Submitted to the British Machine Vision Conference 2004

Unsupervised Clustering, Feature Selection and Attention for Autonomous Visual Learning

Abstract

A framework for fully autonomous (unsupervised) learning of object models in dynamic scenarios from video is presented. The framework consists of a spatio-temporal visual attention module, a multiple (rotationally invariant) texture feature generation module, an unsupervised clustering/feature selection module and a model building module. The clustering/feature selection algorithm presented extends the cluster-based similarity partitioning algorithm of Strehl and Ghosh [12] to allow feature selection as part of the clustering process. This is particularly powerful when working with a high dimensional feature description where many dimensions are irrelevant to a particular classification task, as in autonomous learning scenarios. Rotational invariance in textural features is obtained by using statistics derived from the application of banks of differently orientated directional filters as features, rather than the raw outputs of these filters. In addition a novel temporal attention mechanism is presented. The system has been initially applied to the learning of models of cards and dice from video of tabletop games, but should extend to a wider range of scenarios.

1 Introduction

This work forms part of a larger system to learn object and event descriptions from the real world in a completely unsupervised manner. We would eventually like to point a camera at an arbitrary scene containing unknown objects, and learn models of both object appearance and behaviour in an unsupervised way. This paper deals with learning (non-generative) object appearance models by observation from a dynamic scenario, which may be used to label novel instances of these unknown object classes in the same (or similar) scenarios. As an example we have evaluated our methods using a simple tabletop game playing scenario, where games involving cards and dice are played in the view of a camera connected to a computer (figure 1).

By far the majority of literature on object learning in machine vision relates to supervised learning methods (see for example [6] and associated references). By supervised we mean a set of class labelled images, or image patches, is presented to a learning algorithm to form a classifier that is capable of detecting and/or classifying novel instances of these classes (or class). This is not a practical approach for an autonomous agent (biological or synthetic), which must learn object structure in an unsupervised, emergent way. In recent research, Duygulu *et al.* [7] use the correlation between textual descriptions and the properties of image patches to form models in a ‘semi-supervised’ manner. However, this is not globally applicable, as such textual descriptions do not always exist. The largest

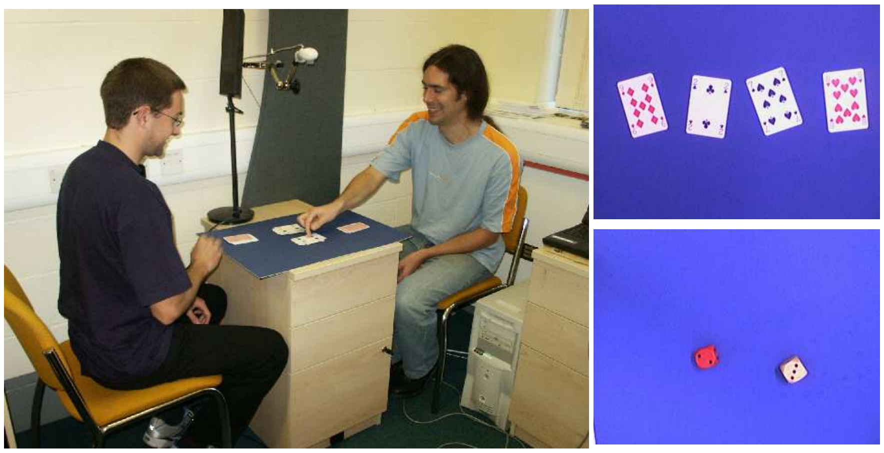


Figure 1: Experimental setup (left) and typical image data (right)

body of research on unsupervised learning appears to be in environmental (rather than object) learning, within the mobile robotics community (e.g. [10]). Unsupervised learning is used to build environmental maps from visual features (or from non-visual data such as sonar). The literature on fully autonomous object learning is rather more sparse. Petrov [8] describes a system which performs unsupervised classification of whole images (containing single objects) using banks of wavelets and a 2 layer self-organising network based on Kohonen maps. This bears some similarity to parts of our system, however this does not deal with multiple objects or objects that are situated in a real world scenario. Not wanting to re-invent the wheel we propose to re-use methods from the large body of work on supervised object learning in the context of unsupervised learning. For unsupervised learning, training examples must be partitioned (clustered) into related groups. These cluster labellings can then be used as a form of supervision for supervised object learning methods. In this paper we present a novel method for performing clustering and feature selection, and evaluate it against a number of standard clustering approaches.

