

LVQ Active Learning for Boar Sperm Head Image Classification

Caesar Ogole

Department of Computing Science
University of Groningen, The Netherlands
C.Ogole@student.rug.nl

Abstract

The highly prohibitive costs associated with obtaining pre-classified training examples for Learning Vector Quantization (LVQ) networks call for techniques that rely on sub-samples of data. We report in this paper our attempt to employ a version of semi-supervised learning procedure commonly referred to as active learning to classify pre-processed boar sperm head images. The undertaking is in quest for automated methods that help reduce the labelling efforts required by the teacher, and to reduce as much as possible the number of examples used in the course of learning. We achieve this by inducing a classifier from a minimal amount of data. The classification scheme is prototype based. This means that prototypes are updated according to some selected measure of (dis)similarity of the codebook vectors with the given training example data. The task of identifying the most informative training examples is non-trivial. We provide an extension of the well-known winner-takes-all (WTA) rule in which the prototype update function is parameterized mainly by the most confusing example at a given training step. Experimental results confirming better performance of the active learner (and variations) over the standard LVQ 1 are included in the report.

Keywords: learning vector quantization, active learning, image classification

1 Introduction

The need for sufficient training examples in Pattern Recognition (PR) systems remains a major setback for the applications of the PR techniques. In this sense, PR concerns itself with the description and classification of measurements or features. No matter what machine learning approach is employed in a given PR system, the success of the application largely depends on the amount (and reliability) of the data used to train the classifier. The greater the amount of training examples used, the more *intelligent* is the classification system.

However, it is often expensive to gather the ideally large samples of the training data. The cost and time of getting the training examples may be highly prohibitive. This clearly suggests that we often have to work with the dramatically fewer training examples. Without having any modifications made on the usual PR models, the classification scheme may not produce any useful results. In any case, it has also been shown that the average amount of new information per sample decreases during learning. This implies that additional information is characteristically redundant.

Active learning procedure is one that attempts to address the problems associated with limited training examples, or better still, restricts itself to only few actively selected training examples. The learner has the capability to select the training data that it expects to be the most informative. In this respect, the learner ceases to be a mere passive recipient of information.

In this study, we have concerned ourselves with an extension of a classification scheme which falls under a family of machine learning algorithms popularly known as learning vector quantization (LVQ). LVQs often provide robust mechanisms for classification of high-dimensional data and employ intuitive metrics (Euclidean distances, for example) as the measure of similarity among the given data and the prototypes (also called codebook vectors). In this perspective, an example data (or data point) is regarded as a point in N-dimensional space, where N is defined by the dimensionality of the data (and N is often large in practice). It is worth mentioning that there exist several variants of LVQ algorithms with diverse applications (Biehl, et al., 2006). The main task in active learning is to involve the learner (LVQ network) in training using only the most informative/interesting training examples. The ultimate goal is to construct a classifier that recognizes correctly an unseen data instance with high certainty (after training). This report therefore includes a brief comparative analysis of the standard LVQ 1 and the LVQ active learning models with respect to generalization. We will, however, start by describing the structure of the data used in this set of experiments. After this, the active training algorithm and the training strategies will be given the subsequent sections. We will discuss general issues before we end with making conclusive remarks on the study.

2 Experimental Data

The input data employed in these experiments consists of real world data samples that represent microscopic images of boar sperm heads. Each sperm head image is accompanied by a label indicating which of the two categories the image belongs -- normal or dysfunctional. This pre-classification was done by a human (veterinary) expert. The images (Fig. 1) were pre-processed using relevant image processing/transformation techniques including segmentation, resizing and rotation in order to obtain the image data used in this study.

The images that were taken using phase-contrast microscope usually consisted of a number sperm head images and background objects (noise). Image segmentation was therefore necessary for extracting individual images from these ‘raw’ images. It is also important that the pixels have same orientation since our technique of image classification relies on pixel-based comparison. Since most of the sperm heads are nearly oval-shaped, it is possible to determine their principal axis, and rotate the images along this axis such that they all have same orientation. The images that are distorted (not ellipsoidal in shape) are assumed to be background objects.

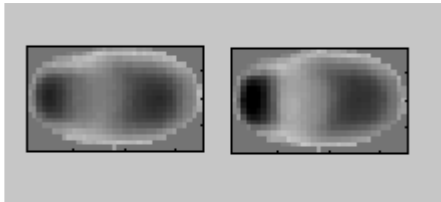


Fig. 1: Example boar sperm head *prototype* images. Which of the two is normal? (And which one is dysfunctional?)

