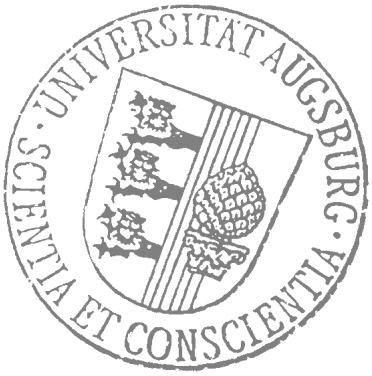
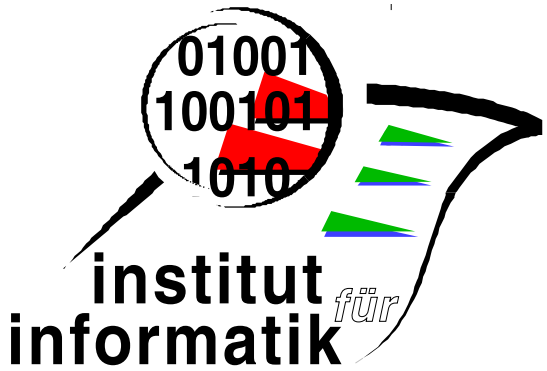
Universität Augsburg



Active Improvement of Hierarchical Object Features under Budget Constraints

Nicolas Cebron

Report 2011-01 Februar 2011



Institut für Informatik
D-86135 Augsburg

Active Improvement of Hierarchical Object Features under Budget Constraints

Nicolas Cebron

***Abstract*—When we think of an object in a supervised learning setting, we usually perceive it as a collection of fixed attribute values. Although this setting may be suited well for many classification tasks, we propose a new object representation and therewith a new challenge in data mining: an object is no longer described by one set of attributes but is represented in a hierarchy of attribute sets in different levels of quality. Obtaining a more detailed representation of an object comes with a cost. This raises the interesting question of which objects we want to enhance under a given budget and cost model. This new setting is very useful whenever resources like computing power, memory or time are limited. We propose a new Active Adaptive Algorithm (AAA) to improve objects in an iterative fashion. We demonstrate how to create a hierarchical object representation and prove the effectiveness of our new selection algorithm on these datasets.**

***Keywords*-Pattern classification; Active vision;**

I. Introduction

The goal of supervised learning is to deduce a function from examples in a dataset that maps input objects and desired outputs. By using a set of labeled training examples, we can train a classifier that can be used to predict the target variable (which may be continuous or nominal) for unseen test data. To achieve this, the learner has to generalize from the presented data to unseen situations. In the traditional machine learning scenario, our objects of interest are described by a given number of attributes values (the so-called feature vectors).

In many situations, a feature vector representation may be the best choice to describe the objects at hand. However, in many circumstances a different object representation is useful, e.g., in the domain of graph mining or structure mining. In this paper, we want to draw the attention to a new representation of objects with multiple views[1]. We describe an object in a hierarchy of views of ascending quality. We start to learn a classifier on the lowest level where all objects are initially given. We can decide to enhance a particular example by obtaining a representation on a higher level. This ”upgrade” comes with a cost that is associated with the corresponding view level.

Figure 1 shows an example from the object recognition domain. We can train a classifier on the attributes of a digit on the lowest level (Level 1) to separate different classes from each other. This might work well if we want to discriminate digits that have a very different appearance,

[1]Please note that this representation differs from Multi-View Learning [1] where objects are described in multiple views and each view contributes to the classification.

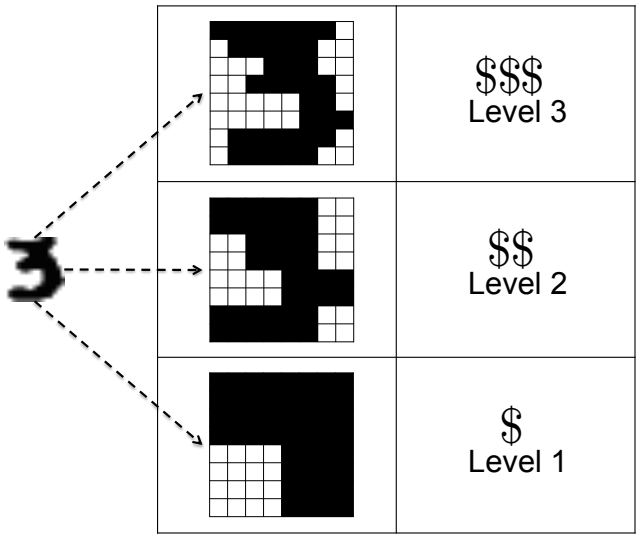


Figure 1. Digit object described in different levels of quality.

e.g., ’0’ and ’3’. However, to differentiate ’3’ from ’8’ (which may have a very similar appearance) we might need to enhance some examples to Level 2 or even Level 3 to reach a classification decision. The underlying idea is that we do not need to enhance all examples to build a good classification model. A subset of carefully selected examples may be sufficient to build a model that is accurate and generalizes well.

This setting is very useful whenever we have limited computer resources or time-constraints. We might not have enough memory to fit all objects into memory or we don’t have the time resources to compute the features for all objects on the highest quality level. For example, consider the domain of robotics: we might have a robot with a low-level camera that records its environment in VGA quality and a high-level camera that has a resolution of several mega-pixels. Calculating all the features on the mega-pixel level is impossible because the computing resources are limited. We aim to handle most of the examples on the low level and to improve only a few selected examples.

We do not need to stick to the vision domain to generate a hierarchy of object representations (although computing features from an image pyramid representation [2] is straightforward). In music or speech recognition, we can sample at different levels to generate a representation at a finer level. The same is true for 3D objects, e.g., point clouds

or objects in a CAD database.

Two points in this setting need careful consideration: First, we need to have the same features at each level. This means that we always compute the same features, just on different representations of the object. Second - depending on the type of features that are computed - there is usually a limit as to when a low representation is sufficient or when obtaining a finer representation of the object still increases the accuracy of the classifier. We will demonstrate how to create the hierarchical representations of objects in the experimental section.

