An Ensemble of Deep Support Vector Machines for Image Categorization

Azizi Abdullah, Remco C. Veltkamp
Department of Information and Computer Sciences
Utrecht University, The Netherlands
azizi@cs.uu.nl, Remco.Veltkamp@cs.uu.nl

Marco A. Wiering
Department of Artificial Intelligence
University of Groningen, The Netherlands
mwiering@ai.rug.nl

***Abstract*—This paper presents the deep support vector machine (D-SVM) inspired by the increasing popularity of deep belief networks for image recognition. Our deep SVM trains an SVM in the standard way and then uses the kernel activations of support vectors as inputs for training another SVM at the next layer. In this way, instead of the normal linear combination of kernel activations, we can create non-linear combinations of kernel activations on prototype examples. Furthermore, we combine different descriptors in an ensemble of deep SVMs where the product rule is used for combining probability estimates of the different classifiers. We have performed experiments on 20 classes from the Caltech object database and 10 classes from the Corel dataset. The results show that our ensemble of deep SVMs significantly outperforms the naive approach that combines all descriptors directly in a very large single input vector for an SVM. Furthermore, our ensemble of D-SVMs achieves an accuracy of 95.2% on the Corel dataset with 10 classes, which is the best performance reported in literature until now.**

***Keywords*-Image categorization, support vector machines, ensemble methods, product rule, deep architectures**

I. Introduction

MACHINE VISION is a subfield of artificial intelligence that focuses on extracting useful information from images. During the last decade a large number of novel algorithms have been described for image recognition and this has led to good recognition performance on many different benchmarks. These algorithms use descriptors for representing an image with feature vectors and then a machine learning algorithm to classify the images. There are several machine learning algorithms, however, here we concentrate on support vector machines, deep architectures, and ensembles of classifiers that are considered to be among the best algorithms.

Deep architectures have been shown to be effective in learning and have been used with impressive performance for example in classification of digits in the MNIST dataset [3], [11] and modeling human motion [19]. In the lowest layer, feature detectors are used to detect simple patterns. After that, these patterns are fed into deeper, following, layers that form more complex representations of the input data. There are several approaches to learning deep architectures. Hinton et al. [12] proposed the deep belief network (DBN), where a multilayer generative model is used to encode statistical dependencies among the units in the layer below. These deep belief networks use neural networks, or more precisely, restricted Boltzmann machines, that are trained in a greedy fashion, that is, one layer is fully trained after which a following layer is added. After the training phase has been completed, fine-tuning of the whole architecture is often done by algorithms such as conjugate gradients.

Instead of DBNs that are grounded on the use of neural networks, we propose to use deep support vector machines (D-SVMs). The deep SVM is constructed by first training an SVM in the standard way. Then the kernel activations of the support vectors are used as inputs for another SVM in the following layer. This next layer SVM is then trained and is able to construct non-linear combinations of the kernel activations of the stored prototype examples (the support vectors). Since the training procedure of the deep SVM is done in a greedy fashion, it is computationally very efficient.

Next to deep architectures and support vector machines, ensemble methods have often been used for efficiently combining classifiers [8]. Based on these ideas, we propose to use an ensemble of deep SVMs. We have chosen to use the product rule [18] to combine the outputs of different classifiers (after computing probability estimates for the different classes). This is an effective method with the advantage that it is fast and uses all the information available in the outputs of the different classifiers (unlike for example bagging [5] that may fail for multi-class problems).

In this paper we use two different datasets, namely Corel and Caltech-101, to compare different combination architectures on four MPEG-7 image descriptors and many different edge and gradient based histograms using color and intensity information. We present the results of three methods that combine all descriptors: (1) The naive approach that combines all descriptors in a single input vector for a support vector machine. (2-3) An ensemble of standard and deep SVMs that uses the product rule to combine the posterior probabilities of classifiers for image classification.

Contributions. The originality of our work is: (1) We present the deep SVM that combines ideas from deep neural network architectures with those of support vector machines. (2) We construct and evaluate an ensemble of shallow and deep SVMs on two different image recognition datasets. (3) We demonstrate the effectiveness of our ensemble of deep SVMs by comparing it to the standard SVM that combines all

image descriptors in a single large input vector. (4) We report an accuracy of 95.2% on the Corel dataset with our ensemble of deep SVMs, which is the best performance reported in literature to the best of our knowledge.

The rest of the paper is organized as follows: Section II describes some fundamental principles of SVMs and introduces the deep SVM. Section III reviews several ensemble methods for combining multiple classifiers and describes our ensemble of deep SVMs. In Section IV, we describe image descriptors that we used to extract features from images. Experimental results on the Corel and Caltech-101 datasets are shown in Section V. Section VI concludes this paper.

