Level Learning Set: A Novel Classifier Based on Active Contour Models

Xiongcai Cai[1,3] and Arcot Sowmya[1,2]

[1] School of Computer Science and Engineering,
The University of New South Wales, Sydney, NSW 2052, Australia
[2] Division of Engineering, Science and Technology, UNSW Asia, Singapore
[3] National ICT Australia, Locked Bag 6016, NSW 1466, Australia
{xcai,sowmya}@cse.unsw.edu.au

Abstract. This paper presents a novel machine learning algorithm for pattern classification based on image segmentation and optimisation techniques employed in active contour models and level set methods. The proposed classifier, named *level learning set* (LLS), has the ability to classify general datasets including sparse and non sparse data. It moves developments in vision segmentation into general machine learning by utilising and extending level set-based active contour models from the field of computer vision to construct decision boundaries in any feature space. This model has advantages over traditional classifiers in its ability to directly construct complex decision boundaries, and in better knowledge representation. Various experimental results including comparisons to existing machine learning algorithms are presented, and the advantages of the proposed approach are discussed.

1 Introduction

Pattern recognition has been crucial for human survival, and development of reliable and accurate pattern recognition by machines is an important research area. Pattern classifier construction is central to pattern recognition. Although there are many techniques for pattern classification, it is well known in the field that each algorithm has its inherent drawbacks. The investigation and design of new efficient and accurate approaches are therefore vital for solving particular kinds of pattern recognition problems.

The research in learning object extraction indicates that machine learning techniques have great potential for automation and optimisation of object recognition from images. Based on existing research, there also remains the potential for generalising, testing and improving machine learning and computer vision techniques for general datasets.

In this paper, we propose a novel classification method that determines a decision boundary directly from any dataset based on active contour models and level set methods. We formulate the classifier learning problem as that of segmenting the feature space, and then apply an active contour (with a level set formulation) to perform the segmentation. The principal idea is that the

J.N. Kok et al. (Eds.): ECML 2007, LNAI 4701, pp. 79–90, 2007.

regularisation term included in the objective function that is minimised by the active contour provides some protection against over-fitting, because it penalises long boundaries. The approach is to utilise and extend level set based active contour models from the field of computer vision to create novel machine learning techniques, namely level learning set, for any general dataset. The optimisation mechanism of active contours is adapted to work in sparse feature spaces rather than in image space. In parallel, a decision boundary creation technique based on level set functions for classifier construction is proposed.

The approach explores a novel direction in pattern classifier construction and seeks to provide new insights into the application of active contours and the level set method. It moves developments in vision segmentation, based on optimisation, into general machine learning. In addition, it enables the level set method to handle sparse, non-spatial datasets rather than only spatial data. The paper shows how to embed a level set based active contour model in the framework of machine learning to achieve a very general pattern classifier. In experiments, our method is compared with standard machine learning and classification methods on the UCI repository dataset.

The rest of the paper is organised as follows. In Section 2 we review related work in classifier construction approaches and active contour models used in Level Learning Set Classifier. In Section 3, the proposed classifier is presented, where the algorithm is described in detail. In Section 4, the experimental results are shown where LLS is evaluated and compared to standard classifiers, and Section 5 concludes the paper.

2 Related Work

2.1 Pattern Classification

There are two general strategies for creating classifiers: namely generative learning and discriminative learning.

Generative learning [1,2] utilises an example data set to build a probability model by finding the best estimate of parameters for some known parametric form of distribution. One problem with these methods is that the best estimate of a parameter may not give the best classifier because the parametric model itself may not be correct. Another problem is that even a well trained classifier obtained using a parametric density model may not be an accurate description of the data due to limited number of parameters in the model.

Discriminative learning ignores probability and attempts to construct a good decision boundary directly, which is often extremely successful, especially when no reasonable prospect of modeling the data exists. It assumes that the decision boundary comes from one or another class of solutions and constructs a solution to choose the best element of that class. Techniques following the discriminative approach include nearest-neighbour methods [3], linear discriminative analysis [4], neural networks [5], support vector machines [6] and decision trees [7,8]. Recently, Yip et al. [9] proposed a data clustering method using level set methods

to identify density peaks and valleys in the density landscape, which advances contours to form cluster cores.

