fig. 3. Three states of a brood cell: *occluded*, *visible and closed*, and *visible and uncapped*

3 Training and test data

Fig. 3 shows the typical states of a cell we have to discriminate. A cell can be *occluded by a bee*, *visible and closed*, or *visible and uncapped*.
For supervised learning and evaluation of classification rules we need a sample of images. Tab. 1 summarizes the number of images corresponding to the three classes of cell states.

class ID	cell state	sample size
C_1	occluded	3600
C_2	visible and closed	5800
C_3	visible and uncapped	800
		10200

Table 1. Image sets

Each sample contains images of different broods cells from different experiments under varying lighting conditions. Hence, it can be assumed that the sample is suitable to find a general rule to distinguish between the three classes. The average size of an image is 40x40 pixels.

4 Design of classifiers

We have tested three techniques for supervised learning in the context of the given 3-class problem. In addition, classification was done in different feature spaces. The methods we applied were minimum distance classifiers (MDC), decision trees (DT), and support vector machines (SVM). For the MDC and decision trees we also implemented adaptive boosting (AdaBoost) to investigate possible improvements in the overall accuracy of the classifiers.

4.1 Minimum distance classifier

To classify unknown feature vectors a distance measure $d_{C_i}(\mathbf{x})$ between any of the classes C_i and the vector $\mathbf{x}$ has to be calculated. The label l of the class for which the distance is minimal is assigned to the vector:

$$l(\mathbf{x}) = \mathbf{argmin_i}(d_{C_i}(\mathbf{x})).$$

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

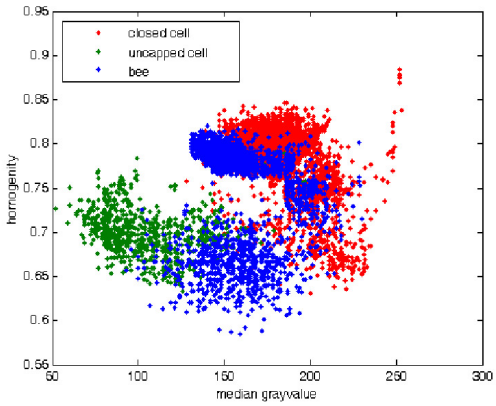

Fig. 4. Two-dimensional space spanned by the features median and homogeneity

Here, the Mahalanobis distance between the mean vector m_i of each class and the vector $\mathbf{x}$ is used for classification. It is defined as:

$$d_{C_i}(\mathbf{x}) = \sqrt{(\mathbf{x} - \mathbf{m_i})^{\mathbf{T}} \cdot \mathbf{G_i^{-1}} \cdot (\mathbf{x} - \mathbf{m_i})},$$

where the $\mathbf{m}_i$ are the mean vectors and $\mathbf{G}_i$ the covariance matrices of the classes C_i. Given a labeled set of training vectors these parameters can easily be derived.

Instead of solving the 3-class problem directly, it is split into two 2-class problems. It is important to note, that this choice is related to the design of the complete image processing system. Detection of C_3 is handled independently, because open cells serve as landmarks in the setup of the system. Additionally, it is not of interest to observe cells once they have been opened.
Therefore, one MDC is applied to distinguish between C_3 and the other classes. Another MDC is applied to discrimate C_1 and C_2.

Naive approach It is an obvious property that cells of classes C_2 and C_3 have a homogeneous appearance compared to most of the occluded cells (C_1; see Fig. 3) so that a measure of homogeneity can be used for classification. To further distinguish between C_2 and C_3 the luminance can be measured.
In Fig. 4 the median and a homogeneity value of all images are plotted for the three classes. The three classes have distinct centers of gravity, but the interclass distance is small. Hence, the classes are overlapping in some regions of the feature space, making classification difficult.

Increasing dimensionality As can be seen in Fig. 4, compactness and interclass distances are limited. To overcome these problems more features can be

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

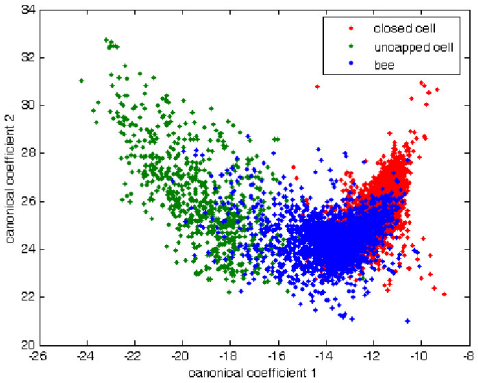

Fig. 5. Two-dimensional space spanned by the canonical features f_{d_1} and f_{d_2}

derived from the images. Tab. 2 lists our choice of 18 features which can be derived from single channel images. All features are calculated for the entire image of a single cell.