II. Deep Support Vector Machines

The support vector machine is a state-of-the-art technique for data classification proposed by Vapnik and his group at AT&T Bell Laboratories [20], [6]. It was originally developed for binary or two-class classification and has been extended to the multi-class case and to regression. In this paper, the classification method is used in all experiments. Given an input pattern $\mathbf{X}$, the support vector machine classifies the input pattern into class $y \in \{-1, +1\}$ according to

$$y = sign(f(\mathbf{X})) \tag{1}$$

where the decision function $f(\mathbf{X})$ is a linear combination of kernels $K(\mathbf{X}_i, \mathbf{X})$ measuring the similarities between the presented vector $\mathbf{X}$ and each of the training vectors $\mathbf{X}_i$:

$$f(\mathbf{X}) = \sum_{i=1}^{n} \alpha_i y_i K(\mathbf{X}_i, \mathbf{X}) + b \tag{2}$$

Equation (2) is called the support vector expansion, and contains examples that store all necessary knowledge of a training set [17]. The α_i's are called support vector coefficients and these values are non-zero only for training data that are support vectors. The y_i values are the class labels belonging to the training data. Finally, b is a bias term.

A. Deep SVM Classifier

The main idea of the D-SVM is to combine kernel activations in non-linear ways. The standard SVM only optimizes the weights between the kernel activations of stored prototype examples and the output. Training is done by solving a quadratic optimization problem to optimize the weights (usually called support vector coefficients). The support vector set contains all information for constructing the decision function of a classifier [17], [20], however, their kernel activations are in the standard SVM combined in a linear way, since otherwise the optimization problem becomes too complex. The deep SVM allows for a hierarchical level representation of patterns via non-linear mixtures of prototype examples. It is inspired by deep belief networks [12] that are becoming more and more popular in the machine learning community. Unlike DBNs that are based on neural networks, the deep SVM is based on SVMs that have usually better generalization performance than standard neural networks.

Training deep SVMs is done by first training the lowest layer SVM in the standard way. Then the kernel activations are computed on the training set and stored together with the desired labels. This creates a new training dataset for the following layer where another SVM is trained using the kernel activations from one layer below. This can in principle continue for as many layers as are needed, and it is possible to use different kernels in different layers as well.

Note that the effect of the deep SVM cannot be achieved with particular choices of (complex) kernel functions. With the D-SVM it is possible to classify an instance as positive if the kernel activation of one support vector is large or the kernel activation of another support vector is large, while the classification can then be negative when both kernel activations are large. This is an example of the famous X-or problem that can be solved with two RBF kernels in the second layer SVM. Although DBNs usually use sigmoid functions, in this paper we have mostly concentrated on the RBF kernel, since it uses less parameters to optimize and preliminary experiments indicated that it performed slightly better than the sigmoid or Tanh kernel.

III. Ensemble Methods

Ensemble methods have received considerable attention in the machine learning community to increase the effectiveness of classifiers. In order to construct a good ensemble classifier, the ensemble needs to construct accurate and diverse classifiers and to combine outputs from the classifiers effectively [8]. There exist several methods to obtain and combine the diverse classifiers.

In bagging [5], a training dataset is divided into several different subsets that may be overlapping. After that, a machine learning algorithm is trained on each subset. Then, the majority voting scheme is used to combine the class-votes of the different classifiers. If the outputs of the different classifiers are strongly uncorrelated, the ensemble may correct for independent mistakes by single classifiers and this improves the classification accuracy.

Constructing and combining a set of classifiers is more complicated in boosting [10]. Boosting methods construct a set of classifiers in a sequential way. First one classifier is trained on all data, and then examples that are misclassified by the first classifier get higher weights in the training process of the next classifier. This is repeated until the whole set of classifiers has been trained. The final ensemble uses a weighted majority voting scheme where the weight of a classifier is dependent on the accuracy of the classifier.

Another ensemble method is the hierarchical mixtures of experts (HME) architecture [14]. In the HME there is a gating network that learns to partition the input space in different regions where different classifiers are used to learn and predict the examples falling in their different regions. The HME exploits the divide and conquer principle, but it is more complicated to use together with SVMs.

Stacking [23] is another ensemble method that learns to combine the outputs of different classifiers. First different classifiers are trained, and then another classifier receives as

inputs all the predictions of the different classifiers and is trained to optimally combine the different classifier outputs. In our previous work we used stacking SVM classifiers to combine different SVMs trained with different image descriptors [1], and this led to better results than using a single SVM with all features from the different descriptors.