When it comes to selecting interesting examples, there is a related concept called Active Learning [3]. It deals with the issue that it is common for many real world classification tasks to have a large pool of unlabeled samples available. As the cost of generating a label for a sample is high (because it has to be determined by a human expert), Active Learning enables a learner to pose specific queries that are chosen from an unlabeled dataset. These queries are then answered by a noiseless oracle in an iterative fashion and added to the set of labeled training examples. The goal is to obtain a precise model with few labeling iterations by picking those samples that can "help" to improve the classifier instead of picking random samples. The concept of Active Learning is very similar to the human form of learning, whereby problem domains are examined in an active manner.

Contrary to classic Active Learning, we do not try to select examples that will be labeled by a human expert. We assume that all labels for the training examples are given. Instead we try to select examples that will be upgraded to a higher representation to obtain a better classification accuracy.

In this paper, we employ this scheme during training time, but the ideas developed here can also be used during testing time to optimize the application.

We will revise related work in Active Learning in Section II and Support Vector Machine (SVM) classification in Section III and introduce our new Adaptive Active Algorithm (AAA) for object improvement in Section IV. We will demonstrate the advantages of this new method with several experiments in Section V before giving conclusions in Section VI.

II. Related Work

Although most of the works in Active Learning deal with selecting an example for labeling, the ideas are very related to our work. An Active Learner is described by its underlying classifier and its query function. The classifier is trained on the labeled data. The query function makes a decision based on the current model as to which samples from the unlabeled data pool should be chosen for labeling. We can categorize the existing Active Learning approaches by their selection strategy:

- **Optimization of a target function**: Based on the minimization of the expected error function (or maximization of a likelihood function) examples can be selected by their contribution to this function. Popular approaches in this field are the works of [4], [5], [6], [7]. From a theoretical point of view the explicit definition of a target function that should be minimized makes it easy to analyze the selection strategy. However, these approaches make several assumptions (e.g., that a stable model built with randomly chosen examples already exists or that the learner does not have a bias [6]); therefore the outcome of the selection strategy depends on how these assumptions apply.
- **Reduction of version space**: The goal of this approach is to reduce the version space with a selected sample as much as possible. One of the most popular approaches is the Query by Committee algorithm [8], which uses a committee of diverse but consistent hypotheses and queries examples for which the disagreement is maximal. Another approach imitates the most general and most specific hypothesis with a neural network and queries examples at the region of uncertainty between those two hypotheses [3]. In the work of [9] the parameter space of a Support Vector Machine (SVM) is related to the version space in order to derive several strategies to query new examples.
- **Uncertainty sampling**: This heuristic approach focuses on selecting examples at the classification boundary. The most popular approaches use an SVM and query examples at the decision hyperplane in the kernel induced space [10], [11] similar to one of the version space reduction approaches described by [9]. Uncertainty sampling is prone to selecting outliers. Like all other approaches it relies on a stable classification model that has been initialized with some randomly chosen examples.

In more recent works in the field of Active Learning one can observe the trends toward making these schemes more robust by using meta techniques to balance strategies for exploration and exploitation [12], [13], [14] and focusing on the theoretical aspects and benefits [15], [16] of Active Learning. A few works have dealt with the issue of varying labeling costs in Active Learning. The authors of [17] propose a decision-theoretic approach that takes into account labeling and misclassification costs. In the work of [18], annotation costs are variable and not known. A regression cost-model is used to predict the real, unknown annotation cost based on meta features of the instances.

In the works of [19], [20], a new setting is introduced where the oracle provides features instead of labels. The feature values may have variable acquisition costs. This is very useful in some domains (e.g., business databases with customer information or medical databases with patient in-

formation) where active feature acquisition seeks to request more complete feature information. In [19], the missing feature values are imputed and then the ones about which the model has the least confidence are acquired. The authors of [20] take a decision-theoretic approach and acquire the feature values which are expected to maximize some utility function.

Although the two preceding works follow a similar idea, the key difference to our work lies in the representation of objects. They assume that objects are given incompletely in one feature space, whereas our representation spans multiple feature spaces. The advantage of our approach is that objects are fully described in each feature space; therefore we can skip the requirement of the other approaches that the induction algorithm must be able to infer or be able to treat missing values. As we shall see in the next Sections, this new object representation also leads to a new strategy for enhancing objects.

There is also the general idea of a cascade of classifiers in the vision domain [21]. Increasingly more complex classifiers are arranged in a cascade to quickly discard background regions while spending more computation time on promising object-like regions. This method can be viewed as an object specific focus-of-attention mechanism which is similar to our approach. However, our setting is different in the way that we deal with one classifier and a hierarchical object structure.

III. Learning with SVM

In this Section, we introduce the underlying Support Vector Machine (SVM) classification algorithm before turning to the selection strategy. Given a set of labeled training data $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_m, y_m)$ where $\boldsymbol{x}_i \in \mathbb{R}^N$ and $y_i \in \{-1, +1\}$, a linear SVM [22] is defined in terms of its hyperplane

$$w \cdot \boldsymbol{x} + b = 0 \tag{1}$$

and its corresponding decision function

$$f(\boldsymbol{x}) = sgn(w \cdot \boldsymbol{x} + b) \tag{2}$$

for $w \in \mathbb{R}^N$ and $b \in \mathbb{R}$. Among all possible hyperplanes that separate the positive from the negative examples, the unique optimal hyperplane is the one for which the margin is maximized:

$$\max_{w,b}\{\min_{\boldsymbol{x}_i}\{||\boldsymbol{x} - \boldsymbol{x}_i|| : \boldsymbol{x} \in \mathbb{R}^N, w \cdot \boldsymbol{x} + b = 0\}\} \tag{3}$$

As the training data is not always separable, a soft margin classifier uses a misclassification cost C that is assigned to each misclassified example. Equation 3 is optimized by introducing Lagrange multipliers α_i and recasting the problem in terms of its Wolfe dual:

$$\begin{aligned} &\text{maximize: } L_D = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i \boldsymbol{x}_j \\ &\text{subject to: } 0 \leq \alpha_i \leq C, \text{and} \sum_i \alpha_i y_i = 0 \end{aligned} \tag{4}$$

All $\boldsymbol{x}_i$ for which the corresponding α_i are non-zero are the support vectors.