minimum	maximum	span	dynamic
mean	effectivity	median	standard deviation
variance	kurtosis	skewness	normalized kurtosis
normalized skewness	entropy	anisotropy	homogeneity
contrast	coocurance matrix entropy		

Table 2. Used statistical features of single channel images

Linear discriminant analysis and canonical features The objective of the linear discriminant analysis (LDA) is to find one or more weight vectors ω_j such that the values of the resulting scalar products $f_{d_j}(\mathbf{x}) = \omega_\mathbf{j} \cdot \mathbf{x}$ improve the separation into classes.
For a feature vector $\mathbf{x}$ the values of the functions f_{d_j} are treated as new -so called canonical- features. Typically, for an N-class problem it is sufficient to calculate N-1 linear combinations $f_{d1}, \ldots, f_{dN-1}$. Fig. 5 illustrates the result of the feature transform. Note, that the projection is based on the complete set of the (weighted) 18 features.
Compared to the two-dimensional feature space shown in Fig. 4, the compactness of the classes increases in canonical feature space while the interclass distance does not.

Adaptive Boosting Adaptive Boosting is a technique for improving the classification performance of weak learning algorithms [3]. We applied this technique

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

to learn, to which extent the performance of the MDC approach can be increased by boosting. The method of Schapire and Freund was implemented as described in [4].

Comparison Tab. 3 summarizes the results of 10-fold cross-validation of different MDCs which have been trained to distinguish between C_3 and the other two classes.

Variant	C_3	$C_1 + C_2$	total
Naive	95.5 %	95.52 %	95.52 %
18-dim	97.62 %	97.96 %	97.93 %
canonical	**99.75 %**	96.3 %	96.57 %
boosted naive	93 %	98.61 %	98.17 %
boosted 18-dim	92.13 %	**99.41 %**	**98.84 %**
boosted canonical	99 %	97.51 %	97.63 %

Table 3. Results of MDC for separating C_3 (10-fold cross-validation)

The comparison shows that the performance of the classifiers increases with the dimension of the feature vectors. In addition, performance can be slightly boosted (3 % in the naive approach). This is not very surprising since the boosting procedure prefers the class $C_1 + C_2$ because the training set is larger compared to class C_3.
Tab. 4 shows the results for separating the classes C_1 and C_2. 10-fold cross-validation of the reference data set (see Tab. 1, C_3 excluded) is used for the evaluation of the classifiers.

Variant	C_1	C_2	total
Naive	85.5 %	80.19 %	82.22 %
18-dim	90.92 %	83.86 %	86.56 %
canonical	89.75 %	82.76 %	85.47 %
boosted naive	86.11 %	80.40 %	82.59 %
boosted 18-dim	**91.61 %**	**88.19 %**	**89.50 %**
boosted canonical	89.81 %	82.34 %	85.20 %

Table 4. Results of MDC for separating C_1 and C_2 (10-fold cross-validation)

Similar effects can be observed. Including more features in the analysis allows a better separation of the classes. With boosting, the overall accuracy can be increased to nearly 90 % (estimated with cross-validation). Using the two most relevant canonical features can improve classification compared to the naive approach, but performance can not be further increased by boosting.

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

For decision tree learning and support vector machines only the results on separating C_1 and C_2 are presented, since it is the more difficult problem.

4.2 Decision tree learning

The algorithms C4.5 and J4.8 were applied to generate decision trees [9, 11]. For evaluation of the classifiers 5-fold cross-validation is used. Additionally, the effects of reduced error pruning [8] and adaptive boosting on the classification performance are tested for the J4.8 algorithm which is an improved Java implementation of C4.5. The tree generated by the C4.5 algorithm was pruned by truncating all subtrees which classify less than 5 % of the examples in the training set. The remaining tree with only 20 test nodes already shows better performance than unboosted MDCs.
Boosted trees perform best and performance increases with the number of the trees in the ensemble. A random forest [2] which is another popular ensemble technique was trained with the same number of trees as used for boosting. Its classification performance keeps up with the results obtained with boosted trees. Tab. 5 summarizes the classification results.

algorithm	pruning	test nodes	C_1	C_2	total
C4.5	none	556	86.84 %	90.56 %	89.14 %
C4.5	truncation, 5 %	20	84.59 %	90.05 %	87.94 %
J4.8	none	153	85.78 %	92.50 %	89.92 %
J4.8	reduced error pruning	137	86.48 %	92.55 %	90.22 %
boosted J4.8	reduced error pruning	5 trees	87.57 %	93.45 %	91.18 %
boosted J4.8	reduced error pruning	10 trees	**88.71 %**	93.93 %	**91.92 %**
Random Forest		5 trees	87.99 %	92.90 %	91.01 %
Random Forest		10 trees	87.29 %	**94.74 %**	91.88 %

Table 5. Classification rates of decision trees and ensemble methods for separating C_1 and C_2 (5-fold cross-validation)

4.3 Support Vector Machines

Support vector machines have been successfully applied to many large-scale classification problems [5]. Classification performance of linear, radial basis functions and polynomial kernels has been tested for the problem of separating C_1 and C_2.
The parameters of all kernel functions have been systematically varied to increase the overall accuracy. Tab. 6 summarizes the results for the 18-dimensional feature vectors.
The operating point was selected to achieve the best overall classification rate. Using the same features the SVM performs comparable or only slightly better than the other classification techniques.