A total of 1368 images were available for use with each image resized to 19x35 (in pixels), or 665 pixels for each image. Resizing is yet another one of the image pre-processing steps to ensure that all the images are equal in size. The arrangement may be visualised as depicted below (Fig 2):

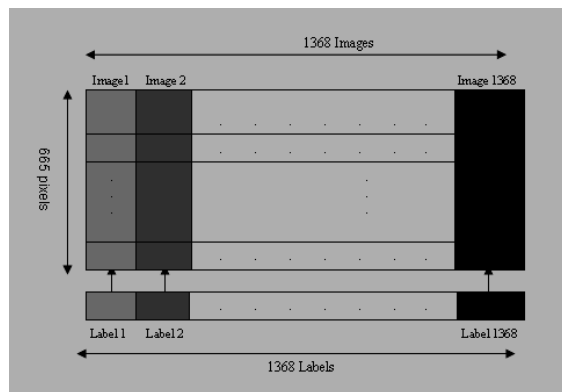


Fig. 2: Image data arrangement

The pre-processing also involved application of image transformation techniques to ensure that the differences in the brightness and contrast of the images are ironed out.

2.1 Notation

We describe a typical data set using the following notation:

Training set of examples is denoted as:

$$D = \{ \xi^1, \xi^2, \dots, \xi^P \} \text{ with (class) labels } L^i, (i = 1, 2, \dots, P)$$

In a only two class problem, D may be described using the compact notation:

$$D = \{ \xi^\mu \in \mathcal{R}^N, S^\mu \in \{-1, +1\} \}_{\mu=1}^{\mu=P}$$

This may be read as “the data set D consists of P feature vectors each of dimension N . The training label consists of the values ± 1 , representing the two different class memberships available”.

The next sections include descriptions of how the data are split into disjoint sets that suit the training techniques adopted here. However, we note, in advance, that for the purpose of evaluating the performance of the active learning procedure, we used only sub-samples of the available data set.

3 Active Learning in Classification

In this section, we describe the algorithm and heuristics used in the query learning for boar sperm head image classification. The basic idea adopted here is analogous to that in the ESANN 2006 paper (Biehl, et al).

3.1 The algorithm

Basically, we have (as stimulus vectors) a training set of examples:

$$D = \{ \xi^1, \xi^2, \dots, \xi^P \} \text{ with labels } L^i.$$

And a set of prototype vectors:

$$W = \{ w^1, w^2, \dots, w^Q \} \text{ with labels } S^k, \text{ with } (k = 1, 2, \dots, Q)$$

The learner is often interested in finding the most informative next example, ξ^v . Several plausible variations are available for the intermediate steps of this procedure. We will first convey the basic idea of this algorithm by adopting one approach, and in the later sections, tackle the other possibilities.

In this approach, the LVQ system is trained by iteratively identifying the next (training) example $\xi^v \in D$ and the winning prototype, $w^* \in W$,

and consequently updating, W^* , according to the rule:

$$w^* \rightarrow w^* \pm \frac{\eta}{N} (\xi^v - w^*)$$

where:

$$\xi^v \text{ is } \xi^\mu \in D \text{ with } M^v = \min_{\mu} \{M^\mu\},$$

and

$$M^\mu = \left| \left(\xi^\mu - w_k \right)^2 - \left(\xi^\mu - w_\tau \right)^2 \right|$$

(that is, the difference in the squared distances, in absolute).

w_k is the closest correct prototype and w_τ is the closest wrong prototype for the respective training instance ξ^μ ($\mu = 1, 2, \dots, P$). η is the learning rate.

In the update step, w^* is the prototype with minimal $(w - \xi^\mu)^2$ and the sign \pm is "+" if $w^* = w_k$ (correct winner), or it is "-" if $w^* = w_\tau$ (wrong winner).

When to stop the training is also an area of interest. Possible heuristic approaches include specifying the number of epochs (or early stopping) or by specifying some values of tolerance.

3.2 LVQ active learning as an extension of WTA rule

The intuition behind the algorithm is based on the winner-takes-all (WTA) rule. If d_+ and d_- be the distances of the closest correct prototype, $w_k \in W$ and the closest wrong prototype $w_\tau \in W$, respectively, from the selected image $\xi^v \in D$ in a given update step, then we know that w_k is closer to ξ^v (than w_τ) if $d_- > d_+$, otherwise w_τ is the winning prototype. Whatever turns out to be the case, we denote the winning prototype by w^* . We say that w^* is a correct winner if its class label and the class membership of ξ^v are the same; otherwise, w^* is a wrong winner. This concept is important as it is applied in the determination of the update direction. w^* is moved closer to the data instance if it is a correct winner (for the classifier is correct currently), otherwise it is moved away from the data instance. In a two prototype situation, the (hyper)hyperplane would be the mid-plane between the correct and the wrong winner. The idea generalizes to more

prototypes; the actual boundary in charge for a particular example is always defined by a subset of two.