2.2 Active Contours in Computer Vision

Machine vision provides interesting and challenging problems and rich techniques in advancing machine learning. Active contours are techniques in vision used to detect objects in a given image u_0 using methods of curve evolution. The basic idea is to deform a curve to the boundary of the object starting with an initial curve C, under some constraints from the image u_0. To address curve evolution, deformable contour models or snakes were first presented [10] for detection and localisation of boundaries. Cohen [11] uses the balloon model to reduce the initialisation requirement of the snake model. This has been improved [12] using a geodesic formulation in a Riemannian space for active contours derived from the image content. Cohen and Kimmel [13] describe a shape modeling method by interpretation of the snake as a path of minimal cost which is solved using numerical methods. The Level set method has been utilised for shape modeling [14] as it allows for detection of automatic topology changes, cusps and corners. Geman and Jedynak [15] present an active testing model to reduce uncertainty in tracking roads in satellite images using entropy and statistical inference. These approaches only work for low level segmentation and are not suitable for higher level object extraction or recognition due to their inability to learn and utilise prior object knowledge. Chan and Vese [16] extended the scalar Chan-Vese algorithm for active contours to the vector valued case. The model minimises a Mumford-Shah function over the length of the contour, as well as the sum of the fitting error over each component of the vector-valued image. We have recently developed a method [17] to introduce control parameters into the speed function of level set methods. It utilises a genetic algorithm to tune those parameters to adjust for the effects of intensity and gradient features and force the marching of the active contours to stop at the object boundaries. We have also presented a novel active contour model using information fusion techniques [18].

We now present a new classification algorithm for constructing a decision boundary in any feature space, based on level set methods.

3 Level Learning Set (LLS) Classifier

Given a set of observations and their labels, we define the decision boundary construction problem as an optimisation problem involving data set partition and using a geometric formulation, with respect to some constraints. In this paper, we first split the general classifier construction problem into many one-class classifier construction problems using the divide and conquer strategy. One-class classification [19] is a new branch in pattern recognition that tries to describe one class of objects, and distinguish it from all other possible outlier objects, in contrast to normal classification where one tries to distinguish between two or more classes of objects. For each class, we make the assumption that the space of

measurements X is divided into two subsets: A belonging to the target class, and its complement outlier class $A^c = X \setminus A$. The classifier construction problem is then to seek a decision area in the feature space that maintains the characteristics of that class and represents the knowledge learned from the training data, which can be used to classify new data. The individual one-class classifiers may then be fused to construct the decision boundary for multi-class classification.

3.1 LLS in One-Class Classification

In one-class classification, often just the probability density of the training set is estimated. When new objects fall under some density threshold, they are considered to be outliers and are rejected. This approach has some shortcomings as discussed in 2.1. We propose a one-class LLS method which does not rely on density estimation. The method is inspired by the level set-based active contour model [12,14,16], which is an image segmentation method based on the minimisation of an energy function that models curve inflation, curve smoothness, and curve advection to image edges. It is designed to be geometric and robust and produces a one-class decision boundary that maximises a measure of within-class similarity, and minimises the length of the boundary and the area of the region inside the boundary, around a training set of objects.

Let Ω be a bounded open subset of $\Re^n$, with $\partial\Omega$ its boundary. Let $f(x)$ be a given function such that $f(x) : \overline{\Omega} \rightarrow \Re$, where $\overline{\Omega}$ is an order type of set Ω. Let $C(s) : [0,1] \rightarrow \Re^n$ be a parameterised curve. The classifier decision boundary is defined as the zero level set of an implicit function $z = \phi(x,t)$ defined on the entire feature domain. The contour at time t must satisfy the function $\phi(x,t) = 0$. The area inside the boundary C then represents an open subset of Ω, in which data points have a strong likelihood of belonging to the class. An energy function modeling the constraint for constructing the decision boundary is defined by