Figure 2 shows the SVM with labeled training examples (squares vs. circles); the support vectors are encircled. The

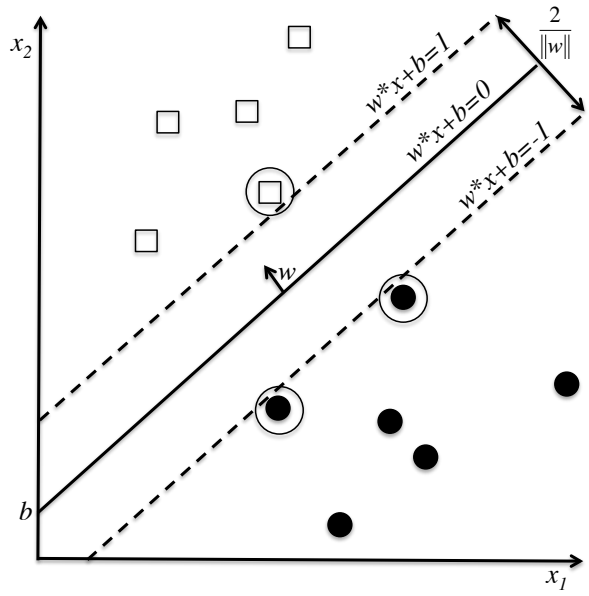


Figure 2. SVM classifier.

support vectors limit the position of the optimal hyperplane. The objects $\boldsymbol{x}_i$ for which $\alpha_i = C$ are the bound examples, which are incorrectly classified or are within the margin of the hyperplane. Support Vector Machines have been widely used in the field of text, music and image mining [23], [24], [25], where a large library of documents, music or images are available and we want to categorize this library with the help of the user. Support Vector Machines with Active Learning (based on uncertainty sampling) have been widely used for image retrieval problems [26], [25] and in the drug discovery process [27].

IV. Object Improvement Algorithm

In this Section, we introduce the Adaptive Active Algorithm to select objects that will be improved. We begin with a description of the hierarchical object representation and give an overview of the complete algorithm. Then we describe a naive and an active selection strategy for selecting examples under budget constraints.

A. Notation and Problem Statement

We assume that the objects $\boldsymbol{x}_i$ are described in views of ascending quality. For instance, we could compute the features from an image object at different levels of an image pyramid, where G_0 corresponds to the original image. G_0 is then low-pass filtered and subsampled by a factor of two to obtain the next pyramid level G_1 and so forth. There might of course be other ways to compute a representation of an object at different levels.

In this paper, we assume that we have j views of ascending quality $V_u, u = 1, \ldots, j$. The best level is denoted by V_j and the worst by V_1. We describe the object $\boldsymbol{x}$ in view j as $V_j(\boldsymbol{x})$. Note that all $\boldsymbol{x}_i$ in the training set are still in $\mathbb{R}^N$ (the number of attributes does not change).The training set at any point in time t consists of the objects - each object at a different view level: $D = \{(V_1(\boldsymbol{x}_1), y_1), (V_2(\boldsymbol{x}_2), y_2), \ldots, (V_m(\boldsymbol{x}_m), y_m)$. At $t = 0$, all objects are only given in the worst view V_1.

Figure 3 depicts the training set D at an arbitrary time point t. In this example, $\boldsymbol{x}_1$ has been enhanced to Level 2

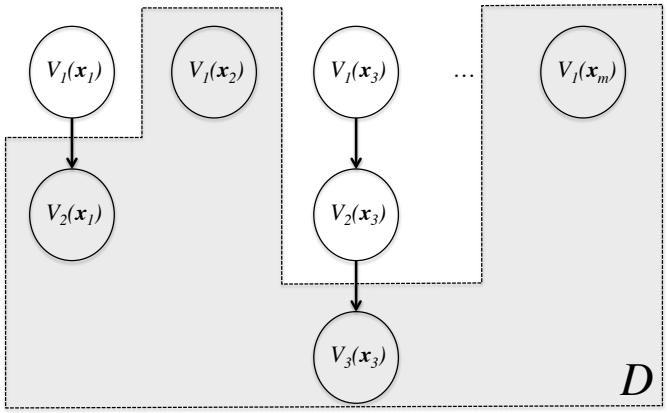


Figure 3. Training Set D at time point t.

and $\boldsymbol{x}_3$ to Level 3. We always use the best representation of each object in the current training set D.

Obtaining a better description of the object comes with a cost $c_u, u = 1, \ldots, j$ that is related to the view level. For simplicity, we use a cost model that is increasing by arithmetic progression. Notice that any cost model can be used in this setting. One may want to use a linear, quadratic or even cubic model, depending on the degree that underlies the increase of complexity in each view level. For example, the number of samples in a sound object may double on each view level, whereas the number of pixels in an image object quadruples in each view level of the image pyramid. In this work, each training point in the worst view V_1 has a cost of 1 unit and a cost of 2 units on the next level and so forth[2]. In the example of Figure 3, the total cost would be 2 ($\boldsymbol{x}_1$) + 1 ($\boldsymbol{x}_2$) + 3 ($\boldsymbol{x}_3$) + 1 ($\boldsymbol{x}_m$).

B. Algorithm Overview

In this Section, we describe the general framework in Algorithm 1. We are given a total budget B, the number of iterations n and the number of view levels j. Our initial training set consists of all examples on the lowest level. We start by computing an initial classifier on the given training set. We update the training set in an iterative fashion, similar to the classic Active Learning setting. Based on the current level of the objects and the current budget in each iteration, we choose a subset of examples. The quality for the chosen examples is enhanced and in the next iteration, a new SVM is computed. If we combine the preceding notion of objects at different view levels in the training set and the properties of the SVM classification in Figure 2, we can think of enhancing objects as 'moving around' data points in the shared feature space. We expect that these enhancements will lead to a better SVM model, as we now have more detailed information about some data points.

Algorithm 1 Object Improvement Algorithm

Require: Budget B, Number of iterations n, Number of view levels j

1: Current Budget $C = B/n$.
2: $D = \{(V_1(\boldsymbol{x}_1), y_1), (V_1(\boldsymbol{x}_2), y_2), \ldots, (V_1(\boldsymbol{x}_m), y_m)$
3: **while** Current iteration $\leq n$ **do**
4: Train SVM with D
5: T = Determine subset of total cost C
6: **for all** $t \in T$ **do**
7: $V_u(\boldsymbol{x}_t) = V_{u+1}(\boldsymbol{x}_t)$
8: **end for**
9: **end while**

In our experiments, we have used a fixed number of iterations. However, in practice one may want to use other stopping criteria. For example, one could restrict the number of iterations based on the given budget. Another possibility is to keep track of the training error error difference between two iterations and to define a threshold and stop when the error does not change significantly.