Perhaps, the main area of interest in this study is the criterion used in selecting ξ^v . In this version of LVQ training, we want to pick the most interesting $\xi^{\mu+1} \in D$ to be used in the next training step. We search *actively* for this example during the course of learning. Recall that

$$d_+ = \left(\xi^\mu - w_k \right)^2 \text{ and } d_- = \left(\xi^\mu - w_\tau \right)^2$$

(Fig 3).

If we let $M^\mu = | -d_+ + d_- |$, then the example

ξ^v with minimal $M^\mu = M^v$ is one that the classifier is currently least certain about (after LVQ training with μ examples). We take this example to be the most informative to the learner. (The interested reader is advised to visualize this heuristic rule geometrically for the sake of clarity). An appropriate prototype update is performed depending on whether the learner classifies this very confusing instance correctly, or not.

Like mentioned before, it is possible to use other principled or heuristic variations of the rule in the selection of ξ^v from the computed values of M^μ , for example, instead of looking for ξ^μ with minimal M^μ , we could go through the list of examples (randomly) until we find a data instance with $M^v < \theta$, where θ is some defined threshold.

We could think of θ as a quantity that defines the acceptable *level of confusion* induced by the example ξ^v . In this learning scenario, learning begins with large θ , and then its value is decreased during the course of the training. However, It should be noted that this strategy is simply a generic form of our preferred search for the minimal M^μ , in which case a student always aims at selecting an example that it knows least about its label (at that time step).

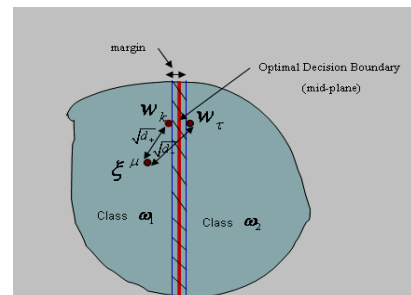


Fig. 3: LVQ active learning for a simple two prototype situation

4 Experiments, Methods and Training Strategies

In this section, we describe the methods employed in carrying out the set of experiments under the active learning scheme. Specifically, we describe how the available data was divided into training and test sets. We also include detailed description of the training and validation techniques adopted as well as the choice of the algorithm's major parameters. The parameters discussed are the learning rate, number of prototypes and the initial values of the prototypes. Earlier experiments have indicated that the choices of the parameter values do have some influence on the experimental results. In this study, we don't include the experiments to (dis)confirm these assertions, but rather, we adopt and explain briefly the some of the suitable methods, strategies and parameter settings as reported in the earlier findings.

4.1 Obtaining the training and test datasets

We begin by providing insight into how the image data was split into training and test sets. Given a data set D , our implementation initially aims at creating two disjoint sets from D , of which one set (one-tenth of D , in size) is put aside for testing, and the remaining nine-tenths is used for training. The elements in either set are selected at random from D .

How well the learner classifies the images in the test set after sufficient training gives the measure of the performance of the classifier. The stimulus vector at a given training step is always the example that is closest to the decision boundary. This means that, if P is the number of examples in the training set, then one training cycle (epoch) is equivalent to P training steps. In our experiments with active learning algorithm, the learners cycles through the data for as long as there is significant difference in the successive test error values.

4.2 The k-fold cross-validation

To heighten our confidence in the reliability of the experimental results, we employed the k-fold cross-validation strategy. Specifically, we chose $k=10$ (hence 10-fold cross validation), in which case, the available dataset is split into ten equal disjoint sets (not only 2 as described before), each set containing approximately the same number of instances of each class. Using this strategy makes it possible to train ten different LVQ networks, each using a different set of the ten datasets for testing, and the remaining nine sets for training. The overall performance of the LVQ system is then computed as the average of all the ten networks. Since the generalization ability of LVQs is often affected by the initialization of the prototypes, the strategy of k-fold cross validation provides a nice mechanism for mediating this problem. Secondly, it is possible that a given dataset might have some unrepresentative

(outlier) elements. The k-fold cross-validation scheme makes a good work-around to this phenomenon by averaging over the results of the different randomly created datasets.