$$F(c_1, c_2, C) = \mu.Length(C) + \nu.Area(inside(C)) + \lambda_1 \int_{inside(C)} |f(x) - c_1|^2 dx + \lambda_2 \int_{outside(C)} |f(x) - c_2|^2 dx \quad (1)$$

where $Length(C)$ is the length of the curve C and $Area(inside(C))$ is the area of the region inside C. x is an n-dimensional vector representing a location in the feature space, c_1 = average(f) inside C and c_2 = average(f) outside C. μ, ν, λ_1 and λ_2 are weighting parameters. $f(x)$, namely feature pixel intensity, is a function over the whole feature space defined by

$$f(x) = count(instance_x^i). \quad (2)$$

$f(x)$ is the number of instances of the class i having a feature vector value of x.

The transformation from non sparse image data to general sparse data is not straightforward. Traditionally, level sets perform two-class segmentation on images, where training instances cover the whole working space. For general datasets, multiple instances may map to a single point in feature space, and

other areas in feature space may be left empty, which requires careful handling. We use definition (2) and divide the multi-class classification problem into several one-class ones to address this.

According to this configuration, the higher the feature pixel intensity that a data point in feature space has, the stronger the probability that it belongs to the class and resides inside the decision boundary C. We specifically use instances belonging to the target class to create the feature pixel intensity in (2), and ignore all other instances, as depicted in Figure 1. Information outside the boundary in definition (1) does not include instances from other classes.

Thus, the minimisation problem is defined as follows:

$$\inf_{c1,c2,C} F(c_1, c_2, C). \tag{3}$$

By calculus of variations [20], the Gateaux derivative of F in (1) is defined as

$$\frac{\partial \varepsilon}{\partial \phi} = -\delta_\epsilon(\phi)[\mu.div(\frac{\bigtriangledown \phi}{\mid \bigtriangledown \phi \mid}) - \nu - \lambda_1(f - c_1)^2 + \lambda_2(f - c_2)^2] \tag{4}$$

The function ϕ that minimises this functional also satisfies the Euler-Lagrange equation $\frac{\partial \varepsilon}{\partial \phi} = 0$, which is parameterised by an artificial time variable t as:

$$\frac{\partial \phi}{\partial t} = -\delta_\epsilon(\phi)[\mu.div(\frac{\bigtriangledown \phi}{\mid \bigtriangledown \phi \mid}) - \nu - \lambda_1(f - c_1)^2 + \lambda_2(f - c_2)^2] \tag{5}$$

We solve the level set problems by discretisation of the divergence operator as in [21] and iterative computing of the level set function value as in [22] using Equation (5). By iteratively computing the level set function value, we obtain a description in which the learned knowledge is represented by the level set function ϕ, inside average feature intensity c_1 and outside average feature intensity c_2.

Therefore, the one-class LLS classifier is built around construction of an indicator function that ties the location of the data to its class:

$$I_A(x) = \begin{cases} 1 \text{ if } x \in A \\ 0 \text{ if } x \in A^c. \end{cases} \tag{6}$$

To represent this indicator function, the implicit function $\phi(x)$ is used, which is in R^n. The implicit representation can be discretised, resolving an n-dimensional set D. This can even be avoided, in part, by placing all the points x very close to the boundary, leaving the rest of D unsolved. Since only the $\phi(x) = 0$ isocontour in dimension n-1 is important, only the points x near it are actually needed to accurately represent the boundary [23].

Instead of storing the complete training set, this knowledge representation is relatively simple. Moreover, due to the evolution of active contours, the learned contour can be directly used as the initial contour for learning from new training data, which makes it very suitable for incremental learning. Examples of decision boundaries created by one-class LLS classifiers are shown in Figure 1, where the two images in each row show the decision boundaries constructed for two different classes in each dataset separately, after 200 iterations, with bin size 30.