C. Random Sampling

Given a budget C in the current iteration, we have to choose a subset of objects T that satisfies the cost constraints. We start with a naive selection strategy called *Random*. This strategy proceeds as follows: we choose a random example and determine the cost for improving it. As long as the cost of improvement does not exceed the given budget C, we add this example to T. Random sampling is usually a very good base line as it makes sure that each object of the dataset has an equal chance of being selected.

[2]This unit could be monetary, memory or time unit.

D. Adaptive Active Sampling

Intuitively, for most of the data points we do not expect significant changes. Taking into account the nature of the SVM, data points that are clearly on either side of the decision hyperplane do not influence the model. But enhancing an example that lies close to the hyperplane will be guaranteed to have an effect on the new solution.

This idea has also been used in Active Learning with SVM in the works of [10], [11] and [9]. Formally, we rank examples by their distance to the dividing hyperplane:

$$\min_{\boldsymbol{x}} |w \cdot \boldsymbol{x} + b| \tag{5}$$

to maximally narrow the existing margin with an enhanced example. Evaluating the examples by their proximity to the hyperplane is computationally inexpensive as it only requires a single dot product computation.

At the same time, the current view level and the cost of enhancing this example need to be taken into account. This problem can be formulated as a 0-1 Knapsack Problem. Formally, we are given a m-tuple of positive numbers $\langle v_1, v_2, \ldots, v_m \rangle$ (the values). In this case, the values correspond to the distance to the hyperplane of the SVM.

$$v_i = |w \cdot \boldsymbol{x}_i + b| \tag{6}$$

They just need to be transformed to positive discrete values for the Knapsack formulation:

$$v_i = 100 - (v_i - \min(v)) \cdot \frac{100}{\max(v) - \min(v)} \tag{7}$$

We are also given a m-tuple of corresponding costs $\langle c_1, c_2, \ldots, c_m \rangle$ and a maximum cost $C > 0$. We wish to determine the subset $T \subseteq \{1, 2, \ldots m)$ that

$$\begin{aligned} \text{maximizes: } & \sum_{i \in T} v_i \\ \text{subject to: } & \sum_{i \in T} c_i \leq C \end{aligned} \tag{8}$$

This is an optimization problem that can be solved with brute force by trying out all 2^m possible subsets in T. In this paper, we trade space for time and use a dynamic programming approach which runs in $O(m \cdot C)$. If C is very large and corresponds to the total cost of enhancing all m objects to the highest level, the running time is $O(m^2)$. However, we expect that C is very small and constant in each iteration, therefore the algorithm runs in linear time.

V. Experiments

Before we go into the detailed descriptions of the experiments, we state our experimental methodology. All experiments have been designed in such a way that they are reproducible. We have created a webpage[3] that contains all datasets and the code that have been used in this work. The algorithms were implemented using the PRTools software [28].

Each experiment has been repeated 100 times. In each iteration, we split up the dataset randomly and use 40% for training and 60% for testing. All training instances are first assumed to have the lowest quality V_1. A batch of examples is selected in each iteration (plotted on the x-axis) and the mean classification accuracy (given the ground truth in the testing data on the highest view level V_j) is plotted on the y-axis. We also plot the mean accuracy of a classifier on each view level. As the approach in this paper is new, we use the random selection from Section IV-C as a baseline.

In all experiments, we have chosen the size of the budget in a way that allows the sampling schemes to reach the accuracy of the highest level.

A. Banana Dataset

The goal of the first experiment is to demonstrate the principle of operation and the benefit of our new algorithm on a dataset that is easy to visualize. We have generated a two-dimensional dataset with two classes with a banana shaped distribution. The data is uniformly distributed along the bananas and is superimposed with a normal distribution with a standard deviation s in all directions. The class priors are $P(1) = P(2) = 0.5$.

We have created two views: a good view V_1 (see Figure 4) by using a small standard deviation and a bad view V_2 (Figure 5) with a large standard deviation. As can be seen, V_2 is very noisy, but the underlying concept of two opposed banana shapes is still the same. We have used a SVM with a Radial Basis Function (RBF) kernel; the γ parameter has been set to 2.0. The SVM classifier is plotted as a solid black line. The classification in view V_1 (Figure 4) reflects our ground truth and is therefore plotted as a dotted black line in the other views in Figure 5-7.

Due to the high standard deviation, the classification in view V_2 is far from optimal. In this experiment, we use a small budget of 60 (resulting in 30 improved examples) and plot the improved dataset and the corresponding classifier that is learned in this data space. Figure 6 shows the new dataset and classifier with our Adaptive Active Algorithm (AAA) and Figure 7 shows the new dataset and classifier with randomly improved examples.

We can observe that the strategy of choosing examples close to the devision boundary of the AAA improvement results in a better classification accuracy than random improvement. Random improvement will eventually reach the same performance but needs more examples (and therefore more budget) to do so. We will take a more detailed look at the classification accuracy at specific points in time in the next experiments.

[3]Not yet published: Double Blind Review

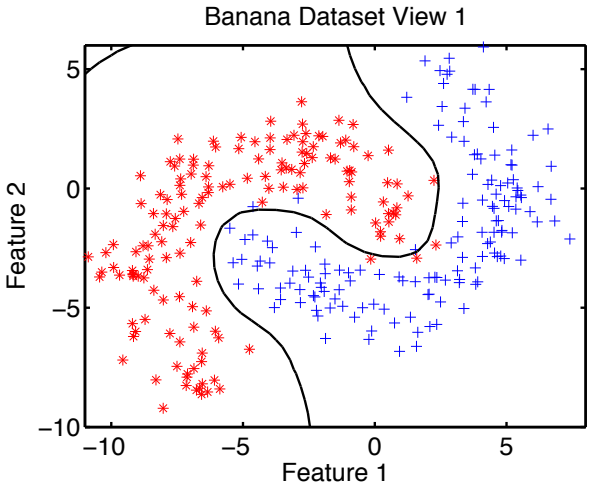


Figure 4. Banana Dataset View 1 ($s = 1$).