4.3 Choice of the exact value of P

Intuitively, P has to be a multiple of 10 (when performing 10-fold cross validation) to ensure that the ten different sets all have equal number of instances. Our implementation, however, provides flexibility in the sense that any $P>10$ may be divided into ten sets with at least nine sets having equal number of data elements. It is to be noted, however, that this rather more difficult implementation serves no better purpose than merely allowing for flexibility in the value P . This flexibility is not needed in the best case.

4.4 The initialization of prototypes

To minimize the effect of classifier sensitivity to prototype initialization, the prototypes' initial values are set to the mean of several random data instances of the same class. Preliminary investigation had indicated that this method would yield more pleasing results compared to, say, initialization of the codebook vectors by setting them to be the pixel values of random instances (of the class in question). Another advantage associated with good initialization of a codebook vector is that the initial prototype is placed in a position that is already close to the desired final position in the search space. Good initial placement of prototypes in the instance space means that we don't need to use large steps for the prototypes to reach their respective final positions. Large step size could mean that the prototype moves past (i.e. overstep) the solution or too far away in the wrong direction. In contrast, very small steps may go in the correct direction, but they also require a large number of iterations. This may be considered a stability/convergence issue in LVQs.

4.5 The learning rate

The 'leap' value discussed in the last part of subsection 4.4 is the so-called learning rate. In our experiments, we used a slowly degenerating function as the learning rate function. The initial value is set to 0.3.

The learning rate is given by the function:

$$\eta(t) = \frac{\eta_0}{[1 + b(\lambda - \lambda_0)]}, \text{ (for } \lambda > \lambda_0 \text{)}$$

where:

t is a variable standing for the number of training steps,

$$\eta_0 = 0.3,$$

λ is the total number of training steps,

λ_0 = amount of epochs to wait before decreasing

the learning rate,

b = speed at which the learning rate decays
(higher = faster)

In our experiments, b is set to the value 0.1.

This is the same learning rate function that was used in the previous study involving LVQ 1. The prototype is adapted by translating it by a factor η towards (or away from) the stimulus vector in a given training step. In the t^{th} step, η is smaller in value compared to the learning rate value used in the $(t - 1)^{th}$ step. For the active learning algorithm, the slowly decreasing values of η even makes more sense (intuitively) since the learner's level of confusion reduces with increasing training steps. Theoretically, the learner approaches zero confusion with sufficient training and thus need not adapt much, if at all, to stimuli (as $t \rightarrow \infty$).

We have discussed in this section the techniques employed in setting up and carrying out the set of experiments under the LVQ active learning scenario. A more general description of the procedural steps used in manipulating the image data is provided in the Appendix section of this paper. The discussion of the results of these experiments follows in the next section.

5 Results

In this section, we present the results of our set of experiments. Specifically, the experimental results given here focus on comparing the learning curves for LVQ 1 and the active learning algorithms.

The performance of the active learning algorithm and its variations with respect to generalization are also compared. For this purpose, we use the misclassification on the test set (generalization error) to quantify this important LVQ attribute.

5.1 Generalization ability of active learning versus that of passive learning

In this experiment, an equal number of training instances ($P=180$) was used to train both the active and passive learners under similar parameter settings and using the same training strategies described see Section 4. The learning curves for the two classifiers are shown in on the graph in Figure 4. From the graph, it is clear that the active learner has a higher generalization ability compared to the algorithm trained on randomly selected examples. The fluctuations are mainly due to prototype initialization.

Appendix B shows sample images reconstructed from the adapted prototypes (i.e. now converted from pixel vectors to real images) and their corresponding class labels.

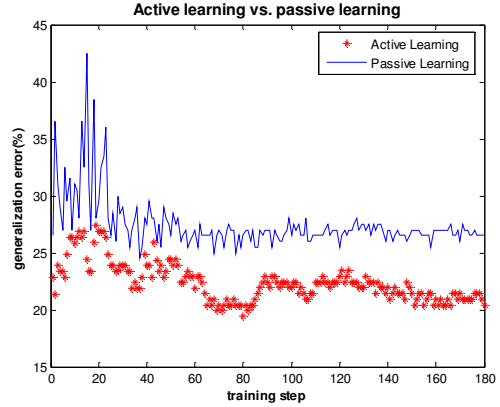


Fig. 4: Learning curves for active and passive learning

5.2 Variations of the Active Learning Algorithm

5.2.1 Active learning with unrepeated use of training examples

Our interest in this part of the study was to find out whether allowing or forbidding repeated use of examples could have some interesting results. The findings given in Section 5.1 are based on experiments where the LVQ network had the freedom to learn from an example repeatedly as long as it qualified to be a stimulus ξ^v in the subsequent step(s).