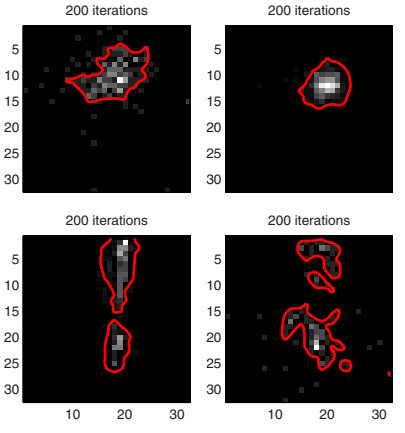

Fig. 1. Decision boundaries constructed by base one-class LLSs for dataset Wis-BC-Diag (top) and Ionosphere (bottom) from UCI dataset, are shown as closed curves

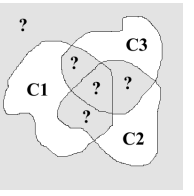

Fig. 2. The areas with question marks show areas where there is inconsistent output from one-class classifiers

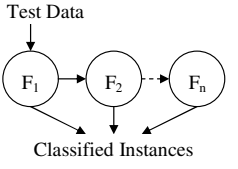

Fig. 3. LLS with cascade of classifiers where acceptance can occur at any stage

3.2 Extension to Multi-class Classification

When we have the one-class decision boundary in feature space for each class, the generalisation to multi-class classification is not straightforward for discriminants that are not based on density estimation. Simple combining methods use voting, but this has the drawback of inconsequent labellings and ties. Before proposing a method to construct the multi-class classifier from one-class classifiers, we first define the following decision areas in feature space:

$$Rejected\ Area_{i,j} = \sim Area(C_i) \cap \sim Area(C_j) \tag{7}$$

$$Competitive\ Area_{i,j} = Area(C_i) \cap Area(C_j) \tag{8}$$

$$Accepted\ Area_i = Area(C_i) \setminus_{j=1,j\neq i}^{n} Competitive\ Area_{i,j}. \tag{9}$$

where $Area(C_i)$ is a subspace of the feature space that remains within the decision boundary curve of class i.

According to the above definition, as shown in Figure 2, a one-class classifier is strongly confident in its decision when the test data point resides in its Accepted Area in feature space, since its decision does not conflict with that of other classifiers. However, dilemmas arise when decisions conflict with each other. Therefore, the decision function for multi-class classification is defined by

$$I_i(x) = \begin{cases} 1 & \text{if } x \in Accepted\ Area_i \\ unknown & \text{if } x \in Competitive\ Area_{i,j} \\ unknown & \text{if } x \in Rejected\ Area_{i,j}. \end{cases} \tag{10}$$

Algorithm 1. Learning(D) - left, Testing(k, T) - right.

```
Ensure: k←learned model
  Y←set of instances in training set D
  R←set of reference class labels
  B←subsets of downward projections on Y
  for all b in B do
    for all class i in R do
      k_{b,i}←add one-class LLS function
    end for
  end for
```

```
Ensure: d←class labels
  for all instance y in test set T do
    B←create downward projections on y
    for all b in B do
      for all class i in k do
        x_b←subset b in B
        D_i(x_b)←decision area of k_{b,i}(x_b)
        if D_i(x_b) is Accepted Area then
          d(x)←class i
        else
          p_i(x_b)←add class probability
        end if
      end for
    end for
    if d_y is not assigned then
      d_y←arg max_i Σ_{b∈B} p_i(x_b)
    end if
  end for
```

To minimise the classification error, the algorithm assigns a class label i to a test instance x only when the location of the instance x remains in the class Accepted $Area_i$ in the feature space, that is the region is inside one decision boundary and not overlapped by others. Otherwise, it refuses to make any decision and defers it to the next step.

3.3 The Final Proposed LLS

As shown in the previous section, all classifiers might conclude that a certain sub feature space does not have their class characteristics and thus consider instances located in this sub space as not being in their class. In this case, the instance is rejected by all classifiers. In contrast, several classifiers might claim an instance and cause a conflict, and the instance has to be rejected in this case too. When rejection is not an option, these rejected instances should still be classified. To solve this problem, we build a classifier out of multiple multi-class LLS classifiers using a cascade fusion strategy based on a degenerate decision tree [24,25], as shown in Figure 3.