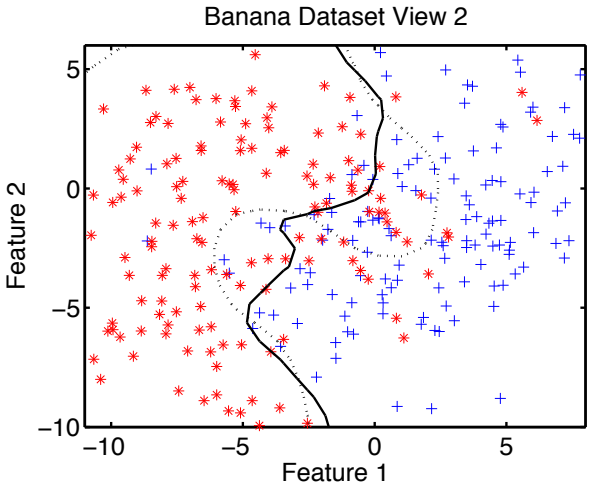


Figure 5. Banana Dataset View 2 ($s = 2.5$).

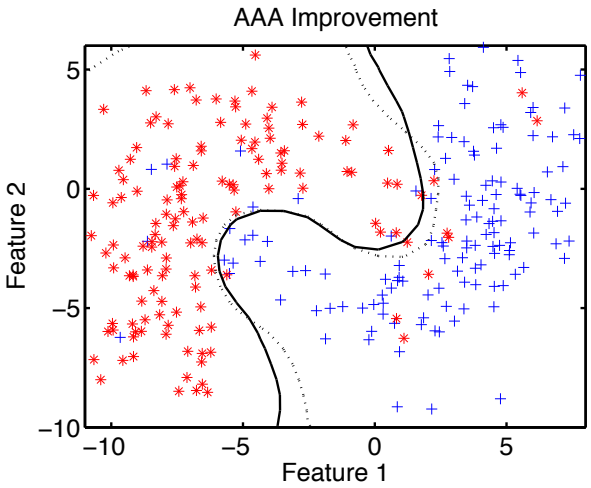


Figure 6. Improved Dataset with the Adaptive Active Algorithm.

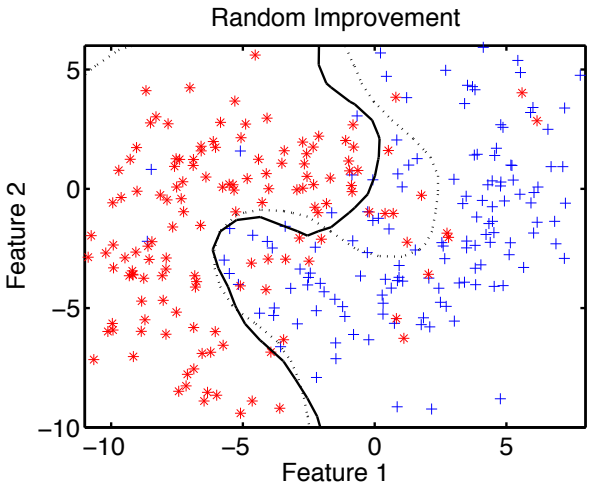


Figure 7. Improved Dataset with Random Adaption.

B. *Faces Dataset*

The faces dataset[4] contains 640 images of faces. The faces themselves are images of 20 former machine learning students and instructors, with about 32 images of each person. Images vary by the pose (direction the person is looking), expression (happy/sad), face jewelry (sun glasses or not), etc. We calculated the Zernike Moments on the order of 9 (see [29]) for each image and considered two binary classification tasks of identifying the pose of the person. We have used a linear SVM for this classification task.

Figure 8 shows some examples from this dataset on different view levels. As we are not using the image objects themselves but the Zernike Moments, we also show a reconstruction of the image based on the Zernike Moments. This gives us an intuitive idea how the algorithm perceives the image on each view level. We can observe that the features on Level 1 are not sufficient to discriminate between the two classes 'Right' and 'Straight'. Whether the person wears sunglasses or not seems to have a strong influence on Level 1. On Level 3, we can see that a clear pattern emerges for each class.

Figure 9 shows the test accuracy on the task of discriminating between a person looking to the right or looking straight. The total budget that can be spent to enhance all training examples to Level 3 is 768. We have spent a budget of 249, meaning we had a budget of 3 in each of the 83 iterations. The performance of our AAA scheme (solid black line) is consistently better than random refinement (dotted black line) over all iterations.

The Level lines in the preceding plot give the wrong impression of being available from the beginning. In fact,

[4]http://www.cs.cmu.edu/afs/cs.cmu.edu/user/mitchell/ftp/faces.html

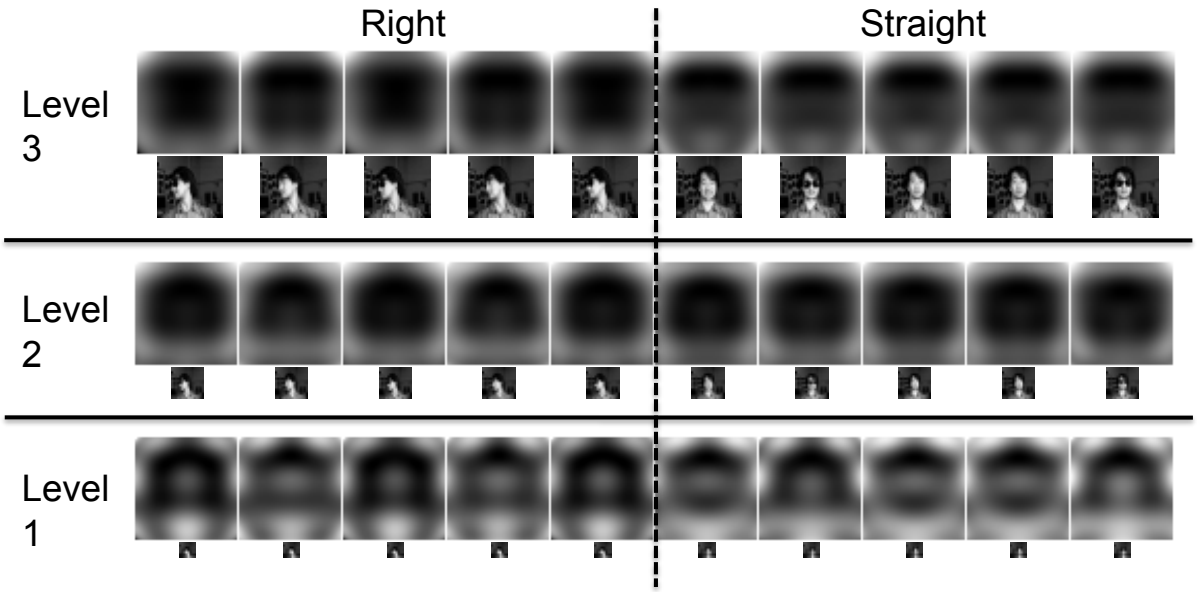


Figure 8. Zernike features of the Faces dataset on different levels.