We therefore provided in the implementation of the active learning algorithm a sub-routine to remove from list of the examples a training instance as soon as the network has adapted to that instance. The known example would be introduced back in the pool of data only in the next training cycle. This restriction ensures that the learner encounters all examples in every cycle.

Although the results of our experiments showed that this adjustment (still) exhibits superiority over LVQ 1, the learning curves of the algorithms indicate that first method (without restriction) yields better generalization (Fig. 5). The relatively poor performance of this method can be explained by the fact that forbidding repeated use of an example in a given training cycle introduces a sort of off-by-one error in the active selection of an example at each training step. What this means is that the learner could have learnt more by retaking and adapting to the removed training example in the following learning step had it not been denied the chance to do so. The network now actively adapts to the next training instance which may (most probably) be a lesser informative example. The impact of this type of error is usually big in the case where the step size is so small, so that an already seen pattern still remains most confusing to the learner in the next consecutive steps.

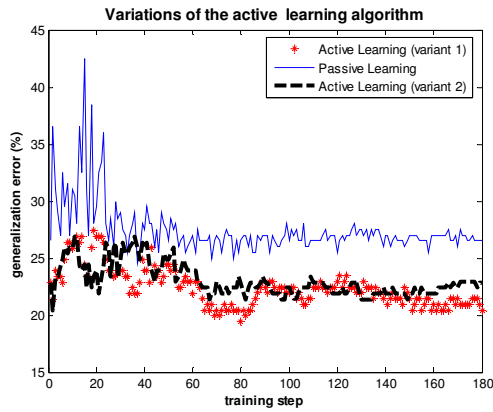


Fig. 5: Learning curves for active and passive learning (variant 1 =repeated use of examples variant 2=restricted use of examples)

5.3 Active learning and amount of training data

Our experiment to investigate the performance of the active learning algorithm with varying number of training examples yielded results that show that a much higher performance can be achieved with a rather smaller amount of training data. (This experiment is based on variant 1 of the algorithm described in Section 5.1). Figure 6 shows the trend for the active and passive learners. Note that in all these experiments, a learner performs only P training steps.

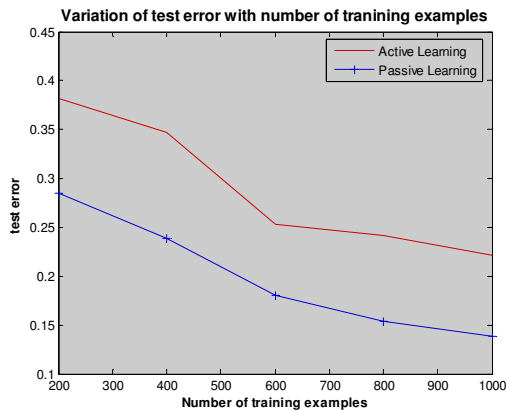


Fig. 6: Active learning and amount of training data

6 Discussions

Our experimental results confirm that it is possible to construct high performance classifiers from relatively smaller amount of data using the method of active selection of training data. The active learner also achieves better generalization faster than the passive learner.

However, these benefits come along with costs such as difficulty in implementation of the procedure, and prolonged training time. Even with this success, we still recommend further study with the same objective of improving performance where there is

limited training data. Possible ways of increasing the accuracy of LVQ under such learning conditions could be found by integrating active learning aspects in other LVQ variants such as LVQ+/-, GRLVQ, and so on.

Another area that needs further investigation is the determination of the learning rate function. Much as we used a trial-and-error method guided by plausible heuristics to determine a suitable function and initial values of associated parameters, we cannot posit that this is the most appropriate learning rate function. Some slight adjustments made on the learning rate values during our experiments indicated sensitivity of the experimental results with respect to this very important learning parameter.

We need to mention that when this work was already in progress, it was found out that some of the images (input to our ambitious algorithm) had not been labelled correctly by the human expert. This is not an uncommon occurrence since such exercises are usually prone to error due to human nature. We could thus attribute some fraction of the test error to the contribution by the incorrectly labelled training and test instances. Nonetheless, the results of our study provide a good insight into the performance of the learning vector quantization active learning algorithm with regards to generalization.