At each step, we train one multi-class LLS classifier for multiple classes using different subsets of the feature set of all instances in the training dataset using the following:

$$B = \{B_s | s = 1 \dots n\},\ B_s = \{b_s\},$$
$$\text{where } b_s = \{s_{th} \text{ downward projection of } Y \text{ in a fixed order}\} \tag{11}$$

where Y is the set of all instances in the training set D and s is the size of the feature subsets. This results in creating a sequence of multi-class LLS models.

Table 1. Datasets Used

Data set	♯Features	♯Classes	♯Cases
Pima	9	2	768
Ecoli	8	7	336
Glass	19	6	214
Icono'	35	2	351
Iris	4	3	150
Liver	7	2	345
Segment'	20	7	1500
Wine	14	3	178
Wis-BC-D	31	2	569
Wis-BC-P	34	2	198
Yeast	9	10	1484

Table 2. Classifiers for Comparison

Classifier	Description
Naive Bayes	
Bayes Net	
kNN	Inverse Distance Weighted
C4.5	Decision Tree
C4.4	C4.5 no pruning, Laplace smoothing
NBTree	
SVM	polynomial kernel
RBF Net	Radial basis function network

We treat the base classifiers as experts having subject-specific knowledge, whose contributions to the final decision are equally weighted. We employ the unanimity rule of decision making, except in the final step of cascade, where voting and ranking are used. Since different experts for the same target are trained by different feature subsets, the unanimity rule makes confident decisions at the beginning of the cascade, and defers more difficult decisions to later stages, when more training information is available.

The model in each step chooses its confident decision area to make a decision against classification and leaves instances in its dilemma area to be resolved by the models in subsequent steps. That is, b_ss are subsequently applied to each test instance until it gets accepted by one of them or until it misses them all. This is because each feature might strongly predict instances with certain characteristics and be weak for others. By creating a variety of models trained by different feature sets, each model works as an expert which only makes a decision in the area where it is confident. In the case of an instance missing all models, ranking and voting is employed to assign them a class, which is done by summing the class probabilities in each model and picking the class with highest total probability. The proposed algorithm is depicted in Algorithm 1.

In this way, most of the instances can be accepted at the beginning stages and therefore most of the classification time is spent on instances that are hard to classify. This means that the system is focusing on the hard-to-classify instances, which agrees with human decision making behaviours and can help to improve the overall performance of the classifier. Another advantage is that each of the stages need not be perfect; in fact, the stages are usually biased towards small false-alarm rate rather than towards high hit-rates.

4 Experiments

To test the performance of the proposed system on real world data, we implemented LLS in Matlab and tested it using datasets from the UCI Repository [26]. We utilised the Weka framework [27] for evaluation of the algorithms.

4.1 Setup

We chose 11 datasets with numerical attributes alone, shown in Table 1 from the UCI repository, since LLS currently handles only numerical attributes. We compared LLS with 8 standard classifiers, that covers most categories of state-of-the-art machine learning algorithms and listed in Table 2.

Due to the cascade strategy, LLS prefers to use the most discriminating features at the beginning of the cascade procedure. Therefore, we employ Principal Component Analysis (PCA) to preprocess the datasets and choose the principal component with large variance first. For visualisation purposes, the feature values are re-scaled to be in the range between 0 and 255 after PCA processing and we only use two features in each base LLS in the experiments by making an assumption that a class can be described well by a feature subset of size 2. We utilise binning to reduce noise and improve computational efficiency by empirically assigning 30 bins to each dimension of the feature space.

4.2 Results and Discussion

We evaluated the classifiers on the datasets by conducting 10-fold cross-validation, and the resulting prediction accuracy is depicted in Table 3, where the outputs of the paired t-test at 95% confidence level with LLS as the base classifier are also shown. The standard deviations are shown in brackets. Under LLS column, accuracies better than 80% appear in bold.