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

kernel	γ	scaling	degree	total
linear				88.62 %
rbf	0.005			**91.59 %**
polynomial		0.1	3	88.89 %

Table 6. Results of 10-fold cross-validation for different kernels on 18-dimensional data

For the next experiment the images were downscaled to 11x11 pixels. Then, they are treated as 121-dimensional vectors and normalized to have unit length. Again, the different kernel functions are tested. The results are given in Tab. 7.

kernel	γ	scaling	degree	total
linear				84.46 %
rbf	1.01			**94.30 %**
polynomial		0.16	4	85.04 %

Table 7. Results of 10-fold cross-validation for different kernels on 121-dimensional data

ROC analysis is used to compare the classification performance. Receiver Operator Characteristics curves plot sensitivity (Y-axis, true positives) versus 1 minus specifity (X-axis, false positives). The area under the curve (AUC) is a common measure to assess the quality of a classifier. A perfect classifier would have an AUC of 1.0. Fig. 6 shows ROC curves for the different kernel functions.

The RBF kernel clearly outperforms the other kernels. Surprisingly, the three kernel functions have the same performance in identifying instances of class C_1 at the operating point. Hence, the difference in the classification rate is mainly related to the specifity of the classifiers. This is in accordance to the findings for the MDC, since the boosting step mainly increases the performance of classification of C_2. For completeness, boosted decision trees were trained with the new set of features. SVM with RBF kernel still performs slightly better.

Given the task to detect visible cells, an instance of class C_1 classified as C_2 would be a false positive. In the reverse case, an instance of C_2 classified as C_1 is a false negative. The consequence of a higher false positive rate is that more images showing bees are included in subsequent steps of the analysis. In this case the system will report more events (openings of cells). The time for the complete analysis increases, because the involved bees are still searched by a human observer. On the other hand, the consequence of more false negatives is that images which potentially include an opening at an early state will not be reported. In practise the operating point is chosen such that the number of false negatives is reduced because hygienic behaviour can be observed very rarely (indicated by a cross in Fig. 6). The other operating points in Fig. 6 mark the maximum overall classification rates of the different classifiers (indicated by dots).

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

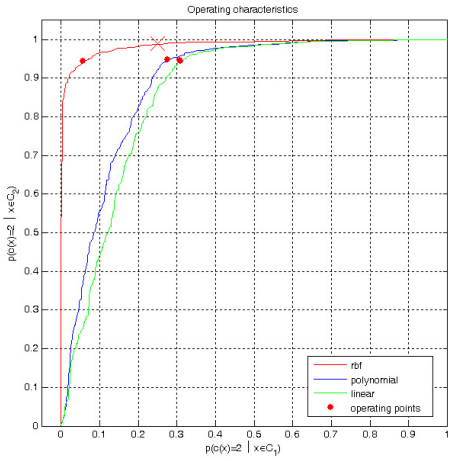

Fig. 6. ROC curves for different kernel functions, typical operating point with low false negative rate (cross), operating points for best overall accuracy (dots)

5 Embedding of the classifiers in the computer vision system

In this section the reader is introduced to the computer vision system which was designed for monitoring the honeybee comb. It consists of three modules:

1. the preparation module,
2. the observation module (naive MDC for quick setup and real-time monitoring of 110 cells), and
3. the postprocessing module (SVM RBF for reliable classification).

The classifiers for the prescreening of brood cells are part of the observation and the postprocessing modules.
The purpose of the preparation module is to provide prior knowledge about position, size, and type of each brood cell. The information is saved as an XML configuration file. In Fig. 7 the editor window (1), a highlighted brood cell (2), and the generated XML file (3) are shown.
The purpose of the observation module is to generate a report of the experiment. For each brood cell the report provides a series of images of the cell surface and the corresponding recording times. The report allows the quick assessment of possible changes on the surface. Basically, the observation module has to handle the following tasks:

1. prescreening of the brood cells,
2. generation of an HTML based report to present the selected images, and
3. tracking of the combs surface which shifts because of temperature variations.