We use a spatio-temporal attention mechanism, based on an existing blob tracker ([13], an extension of [11]), to identify the positions of 'interesting' objects over time, and select key-frames, from which feature vectors for each object are extracted. These feature vectors are the composite output of a number of different algorithms. Features used in this paper are texture (convolution kernel) based, however we plan to extend the approach to other methods using colour, and local and global shape descriptors. A multiple-clustering approach (an extension of work by Strehl and Ghosh [12]) is used to cluster observed examples into perceptually separate classes, and the output of this used as supervision for a model building (learning) algorithm. This allows the output of the clustering to be applied to unseen data. Our clustering technique allows the selection of a subset of 'significant' features from a large feature set that contains a high proportion of insignificant features (with respect to a particular application domain). This negates the need for manual selection of appropriate features for a particular task *a priori*. An overview of our approach is shown in figure 2.

The rest of this paper is divided into sections relating to the elements shown in figure 2. Section 2 deals with the blob tracker/attention mechanism, section 3 deals with the

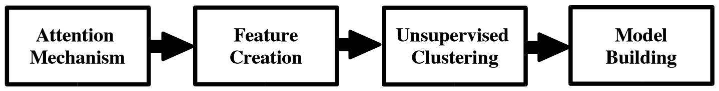


Figure 2: Overview of the Autonomous Learning System

methods used to create the composite feature space used, section 4 deals with the novel clustering combination method used for clustering and feature selection, and section 5 deals with evaluating various machine learning algorithms that may be used within our system for model building.

2 Spatio-temporal Attention Mechanism

An existing blob tracker [13] (an extension of the system described by Stauffer and Grimson [11]) is used to locate and track objects within the scene of interest. The tracker is based around a multi-modal (Gaussian Mixture) per-pixel background model. Pixels classified as foreground by this model are associated with object models, based on position and colour information. The output of the tracker is a centroid position and image mask for each object tracked.

A simple temporal attention mechanism for the detection of significant key-frames is used in the experiments in this paper (although there is ongoing work into more general mechanisms). This works by detecting sequences of frames where no motion occurs (i.e. the change in all object centroids and sizes is below a threshold) for N frames (where N is typically 3 for a 10fps sequence). A single image (key-frame) is then passed on to the remainder of the learning system (see figure 2). The attention mechanism then requires there to be motion for a number of frames (typically 3) before another key-frame is detected. Qualitative output of the temporal attention mechanism is shown in figure 3 (key-frames are shown in bold squares. N.B. There are actually several frames between each key-frame, although only one example is shown).

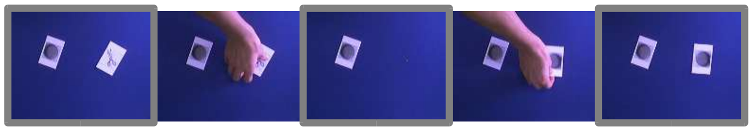


Figure 3: Output of Attention Mechanism

3 Rotationally Invariant Feature Descriptions

In our experiments, the primary feature set used was based on Gabor wavelet [2] and other convolution based textural descriptions (figure 4), although a wide range of feature descriptions based on texture, shape (local and global) or colour could have been used.

These were applied at the object centroid provided by the object tracker, to form a 94 dimensional feature vector. To make these features rotationally invariant (where they were not already) sets of features at multiple orientations were applied to the images

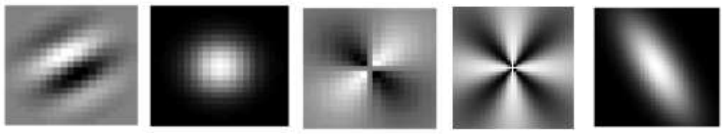


Figure 4: Example Wavelet and Convolution Based Textural Feature Descriptors Used

and the statistics (mean, minimum, maximum and maximum/mean) of these feature subsets used as the feature description, rather than the raw values. The features used were; Gabor wavelets (various scales), equal covariance Gaussians (various scales), non-equal covariance Gaussians (various scales) and polar wavelet-like features of the form:

$$K(r,\theta) = G(r,\sigma) sin(K_1\theta + K_2) \tag{1}$$

where $G(r)$ is a Gaussian distribution with standard deviation σ, K_1 relates to angular frequency ($K_1 = 1,2$ or 4 in our experiments) and K_2 relates to the filter orientation. Convolution kernels of the form in equation 1 were applied at various scales and angular frequencies. The scales chosen for all feature descriptor types were selected as (approximately) evenly distributed between sensible limits, so as to be as general as possible.