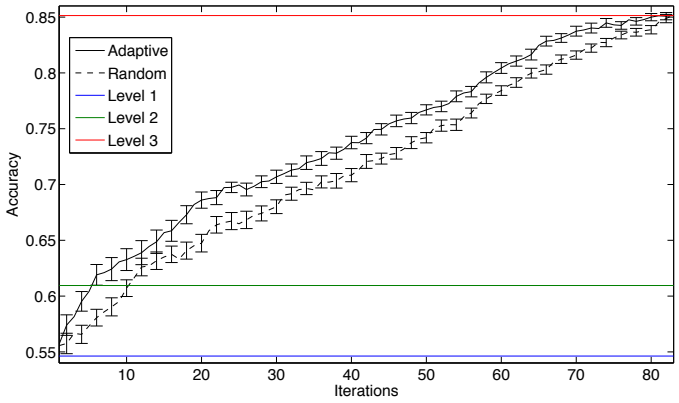


Figure 9. Test Accuracy 'Right' vs. 'Straight', Budget 249.

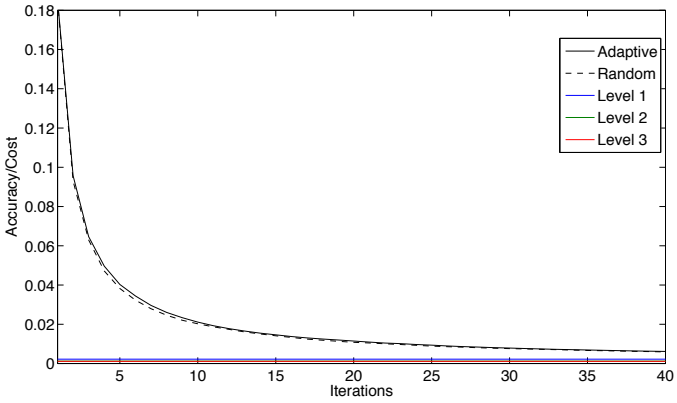


Figure 10. Test Accuracy / Cost, 'Right' vs. 'Straight', Budget 249.

one needs to invest the full cost of the whole level to obtain the corresponding classification accuracy. In order to clarify this issue, we show another perspective on this result in Figure 10. Instead of plotting the classification accuracy vs. the costs, we plot the classification accuracy per budget spent. That means, we take the classification accuracy of each scheme and divide it by the budget that needs to be spent to obtain it. As can be seen, this value is very low for all view levels, as the better accuracy value in each level gets divided by a higher cost value. Especially in the first iterations, a selective sampling scheme is very effective in terms of obtaining a better classification accuracy at less costs.

Figure 11 shows the test accuracy on a second classification task of discriminating between a person looking to the left or looking up. The budget amount stays the same as in the preceding classification task. This time, our AAA scheme needs more iterations to reach a better performance than random refinement. If we look at the classifier performance on the different levels, we note that the classification on the lowest level is poor (under 50%, meaning that it is worse than guessing). This shows the dependency of the AAA scheme on a useful initialization, which depends on a careful design of quality levels. This is a general problem of the quality of the input data and not a problem of the algorithm itself.

C. Numbers

The multiple features dataset from the UCI Machine Learning Repository [30] consists of features of handwritten numerals ('0'-'9') extracted from a collection of Dutch

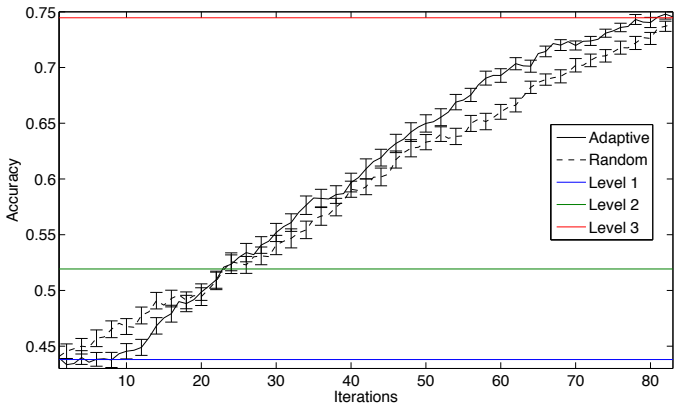


Figure 11. Test Accuracy 'Left' vs. 'Up', Budget 249.

utility maps. Two hundred patterns per class (for a total of 2,000 patterns) have been digitized in binary images. The original dataset consists of 6 different feature sets; we have used the pixel values to reconstruct the original images.

Based on these images, we have computed two views on these objects: the Zernike Moments of order 10 based on the original image of size 16x16 (Level 2) and of a subsampled image of size 8x8 (Level 1). We have used a linear SVM for this classification task.

Figure 12 shows some example objects from each class on the different levels along with the reconstructed images from the Zernike Moments. Although the patterns of the Zernike Moments in Figure 12 on Level 1 are hard to distinguish from a human perspective, they are already very discriminative. Each class has its own pattern, but some classes have very similar patterns, e.g., class '8' and class '9'. The quality on Level 2 is good enough to reach nearly a 100% accuracy.

The total budget that can be spent on the training examples is 800. We have used a total budget of 200 to enhance the training examples; therefore we could spend 4 units in each iteration to enhance some examples. As can be seen in the next examples, the performance of the AAA classifier reaches the same performance as a classifier on the best level, but with spending only a quarter of the budget.

Figure 13 shows the test accuracy of discriminating between class '1' and '5', Figure 14 between class '3' and '8', Figure 15 between class '5' and '7' and Figure 16 between class '7' and '8'.