To compare the performance of algorithms under a variety of configurations, we calculated the values of AUC, the area under the ROC (Receiver Operating Characteristics) curve, shown in Table 4. We also show the Mean Square Error (MSE) which is another commonly used evaluation metric in Table 5. The classifiers are ordered left to right in decreasing order, based on their average measures. The bold entries in Tables 4 and 5 are LLS values better than average.

LLS are competitive with most of state-of-the-art machine learning algorithms besides exploiting a novel direction of classifier construction that is not based on any existing machine learning algorithm. From the AUC and MSE scores in Table 4 and 5, it is clear that LLS outperforms at least half of the other classifiers and is very competitive with the rest. The prediction accuracy measures in Table 3 shows that LLS outperforms all other classifiers on dataset Iris. It also outperforms about half of other classifiers on Liver, Wis-BC-D and Ecoli. Although it appears that some other classifiers produce a better prediction accuracy than LLS, the paired t-test shows that LLS is statistically competitive with most classifiers on most datasets except Segmentation and Yeast, where instances from different classes are mixed together, with very confusing boundaries. For these datasets, the use of binning with fixed size may have caused instances from different classes to be collected into the same bins and confused the boundary construction.

Table 3. Prediction Accuracy ('v' - better, '*' - worse than LLS)

Data Set	NB Tree	RBF Net	C4.5	C4.4	SVM	Bayes Net	kNN	Naive Bayes	**LLS**	LLS Rank
iris	94.00	95.33	96.00	96.00	96.00	92.67	95.33	96.00	**96.67**	**1**
	(4.92)	(4.50)	(5.62)	(5.62)	(4.66)	(6.63)	(5.49)	(4.66)	(4.71)	
liver	66.13	64.35	68.71	68.99	58.28	56.25	58.25	55.39	62.61	**5**
	(8.31)	(6.86)	(8.74)	(6.80)	(1.48)	(3.65)	(6.39)	(8.86)	(9.60)	
wis-d'	92.79	94.21	93.15	92.80	97.72	95.08	96.32	92.98	**93.50**	**5**
	(2.40)	(3.61)	(3.64)	(3.91)	(1.66)v	(2.58)	(2.92)	(4.30)	(3.10)	
ecoli	80.94	83.63	81.55	80.07	84.82	79.14	84.80	85.40	**81.2**4	**6**
	(5.12)	(4.88)	(7.61)	(8.36)	(5.52)v	(6.17)	(4.4)v	(5.87)	(3.22)	
glass	75.22	65.50	65.87	66.32	57.51	74.76	70.11	49.48	61.23	7
	(9.63)v	(9.33)	(8.91)	(8.37)	(8.22)	(6.26)v	(9.72)v	(9.02)	(7.88)	
iono'	89.73	92.62	91.46	91.46	88.60	89.46	82.62	82.62	**86.89**	7
	(3.64)	(5.66)	(3.27)	(3.27)	(4.26)	(4.47)	(3.43)	(5.47)*	(2.86)	
wine	96.63	98.30	93.86	93.86	98.33	98.89	95.00	96.63	**94.97**	7
	(2.90)	(2.74)	(5.52)	(5.52)	(2.68)v	(2.34)v	(4.86)	(5.38)	(4.11)	
wis-p'	73.24	77.79	75.74	74.74	76.29	74.79	77.34	67.16	73.76	8
	(4.68)	(6.45)	(8.84)	(7.51)	(3.25)	(4.48)	(6.58)	(11.39)	(7.36)	
segm'	94.33	86.93	95.73	95.73	91.93	90.40	94.80	81.07	**85.60**	8
	(2.31)v	(3.03)	(0.72)v	(0.90)v	(2.40)v	(2.18)v	(1.66)v	(2.33)*	(2.74)	
pima	74.36	75.40	73.83	72.67	77.34	74.36	72.14	76.31	70.06	9
	(6.68)	(4.36)	(5.66)	(6.64)	(4.07)v	(4.71)	(4.36)	(5.52)v	(4.07)	
yeast	56.80	59.16	55.99	54.78	57.08	56.74	55.19	57.61	50.95	9
	(3.55)v	(4.71)v	(4.77)v	(4.85)	(4.10)v	(3.79)v	(2.53)v	(3.01)v	(3.32)	