It should be noted that our method could have been made scale independent by simply scaling the image to a standard size, based on the object size provided by the object tracker. However, this was not necessary for the class of scenarios we are currently interested in.

4 Forming Natural Clusters Using Weighted Combinations of Feature Clusterings

There are a number of methods in the literature for unsupervised clustering such as K-means [3], agglomerative methods [9, 1] and graph partitioning based methods [5]. Such methods generally work by associating similar data items in an N-Dimensional feature space description. All features are used in this clustering process and feature selection is usually seen as a supervised pre-processing step (if performed at all). An alternative approach, taken by us, is to combine the results of multiple clusterings (either multiple methods on a single feature set or a single method on multiple data sets) [12]. We take as our basic sub-space clustering algorithm an agglomerative clustering algorithm based loosely on that presented in [1]. This works as follows:

1. Normalise each data dimension to zero mean unit standard deviation
2. Initialise one cluster per data item
3. Calculate the mean Euclidean distance between each pair of clusters: $D(C_n,C_m) = \frac{1}{N_n N_m} \sum_{c_a \in C_n, c_b \in C_m} |c_a - c_b|$
4. if $min(D(C_n,C_m)) < T$ (where T is a fixed threshold), merge the corresponding cluster pair and repeat from 3

We use $T = K\sqrt{d}$, where d is the dimensionality of the subspace and K is a constant (typically 1). Agglomerative clustering is performed on multiple low-dimensional subspaces of the full feature description (all 1D subspaces in our experiments). The results of these clusterings are combined using a novel weighted version of the Cluster-based Similarity Partitioning Algorithm of Strehl and Ghosh [12]. In the original algorithm a (non-directional, fully connected) weighted graph is formed where the vertices (unweighted) represent data items and the (weighted) edges represent the similarity between pairs of data items. Similarity is measured as the number of times the pair of data items occurs in the same cluster over all sub-space clusterings. This graph is then partitioned using a graph partitioning algorithm such as [5] (implementations of several graph partitioning algorithms are freely available to download from *http://www-users.cs.umn.edu/ karypis*, as part of the METIS and CLUTO packages). Such algorithms attempt to form a set of sub-graphs, such that the edge cut required is minimised. This is an NP complete problem, however there are a number of suitable approximate methods at our disposal.

We extend the method in [12] to form a graph in which the edge weights are the sum of weights associated with the low dimensional (1D) clusterings produced using the agglomerative clustering algorithm (equation 2). In this way the relative importance of the individual features in our feature description can be weighted.

$$EdgeWeight(V_a, V_b) = \frac{\sum_{c=1}^{N_c} W_c S_c(V_a, V_b)}{\sum_{c=1}^{N_c} W_c} \tag{2}$$

where W_c is the weight associated with clustering/feature c and $S_c(V_a, V_b)$ is 1 if the data items relating to vertices V_a and V_b are contained within the same cluster for clustering c, and 0 otherwise. N_c is the number of clusterings used. It should be noted that the denominator is simply a constant and, as such, has no effect on the clustering produced. However, the inclusion of this normalisation enables edge weights (and thus edge cuts) to be compared across different feature sets and weightings.

4.1 Subspace Weighting

Once the initial clustering has been performed (with equal weights) the discriminative power of the individual sub-spaces used to build the clustering can be evaluated with respect to this clustering (i.e. assuming this clustering is "correct"). We take the view that if a subspace can discriminate well between any two classes then its discrimination power is good, as it is unusual for a low dimensional subspace (1D in our experiments) to be able to discriminate between all classes. We consider the accuracy of an optimal two class stochastic classifier based on the mapping between subspace clusters and the clusters formed using the graph partitioning method (assumed to be the "correct" class labelling). Figure 5 illustrates the formulation of such a classifier.

From figure 5 it is easy to see that the overall accuracy of an optimal two class stochastic classifier based on a particular low dimensional cluster set $\mathscr{C} = C_1, C_2, ..C_n$ would be:

$$P_{Correct}(A, B, \mathscr{C}) = \sum_{i=A,B} \frac{P(i)}{P(A) + P(B)} \sum_{j=C_1,C_2,..C_n} P(j|i)P(i|j) \tag{3}$$

where $P(A)$, $P(B)$, $P(j|i)$ and $P(i|j)$ can be calculated from a co-occurrence frequency matrix of clusters formed from the graph partitioning vs. sub-space clusters for