In all classification tasks, the performance of our AAA classifier is clearly above random refinement and also converges faster.

VI. Conclusions

In this work, we have proposed a new learning setting, where the objects of interest can be obtained at differing quality levels and corresponding costs. The underlying idea of this approach is that we do not need to compute the best representation for all objects. We argue that it is sufficient to learn a classifier on the lower quality level and use a selection strategy to enhance useful objects. This setting is very useful whenever we have limited resources to compute a full feature representation of an object.

We have proposed a new scheme that improves a few selected examples with a support vector machine as the underlying classifier. Examples are ranked by their distance to the separating hyperplane and a dynamic programming approach is used to select a subset of interesting examples under a given budget constraint. Experiments on different datasets have shown that this scheme clearly outperforms random improvement and provides high classification accuracy with lesser costs.

In future work, we want to analyze the effect of different cost models and their influence on the selection scheme. We also want to explore how we can create hierarchical representations of objects in other domains. As can be seen, this new notion of objects in a hierarchy raises several interesting research questions and is worth to be further explored.

References

[1] S. Rueping and T. Scheffer, Eds., *Proceedings of the ICML 2005 Workshop on Learning with Multiple Views*, 2005. [Online]. Available: http://www-ai.cs.uni-dortmund.de/MULTIVIEW2005/MultipleViews.pdf

[2] E. H. Adelson, C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden, "Pyramid methods in image processing," *RCA Engineer*, vol. 29, no. 6, pp. 33–41, 1984.

[3] D. A. Cohn, L. Atlas, and R. E. Ladner, "Improving generalization with active learning," *Machine Learning*, vol. 15, no. 2, pp. 201–221, 1994.

[4] D. J. C. MacKay, "Information-based objective functions for active data selection," *Neural Comput.*, vol. 4, no. 4, pp. 590–604, 1992.

[5] N. Roy and A. McCallum, "Toward optimal active learning through sampling estimation of error reduction." in *International Conference on Machine Learning (ICML)*, C. E. Brodley and A. P. Danyluk, Eds. Morgan Kaufmann, 2001, pp. 441–448.

[6] D. A. Cohn, Z. Ghahramani, and M. I. Jordan, "Active learning with statistical models," in *NIPS*, G. Tesauro, D. S. Touretzky, and T. K. Leen, Eds. MIT Press, 1994, pp. 705–712.

[7] M. Lindenbaum, S. Markovitch, and D. Rusakov, "Selective sampling for nearest neighbor classifiers." *Machine Learning*, vol. 54, no. 2, pp. 125–152, 2004.

[8] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby, "Selective sampling using the query by committee algorithm," *Machine Learning*, vol. 28, no. 2-3, pp. 133–168, 1997.

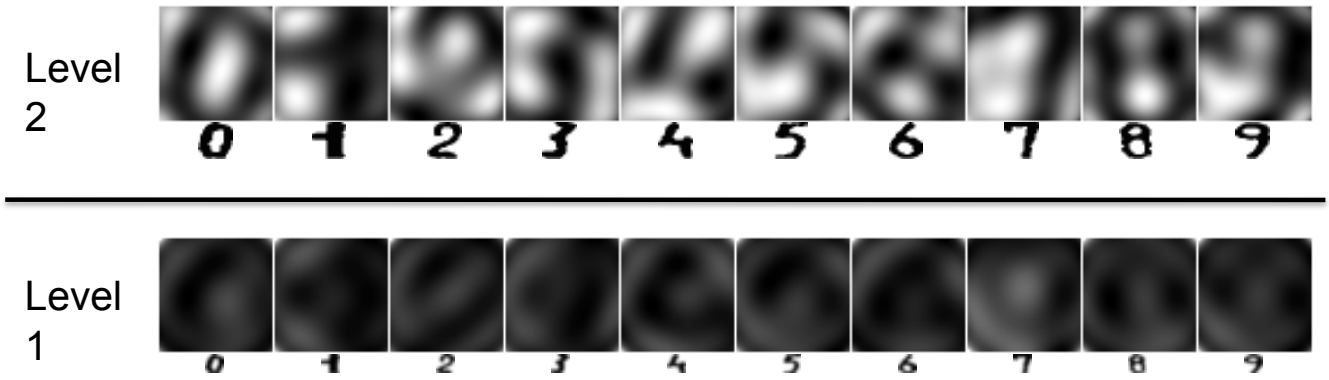


Figure 12. Zernike features of the Numbers dataset on different levels.

[9] S. Tong and D. Koller, "Support vector machine active learning with applications to text classification," *Journal of Machine Learning Research*, vol. 2, pp. 45–66, November 2001.

[10] G. Schohn and D. Cohn, "Less is more: Active learning with support vector machines." in *ICML*, P. Langley, Ed. Morgan Kaufmann, 2000, pp. 839–846.

[11] C. Campbell, N. Cristianini, and A. J. Smola, "Query learning with large margin classifiers," in *ICML '00: Proc. of the Seventeenth International Conference on Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, pp. 111–118.

[12] Y. Baram, R. El-Yaniv, and K. Luz, "Online choice of active learning algorithms," *J. Mach. Learn. Res.*, vol. 5, pp. 255–291, 2004.

[13] T. Osugi, D. Kun, and S. Scott, "Balancing exploration and exploitation: A new algorithm for active machine learning," in *ICDM '05: Proceedings of the Fifth IEEE International Conference on Data Mining*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 330–337.

[14] N. Cebron and M. R. Berthold, "Active learning for object classification: from exploration to exploitation," *Data Min. Knowl. Discov.*, vol. 18, no. 2, pp. 283–299, 2009.

[15] M.-F. Balcan, A. Beygelzimer, and J. Langford, "Agnostic active learning," in *ICML '06: Proceedings of the 23rd international conference on Machine learning*. New York, NY, USA: ACM, 2006, pp. 65–72.

[16] S. Dasgupta, A. T. Kalai, and C. Monteleoni, "Analysis of perceptron-based active learning," *J. Mach. Learn. Res.*, vol. 10, pp. 281–299, 2009.

[17] A. Kapoor, E. Horvitz, and S. Basu, "Selective supervision: guiding supervised learning with decision-theoretic active learning," in *IJCAI'07: Proceedings of the 20th international joint conference on Artifical intelligence*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007, pp. 877–882.