Table 4. Area under ROC curve ('v' - better, '*' - worse than LLS)

Dataset	Naive Bayes	Bayes Net	RBF Network	**LLS**	kNN	C4.4	NB Tree	C4.5	SVM
ecoli	0.99	0.99	0.98	**0.95**	0.98	0.97	0.97	0.95	0.97
glass	0.73*	0.91	0.82	0.82	0.88	0.82	0.89	0.79	0.77
iris	1.00	1.00	1.00	**1.00**	1.00	0.99	1.00	0.99	1.00
ionosphere	0.94	0.95	0.96	**0.97**	0.90*	0.92*	0.91	0.89*	0.85*
liver	0.65	0.52*	0.67	0.62	0.65	0.69	0.66	0.67	0.5*
pima	0.82v	0.81v	0.79	0.72	0.75	0.77v	0.79	0.75	0.72
wis-bc-d	0.98	0.99v	0.97	**0.96**	0.99v	0.97	0.95	0.93	0.97
wis-bc-p	0.66	0.68	0.61	0.66	0.57	0.66	0.54	0.61	0.52
wine	1.00	1.00	0.99	**0.99**	1.00	0.97	1.00	0.98	1.00
yeast	0.75	0.76	0.78	0.73	0.72	0.74	0.76	0.70	0.72
segment	0.99	1.00	1.00	**0.99**	1.00	1.00	0.98	0.98	1.00
Average	0.87	0.87	0.87	**0.86**	0.86	0.86	0.86	0.84	0.82

All the above performances are achieved with a generic speed function. On a specific application, a more sophisticated speed function specially constructed with embedded domain knowledge may lead to even better performance.

Table 5. Mean Square Error ('v' - better, '*' - worse than LLS)

Dataset	Bayes Net	RBF Network	kNN	NB Tree	**LLS**	C4.4	C4.5	Naive Bayes	SVM
ecoli	0.19	0.20	0.18v	0.20	**0.21**	0.21	0.21	0.17v	0.31*
glass	0.24v	0.27	0.24v	0.23v	0.29	0.28	0.29	0.33	0.32
iris	0.15	0.13	0.13	0.14	**0.06**	0.13	0.11	0.13	0.29*
ionosphere	0.31	0.25	0.34*	0.30	**0.26**	0.27	0.28	0.39*	0.33*
liver	0.5	0.48	0.49	0.48	0.49	0.48	0.50	0.51	0.65*
pima	0.42	0.42	0.44	0.43	0.45	0.44	0.44	0.41	0.47
wis-bc-d	0.21	0.21	0.16v	0.25	**0.24**	0.23	0.25	0.24	0.13v
wis-bc-p	0.41	0.42	0.45	0.45	0.46	0.44	0.46	0.52	0.49
wine	0.06v	0.06	0.14	0.11	**0.15**	0.18	0.17	0.09	0.28*
yeast	0.24v	0.24v	0.25	0.24v	**0.26**	0.27*	0.27	0.24v	0.28*
segment	0.15v	0.16v	0.10v	0.12v	**0.20**	0.11v	0.11v	0.23*	0.30*
Average	0.26	0.26	0.27	0.27	**0.28**	0.28	0.28	0.30	0.35

5 Concluding Remarks

LLS has certain advantages over other classifiers. Firstly, it has an efficient knowledge representation where the knowledge learned is represented as the level learning set isocontours in the feature space. This makes the testing very efficient by requiring only a lookup of the class indicator value in the learned indicator function for each feature vector. Secondly, the proposed classifier constructs decision boundaries directly, thereby avoiding the difficulty of determining a correct parametric density model and its parameter values. In addition, the learned level set function can be directly used as the initialisation of an LLS training procedure for new instances, which makes it suitable for incremental learning.

Acknowledgement

National ICT Australia is funded through the Australian Government's *Backing Australia's Ability* initiative, in part through the Australian Research Council. The authors also thank Dr Mike Bain, School of Computer Science and Engineering, UNSW for asking the 'what if' question.