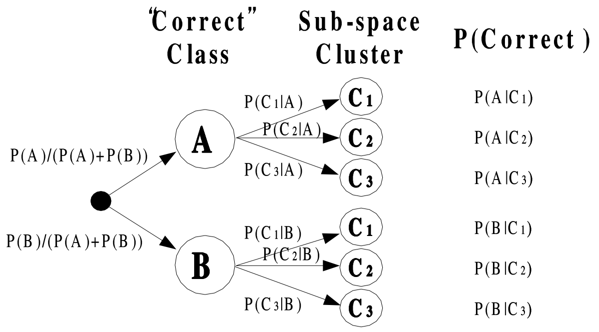


Figure 5: Formulation of an optimal stochastic classifier for two class discrimination

the training data. Thus we define discriminative power ($D(\mathscr{C})$) as the maximum value of $P_{Correct}(A,B)$ over all pairs of classes, as in equation 4.

$$D(\mathscr{C}) = \max_{a,b=A,B,C,\ldots,a!=b} P_{Correct}(a,b,\mathscr{C}) \tag{4}$$

$D(\mathscr{C})$ lies between 0.5 (no discriminative power) and 1.0 (perfect discrimination for at least one pair of classes). This value can be used to weight the graph partitioning based clustering described in section 4, however a better approach is to threshold this value and use binary weights (we use a fairly low threshold of 0.6 to exclude only those sub-spaces that are very poorly discriminative). Essentially this is an unsupervised feature selection approach.

In principle our method could be applied iteratively by repeatedly using the clustering output to calculate new weights and re-clustering. Our experiments suggest that improved performance and convergence occurs in some circumstances, however in other circumstances over-fitting and other stability problems have been observed. We recommend only a single application of the re-weighting for this reason. It should be noted that our approach relies on the initial clustering having a (more or less) one-to-one mapping with the true class labelling (i.e at least 50% of items in a cluster belong to the same true class). If this is not the case it is unlikely an improvement will be obtained.

4.2 Evaluation of Clustering Performance

We evaluated the performance of our proposed clustering-combination method against a range of standard clustering methods, and the original clustering-combination method of Strehl and Ghosh [12], on which our method is based. With no modifications of code or feature descriptors the system was applied to three data set types, one of a dice game (6 sides/classes), one of a card game (standard playing cards with picture cards and aces were removed to leave 9 classes), and one of a card game based on the game paper-scissors-stone (3 classes). The training data sets consisted of around 5-10 minutes of video of each scenario (approximately 120-220 examples in each). To compare results the concept of 'Non-maximal association' (NMA) is introduced. If each true class is mapped to a single cluster (by largest unique association[1]), the non-maximal association for each class is

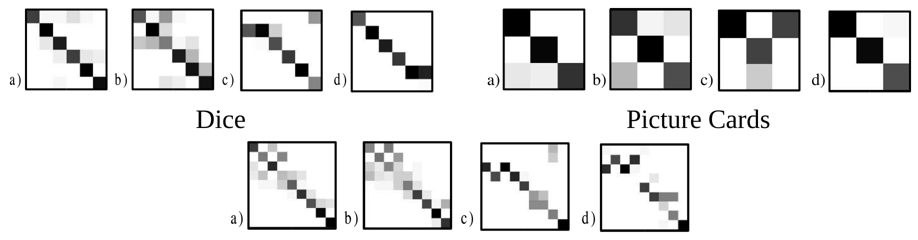
[1]If there is a conflict in this mapping the association that minimises the mean Non-maximal association is used

the proportion of examples of the class that are not included in this cluster. Ideally this should be zero for each class. We present results for mean and maximum non-maximal association. Results are given in table 1 below.

	Dice NMA Mean (Max)	Picture Cards NMA Mean (Max)	Playing Cards NMA Mean (Max)
Our Method	0.09 (0.14)	0.066 (0.152)	0.26 (0.71)
Strehl and Ghosh [12]	0.29 (0.56)	0.151 (0.261)	0.36 (0.79)
K-means	0.29 (1.00)	0.350 (1.00)	0.380 (1.00)
Agglomerative clustering	0.160 (0.97)	0.020 (0.059)	0.380 (1.00)

Table 1: Comparative Evaluation of Clustering Algorithms

For a qualitative evaluation, class-cluster association matrices are given in figure 6. These are analogous to confusion matrices in supervised classification problems, with the true class along the horizontal axis and the clusters along the vertical axis. The clusters have been re-arranged manually so the maximal associations lie along the diagonal for ease of visualisation.