[18] B. Settles, M. Craven, and L. Friedland, "Active learning with real annotation costs," in *Proceedings of the NIPS Workshop on Cost-Sensitive Learning*, 2008, pp. 1–10.

[19] Z. Zheng and B. Padmanabhan, "On active learning for data acquisition," in *ICDM '02: Proceedings of the 2002 IEEE International Conference on Data Mining*. Washington, DC, USA: IEEE Computer Society, 2002, p. 562.

[20] M. Saar-Tsechansky, P. Melville, and F. Provost, "Active feature-value acquisition," *Management Science*, vol. 55, no. 4, pp. 664–684, 2009.

[21] P. A. Viola and M. J. Jones, "Rapid object detection using a boosted cascade of simple features," in *CVPR: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, 2001, pp. 511–518.

[22] B. Schölkopf, C. J. C. Burges, and A. J. Smola, Eds., *Advances in kernel methods: support vector learning*. Cambridge, MA, USA: MIT Press, 1999.

[23] A. McCallum and K. Nigam, "Employing em and pool-based active learning for text classification," in *International Conference on Machine Learning (ICML)*, J. W. Shavlik, Ed. Morgan Kaufmann, 1998, pp. 350–358.

[24] M. I. Mandel, G. E. Poliner, and D. P. W. Ellis, "Support vector machine active learning for music retrieval," *Multimedia Syst.*, vol. 12, no. 1, pp. 3–13, 2006.

[25] L. Wang, K. L. Chan, and Z. Zhang, "Bootstrapping svm active learning by incorporating unlabelled images for image retrieval." in *CVPR: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, 2003, pp. 629–634.

[26] T. Luo, K. Kramer, D. B. Goldgof, L. O. Hall, S. Samson, A. Remsen, and T. Hopkins, "Active learning to recognize multiple types of plankton." *Journal of Machine Learning Research*, vol. 6, pp. 589–613, 2005.

[27] M. K. Warmuth, J. Liao, G. Rätsch, M. Mathieson, S. Putta, and C. Lemmen, "Active learning with support vector machines in the drug discovery process." *Journal of Chemical Information and Computer Sciences*, vol. 43, no. 2, pp. 667–673, 2003.

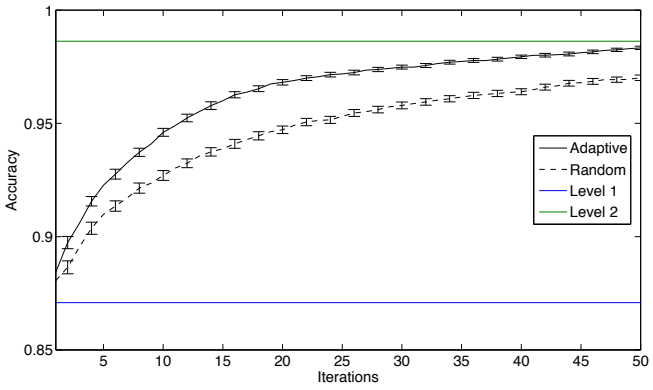


Figure 13. Test Accuracy '1' vs. '5', Budget 200.

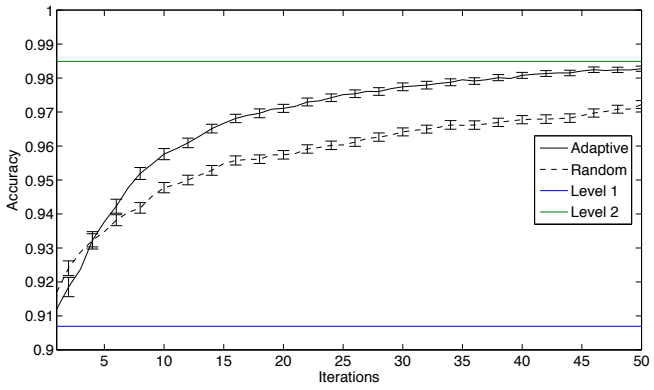


Figure 14. Test Accuracy '3' vs. '8', Budget 200.

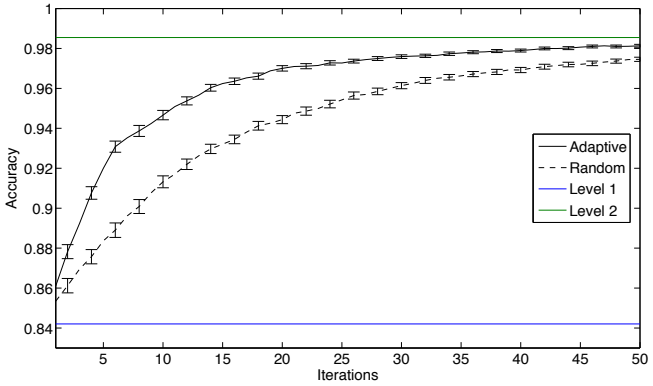


Figure 15. Test Accuracy '5' vs. '7', Budget 200.

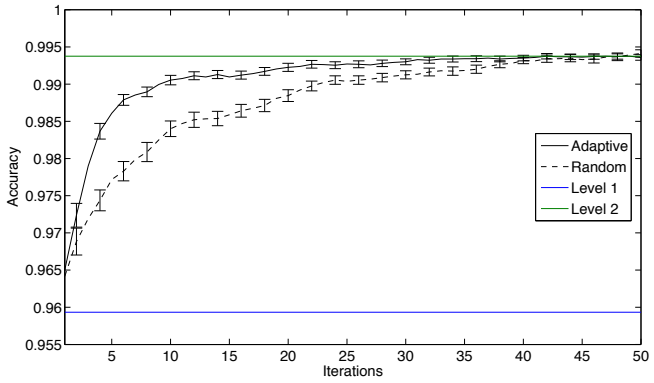


Figure 16. Test Accuracy '7' vs. '8', Budget 200.

[28] F. van der Heijden, R. Duin, D. de Ridder, and a. Tax, *Classification, parameter estimation and state estimation: An engineering approach using Matlab*. Wiley, 2004.

[29] F. Zernike, "Diffraction theory of the cut procedure and its improved form, the phase contrast method," *Physica*, vol. 1, pp. 689–704, 1934.

[30] A. Asuncion and D. Newman, "UCI machine learning repository," 2007. [Online]. Available: http://mlearn.ics.uci.edu/MLRepository.html