7 Conclusions

The performance of the active learning algorithm, as shown by our experiments, is generally higher compared to that of the standard LVQ 1. Even with only a small amount of data, it is possible to train prototypes that will classify as correctly as an LVQ 1 classifier. We therefore posit that active learning is a very suitable extension to (machine) learning algorithms and very applicable in learning scenario where amount of available training data is a major constraint.

8 References

- [1] M. Biehl, A. Ghosh, B. Hammer, "Dynamics and generalization ability of LVQ algorithms", University of Groningen, Chausthal University of Technology
- [2] M. Biehl, P. Pasma, M. Pijl, L. Sanchez, N. Petkov, "Classification of Boar Sperm Head Images using Learning Vector Quantization", Groningen University, Campus de Vegazana
- [3] R. Schakoff "Pattern Recognition: Statistical, Structural and Neural Approaches"
- [4] R. Duda, P. Hart, D. Stock, "Pattern Classification"

Acknowledgements

This work was supervised by Prof. M. Biehl. Michael Biehl leads the Intelligent Systems Research Group on Learning Vector Quantization (LVQ) at the Department of Computing Science, University of Groningen, The Netherlands. I do thank him for the great work. My appreciation is extended to the LVQ Research team as well.

Appendices

Appendix A

General Procedure/Strategies used in carrying out a typical experiment

In order to carry out an experiment using a sub-sample of the original 1368 images (Call this original set D_0), the following strategy was adopted:

1. Shuffle the data, D_0 (by randomly permuting through the data)
2. Take the first P feature vectors. Call this set D_1
3. Shuffle D_1
4. In order to perform 10-fold cross validation:
 - i. Divide D_1 into 10 disjoint sets
 - ii. Put one set aside for testing
 - iii. Use 9 (out of the 10) sets for training (Call this set D_2)
5. Generate n prototypes from D_2 by taking the average of 1/10 of the elements in D_2 (same class) as the initial value of the n prototypes. (In our experiments, $n=8$; 4 prototypes for class 1 and 4 prototypes for class 2)
6. Perform training (active learning) *according to the algorithm description given in Section 3.1 of this paper.*
7. Evaluate performance of the learned classifier basing on how well it classifies the novel instances (in the test set)

Appendix B

Sample images reconstructed from the adapted prototype pixel vectors (*axes in pixels*)

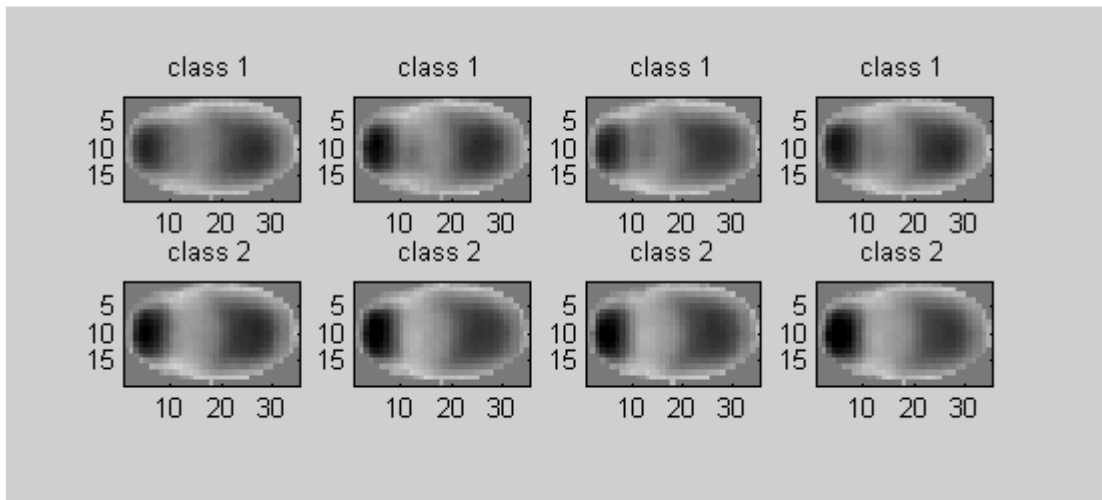


Fig.7: Reconstructed images: Observe that whereas *class 2* images (damaged/bad/dysfunctional sperm heads) share common features (such as dark spots to the right of the image, followed by a lighter spot, and again by a relatively darker spot to the left), these features are not that prominent in *class 1* images (normal sperm heads). Human veterinary experts would use some of these observable features (in so-called visual inspection) to distinguish between cells having “achrosomes intact” and those with “achrosomes not intact”.