Playing Cards

Figure 6: Association Matrices for a) Our Method, b) Strehl and Ghosh's Original Method, c) K-means clustering, and d) Agglomerative clustering

4.3 Automatic Selection of the Number of Clusters

The observant reader will have noticed that the evaluation in the previous section is based on the correct number of clusters (in terms of the number of true classes) being formed. This is hand specified for the purposes of the evaluation, however our system actually automatically estimates the number of clusters. Unsurprisingly, we were unable to come up with a metric to estimate exactly the number of clusters for an arbitrary data set. Our alternative approach is to detect the occurrence of over-clustering (i.e. when too many clusters are selected). Considering our discriminative power metric ($D(\mathscr{C})$, equation 4), a range of values between 0.5 and 1 are expected over all features used, for a correct clustering . When over-clustering occurs, values of $D(\mathscr{C})$ for non-discriminative features rise erroneously towards 1. This is detectable by a drop in the standard deviation of $D(\mathscr{C})$ over all features. One method for cluster selection would be to select the number of clusters corresponding to the maximum value of the standard deviation of $D(\mathscr{C})$ over all features. This gives the correct number of clusters for the picture cards data set (3), and the dice data set (6), however too few clusters are selected for the playing cards data set. For the sort of temporal sequence learning methods that the work in this paper will be used with, over-clustering is preferable to under-clustering, as cluster equivalences are learned in a spatio-temporal context. We therefore chose a method that tends to over-cluster data sets. The proportion of features where $D(\mathscr{C})$ is near 1 (we use a threshold of 0.99 on $D(\mathscr{C})$) is a good indicator of over clustering. When an upper threshold of 0.5 on this proportion is used to detect over-clustering, the number of clusters detected are given in table 2.

Data Set	Over-clustering Occurs @	Clusters Selected	Actual Classes
Dice	11	10	6
Picture Cards	5	4	3
Playing Cards	10	9	9

Table 2: Clusters selected Using Automatic Selection Method

5 Model Building

There are numerous standard supervised machine learning algorithms that could be used for our task. Thus, we feel our decision to simply evaluate a number of these methods using cluster membership (rather than hand labelling) as supervision is fully justified. The purpose of building a model is twofold; i) To allow the clustering applied to the training data to be applied to novel, unseen data, and ii) to generalise the description of the various classes to remove noise/outliers (including mis-clustered examples). We have evaluated the performance of the following algorithms on the original and unseen data with respect to this; C4.5 Decision tree analysis [4], a Radial Basis Function classifier (RBF), a Vector Quantisation based Nearest Neighbour classifier (VQ-NN), a Multi-layer perceptron (MLP), and a K-nearest neighbour classifier (KNN). Implementations of all methods, except C4.5, were taken from the freely available lnknet package (*http://www.ll.mit.edu/IST/lnknet/*). C4.5 is also freely available on the world wide web (*http://www.cse.unsw.edu.au/ quinlan/*)

5.1 Evaluation of Model Building Algorithms

The results of the model building algorithms were applied to unseen data, to show generalisation potential from partially erroneously labelled data (table 3). The algorithm parameters were left at package default values (listed in tables), as these were sensible. In addition, an extra test for the vector quantisation method with 50 prototypes was carried out to illustrate the effect of increasing model complexity. Results are presented in (approximate) order of suitability for the task at hand (best to worst).

It is interesting to note the simpler methods (i.e those with fewer degrees of freedom), such as the Radial Basis Function classifier and the vector quantisation based nearest neighbour classifier, come out the best in this evaluation. In particular the RBF classifier outperforms all other methods on the two higher dimensional data sets (and is joint best on the other). The reason for this is obvious, as methods with fewer degrees of freedom will not over-fit the training data as much. As our training data is the result of an unsupervised clustering it is contains significant noise (mis-clusterings). The simpler methods

	Dice	Picture Cards	Playing Cards
	NMA Mean (Max)	NMA Mean (Max)	NMA Mean (Max)
RBF (16 Kernels)	0 (0)	0.008 (0.025)	0.20 (0.71)
VQ-NN (16 Prototypes)	0.03 (0.18)	0.008 (0.025)	0.21 (1.0)
VQ-NN (50 Prototypes)	0.05 (0.18)	0.008 (0.025)	0.33 (0.67)
MLP (25 hidden nodes in 1 layer)	0.05 (0.24)	0.008 (0.025)	0.35 (0.71)
KNN (K=1)	0.07 (0.24)	0.008 (0.025)	0.35 (0.67)
C4.5	0.13 (0.38)	0.017 (0.025)	0.42 (0.67)

Table 3: Evaluation of Various Model Building Algorithms for Unseen Data

generalise out this noise, the more complex ones fit to it. This effect is particularly well illustrated by the fact that the performance of the vector quantisation based nearest neighbour classifier drops when the number of prototypes is increased above the default (16). In particular C4.5 performs poorly as there is no built in redundancy in its decision tree approach, so it can generalise out good rules and keep poorer ones if data support is locally sparse.