The product rule is one of the simplest and most efficient ways for combining outputs of classifiers [18] and is used in our ensemble architecture of this paper. When the classifiers have small errors and operate in independent feature spaces, it is very efficient to combine their (probabilistic) outputs by multiplying them. Thus, we use this product rule to determine the final decision of the ensemble. First the posterior probability outputs $P_j^k(x^k)$ for class j of n different classifiers are combined by the product rule:

$$P_j(x^1, ..., x^n) = \prod_{k=1}^{n} P_j^k(x^k) \tag{3}$$

where x^k is the pattern representation of the k^{th} descriptor. Then the class with the largest probability product is considered as the final class label belonging to the input pattern.

A. Ensemble of deep SVMs

Combining multiple classifiers that receive different features as inputs is an important topic in pattern and image recognition. The main idea is that each descriptor produces different information for representing the input pattern, which makes the classifiers diverse enough for efficient use in an ensemble. This is in contrast with the naive approach, where the feature vectors from all sources are concatenated to train a single classifier. In this case, care has to be exercised regarding the increase of the feature dimensionality that may cause overfitting and worse generalization. One strategy to overcome the problem is to learn different classifiers with different features separately. After that, the outputs are combined by an ensemble method to generate the final output. In this paper, we report the results of two different multiple image descriptor combination methods and compare these to our proposed ensemble of deep SVMs. We will first describe these three combination methods.

1) Naive approach: This approach concatenates the feature vectors from different sources and creates a single feature vector for modeling the content of an image. Fig 1 shows how the naive approach combines multiple image features. In this figure, the feature calculation function contains an algorithm to describe images by histograms.

2) Ensemble of SVMs: We train different SVMs and compute the class probabilities with the probability estimation function of SVMs. Then we use the product rule [18] to combine all probability outputs of the SVM classifiers. The main reason we use this approach is that it is a simple and effective method to combine classifiers trained with different image descriptors. This approach can be used to produce diverse classifiers, since the image descriptors provide complimentary representations of images. Fig 2 shows the ensemble of SVMs where the product rule is used to compute the final classification.

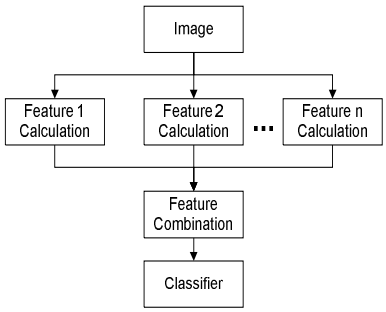

Figure 1. *Combining multiple image features using the naive approach.*

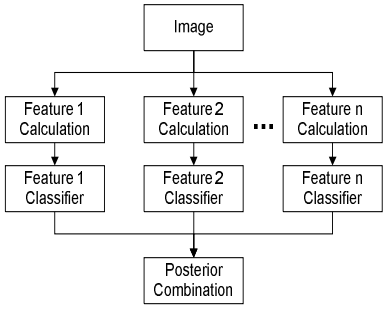

Figure 2. *Ensemble of support vector machines.*

3) Ensemble of deep SVM classifiers: We adopt the product rule [18] for combining multiple probability outputs of the deep SVM classifiers. Based on this idea, we construct a two-layer SVM classifier for each one-vs-all classification task. The system first trains a set of SVM classifiers separately and this process is performed at the first layer of the architecture. After that, the support vector activations are extracted from each classifier of the first layer to learn another SVM classifier for the second layer of the architecture belonging to the same one-vs-all classifier. The outputs from the second layer can give better distinctions than the first layer since inputs to the second layer classifiers are based on activations of prototype examples, rather than simple features.

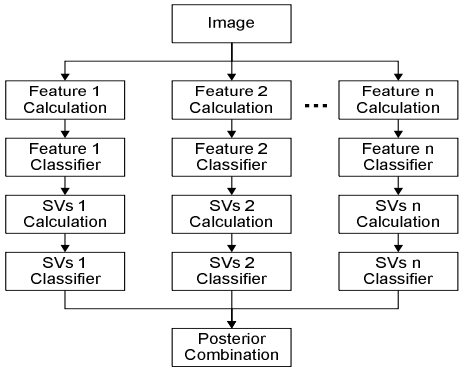

Figure 3. *Ensemble of deep support vector machines.*

IV. Image Representation and Descriptors

A good image feature for visual content description is crucial and helps to discover meaningful patterns in the image. There is no agreement what type of features should be used to produce an optimal result for all images. However, using more than one image descriptor has been shown to be effective in increasing the recognition performance.