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

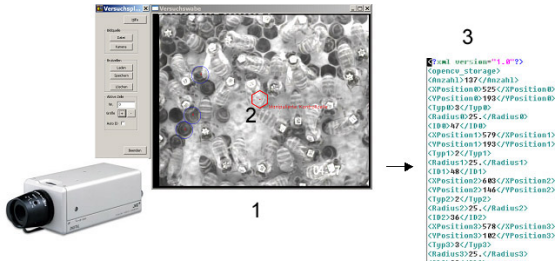

Fig. 7. User interface of the preparation module for acquisition of prior information.

As discussed in the previous sections, the prescreening requires an appropriate classifier. So far, we have focused only on the performance of the classifiers in terms of the classification rates. In addition to this factor the computational complexity and the adaptivity to changed lighting conditions are important criteria for on-line processing. Therefore, a modification of the naive MDC is used for real-time monitoring. This approach is the fastest because only two features have to be calculated for each cell. Additionally, the parameters of the naive MDC are defined interactively once the experiment has started. This allows to adapt the system to changed experimental settings without collecting a new sample of images for training.

Fig. 8 shows the user interface of the observation module. The system stores a single image of each cell that is classified as C_2 after a fixed time interval, which currently is set to 1 minute. The time interval is used to find an image for which the Mahalanobis distance to the corresponding class mean vector is minimal. The HTML report pages are updated to reflect latest changes to the combs surface.

The SVM classifier is applied in a postprocessing step to compensate the lack of classification performance due to the use of the naive MDC approach. Apart from this, the postprocessing module is intented to locate initial openings in the prescreened images. The postprocessing produces a sparse report which highlights potential events of interest.

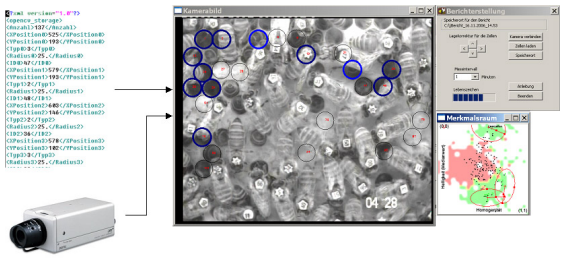

Fig. 8. User interface of the observation module which uses a naive MDC.

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de

The system has successfully undergone a field test. To evaluate the quality of the reports, the analysis of the field test experiment was done twice. First, the expert followed the established procedure of analysing the video recordings. Second, the software generated reports were used to guide the search for hygienic bees. Using the reports, the effort was reduced approximately by a factor of two.
The generated sparse report was checked, whether it still contains the relevant images of initially opened cells or not. The described prescreening approach seems to be an appropriate starting point to locate initial openings, because nearly all of the events have been successfully recorded.

6 Summary

In this paper we reported on the performance of different classifiers for the prescreening of brood cells. This is an important task because the exclusion of uncapped and occluded cells from further processing steps accelerates the analysis and prevents aftereffects.
The problem of finding openings automatically, reliably, and as early as possible is still under investigation.

References

1. K. Bienefeld and G. Arnold. Studies on the genetic determination of uncapping of varroa-infested brood cells. In *First European Conference of Apidology, EurBee*, pages 103–104, 2004.
2. L. Breiman. Random forests. *Machine Learning*, 45:5–32, 10 2001.
3. R. O. Duda. *Pattern Classification*. John Wiley & sons, New York, 2. edition, 2001.
4. Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *European Conference on Computational Learning Theory*, pages 23–37, 1995.
5. T. Joachims. Making large-Scale SVM Learning Practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods*. MIT Press, 1999.
6. D. D. Jong. Mites: Varroa and other parasites of brood. In R. A. Morse and R. Nowogrodski, editors, *Honeybee Pests*, pages 200–218. Cornell University Press, Ithaca, 1990.
7. U. Knauer, M. Himmelsbach, F. Winkler, F. Zautke, K. Bienefeld, and B. Meffert. Application of an adaptive background model for monitoring honeybees. In *IASTED International Conference on Visualization, Imaging & Image Processing*, pages 46–50, 2005.
8. J. R. Quinlan. Simplifying decision trees. *International Journal of Man-Machine studies*, 27(3):221–248, 1987.
9. J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufman, 1993.
10. J. Weston, A. Elisseeff, G. BakIr, and F. Sinz. Spider: Machine Learning in Matlab. http://www.kyb.tuebingen.mpg.de/bs/people/spider/main.html.
11. I. H. Witten and B. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufman, 1999.

Proceedings of the 5th International Conference on Computer Vision Systems (ICVS 2007)
Published in 2007 by Applied Computer Science Group, Bielefeld University, Germany, ISBN 978-3-00-020933-8
This document and other contributions archived and available at: http://biecoll.ub.uni-bielefeld.de