The relatively high values of Maximum (and to a lesser extent mean) Non-maximal association for all methods on the playing cards data are a result of the representations for ‘threes’ and ‘fives’ being very similar in our feature space (only one or two dimensions can separate them). This is illustrated particularly by the vector quantisation based nearest neighbour classifier (16 prototypes) on unseen data. All instance of ‘threes’ and ‘fives’ are classified as the same class by this classifier, and there is an ‘empty’ class to which no data items are classified. We propose that the separation of these two classes will require an extension to our feature space, or the use of temporal, contextural or functional information (there is a discussion of this in the next section). All other classes are fairly well separated.

6 Discussion and Future Work

A system for fully autonomous (unsupervised) class learning of dynamic objects from video is presented in this paper. This involves the use of a spatio-temporal attention mechanism, developed from an existing object (blob) tracker. Rotationally invariant textural feature descriptions of each tracked object are built up from the statistics of the outputs of a relatively large number of convolution based image processing operators. This feature description is non-application specific and, as such, contains a large proportion of irrelevant dimensions. The combination of multiple clusterings in a novel, graph partitioning based, formulation serves to perform unsupervised classification (class building) and feature selection. These clusterings are used to supervise a range of model building (learning) algorithms in order for unseen objects to be recognised. Evaluation of the novel clustering method demonstrated it generally outperformed a number of existing techniques on this type of data. An interesting feature of the model building algorithm evaluation was that the simpler methods came off better. This can be explained by the outlier rejection that is inherent in models with fewer degrees of freedom. In particular the Radial Basis Function classifier proved extremely proficient at noise rejection.

It was possible to learn near perfect models for the dice and picture card data sets from spatial information alone, however the models built for the playing cards (although reasonably good) still leave a little room of improvement. A related part of our research (submitted separately) is to learn temporal patterns and protocols using Inductive Logic Programming. During this process, cluster equivalences may be learned from over-clustered data. In the future it is hoped that labelling of outliers in this learning process could be fed back into the spatial learning to improve the model. The graph partitioning approach to clustering lends itself well to the addition of this additional data, as it is simply a matter of re-weighting the similarity graph edges in an appropriate manner.

7 Acknowledgements

This work was funded by the European Union, as part of the ‘XXX’ project.

References

[1] S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *Proc. European Conference on Computer Vision*, volume 4, pages 113–127, 2002.

[2] J. Daugman. Uncertainty relation for resolution in space, spatiol frequency, and orientation optimised by two-dimensional visual cortical filters. *Journal of the Optical Society of America*, 2(7):1160–1169, 1985.

[3] E. Forgy. Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. *Biometrics*, 21:768, 1965.

[4] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[5] G. Karypis and V. Kumar. Multilevel graph partitioning schemes. In *Proc. International Conference on Parallel Processing*, volume 3, pages 113–122, 1995.

[6] B. Leibe and B. Schiele. Analyzing contour and appearance based methods for object categorization. In *Proc. International Conference on Computer Vision and Pattern Recognition*, volume 2, pages 409–415, 2003.

[7] J. de Freitas P. Duygulu, K. Barnard and D. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Proc. European Conference on Computer Vision*, volume 4, pages 97–112, 2002.

[8] N. Petkov. Image classification system based on cortical representations and unsupervised neural network learning. In *Proc. IEEE Workshop on Computer Architectures for Machine Perception*, pages 430–437, 1995.

[9] H. Shin and C. Kim. A simple yet effective technique for partitioning. *IEEE Transactions on VLSI Systems*, 1(3):380–386, 1993.

[10] R. Sim and G. Dudek. Mobile robot localisation from learned landmarks. In *Proc. IEEE/RSJ Conf. on Intelligent Robots and Systems*, pages 1060–1065, 1998.

[11] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. In *Proc. Computer Vision and Pattern Recognition*, pages 246–252, 1999.

[12] A. Strehl and J. Ghosh. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, 2002.

[13] X. Xxxxx. An interesting paper by one of the authors. *XXXXX*, XXXX.