References

1. Braverman, D.: Learning filers for optimum pattern recognition. IRE Transactions on Information Theory IT-8, 280–285 (1962)
2. Baum, L.E., Petrie, T.: Statistical inference for probabilistic functions of finite state markov chains. Annals of Mathematical Statistics 37, 1554–1563 (1966)
3. Fix, E., Hodges, J.L.: Discriminatory analysis - nonparametric discrimination: Consistency properties. USAF School of Aviation medicine 4, 261–279 (1951)
4. Highleyman, W.H.: Linear decision functions with application ot pattern recognition. Processings of the IRE 50, 1501–1514 (1962)

5. McCulloch, W.S., Pitts, W.: A logical calculus of ideas imminent in nervous activity. Nulletin of Mathematical Biophysics 5(115-133) (1943)
6. Boser, B.E., Guyon, I., Vapnik, B.: A training algorithm for optimal margin classifers. In: Haussler, D. (ed.) Processings of the 4th Workshop on Computational Learning Theory, San Mateo, CA, pp. 144–152. ACM Press, New York (1992)
7. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. Chapman and Hall, New York (1993)
8. Quinlan, J.R.: C4.5 Programs for machine Learning. Morgan Kaufmann, San Francisco, CA (1993)
9. Yip, A.M., Ding, C., Chan, T.F.: Dynamic cluster formation using level set methods. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(6), 877–889 (2006)
10. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: Active contour models. International Journal of Computer Vision, 321–331 (1988)
11. Cohen, L.: On active contour models and balloons. CVGIP Image Understanding 53 (1991)
12. Caselles, V., Kimmel, R., Sapiro, G.: Geodesic active contours. In: ICCV'95, Cambridge, USA, pp. 694–699 (1995)
13. Cohen, L.D., Kimmel, R.: Global minimum for active contour models: A minimal path approach. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 666–673. IEEE Computer Society Press, Los Alamitos (1996)
14. Malladi, R., Sethian, J.A., Vemuri, B.C.: Shape modeling with front propagation: a level set approach. IEEE Transactions on Pattern Analysis and Machine Intelligence 17(2) (1995)
15. Geman, D., Jedynak, B.: An active testing model for tracking roads in satellite images. IEEE Trans. Pattern Anal. Machine Intell. 18(1) (1996)
16. Chan, T.F., Vese, L.A.: Active contours without edges. IEEE Transactions on Image Processing 10(2), 266–277 (2001)
17. Cai, X., Sowmya, A., Trinder, J.: Learning parameter tuning for object extraction. In: Narayanan, P.J., Nayar, S.K., Shum, H.-Y. (eds.) ACCV 2006. LNCS, vol. 3851, pp. 868–877. Springer, Heidelberg (2006)
18. Cai, X., Sowmya, A.: Active contour with neural networks-based information fusion kernel. In: King, I., Wang, J., Chan, L., Wang, D. (eds.) ICONIP 2006. LNCS, vol. 4233, pp. 324–333. Springer, Heidelberg (2006)
19. Tax, D.: One-class classification. PhD thesis, Delft University of Technology (2001)
20. Evans, L.: Partial Differential Equations. American Mathematical Society (2002)
21. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D 60, 259–268 (1992)
22. Aubert, G., Vese, L.: A variational method in image recovery. SIAM Journal on Numerical Analysis 34(5), 1948–1979 (1997)
23. Osher, S.J., Fedkiw, R.P.: Level Set Methods and Dynamic Implicit Surfaces. Springer, Heidelberg (2003)
24. Amit, Y., Geman, D., Wilder, K.: Joint induction of shape features and tree classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence 19(11), 1300–1305 (1997)
25. Viola, P., Jones, M.J.: Rapid object detection using a boosted cascade of simple features. In: IEEE CVPR, pp. 511–518. IEEE Computer Society Press, Los Alamitos (2001)
26. Newman, D., Hettich, S., Blake, C., Merz, C.: Uci repository of machine learning databases (1998) `http://www.ics.uci.edu/~mlearn/mlrepository.html`
27. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)