A. MPEG-7 cluster correlogram descriptors

A set of MPEG-7 descriptors with the fixed partitioning cluster correlogram [2] is used to evaluate the proposed methods on the Corel image dataset. It contains two main low-level descriptors, i.e., color and texture descriptors. We used the MPEG-7 features, because our preliminary results showed that these descriptors are informative to describe scenes and objects in this dataset. The fixed partitioning cluster correlogram consists of three main steps. The first step is extracting the visual features for each MPEG-7 low-level descriptor in each block of the image. The MPEG-7 descriptors that we used are Scalable Color, Color Layout, Color Structure and the Edge Histogram. After that we use the K-means algorithm to construct a set of visual keywords (we used 24 or 32 keywords for the different descriptors). Finally, the cluster correlogram is constructed for each descriptor to index images in the dataset. The cluster correlogram is basically a matrix representing how often a keyword dominating a block is adjacent to another keyword in a neighboring block (we use 8 neighbors).

B. Spatial pyramid with edge and orientation descriptors

We used the spatial pyramid as described in [15] and shape-based descriptors to evaluate the classifiers on Caltech-101. The spatial pyramid consists of one global and several local feature histograms to describe images using multiple resolutions. We used three different levels of resolution and our descriptors in [1] to index images on the Caltech dataset. The descriptors are the MPEG-7 Edge Histogram, Histograms of Threshold-oriented Gradients (HTOG) [7] and gradient based histograms of the Scale Invariant Feature Transform (SIFT) [16]. We used color and intensity information for all descriptors and two angular ranges namely 180° and 360° for HTOG and SIFT. The total number of descriptors we used for Caltech is 10.

V. Experimental Results

For our comparison between the independent descriptors, the naive SVM classifier, the ensemble of SVMs, and the ensemble of deep SVMs, the Corel and Caltech-101 datasets were chosen. For Corel we used the first 10 categories and a total of 10x100=1000 images. The images in the Corel dataset seem quite simple with little or no occlusion and clutter, and the pictures in each class tend to be similar in viewpoints and orientations. In contrast, the Caltech-101 dataset contains 101 different classes. In our experiments, we used only the first 20 classes due to computational restrictions. Each object in the dataset has a different size and is seen from different viewpoints, which makes the recognition task more challenging.

A. SVM classifiers

As mentioned before, we employ SVMs [20] to learn to classify images. The one-vs-all approach is used to train and classify the images in the Corel and Caltech-101 datasets. For the SVMs, we have tried several kernels in the naive and ensemble classifiers, however, in this paper, only the results of the best kernel (the RBF kernel) are reported. All attributes in the training and testing datasets were normalized to the interval [-1,+1] by using the following equation:

$$x' = \frac{2(x-min)}{(max-min)} - 1.$$

The normalization is used to avoid numerical difficulties during the calculation and to make sure the largest values do not dominate the smaller ones. The min and max values are determined from the training dataset. We have used the same normalization scheme when passing the kernel activations from one layer to the next in the deep SVM.

We also did experiments to find the values for the SVM parameters C and γ that perform best for the descriptors. We found that it can be difficult to find the best parameters due to unbalanced datasets caused by the one-vs-all training scheme. The unbalanced datasets may cause a biased classification performance — a high accuracy on the majority class (-1), but a very low accuracy on the minority class (+1). Therefore we employed two parameter optimization methods in our experiments: (1) With accuracy the learning parameters were determined by using the libsvm grid-search algorithm [13]. In this approach, 5-fold cross-validation is used to find the best parameters by measuring the performance of the classifiers in the one-vs-all classification tasks. (2) We also employed the Weka [22] machine learning software package for optimizing the learning parameters using the F1-measure. In this approach, 5-fold cross-validation is used to find the best parameters by measuring the performance of different parameters in the one-vs-all classification tasks.

With both parameter optimization methods, we tried the following values: $\{2^{-5},2^{-3},...,2^{15}\}$ and $\{2^{-15},2^{-13},...,2^{3}\}$ for C and γ, respectively. We report only the results obtained with the best found learning parameters below.

B. Results on the Corel dataset

The Corel dataset is one of the most popular and widely used datasets to demonstrate the performance of CBIR systems [21]. It contains images that were categorized into 10 different groups as shown in Fig. 4. For evaluating the SVM classifiers, we used 90% of the images for training and 10% for testing for each class. To compute the performances of the different methods, we chose 15 times different training and test images. We used the accuracy measure (as explained above) to optimize the learning parameters of all methods. The values which gave the best performance on the first training dataset are used on all training sets. Only tuning using the first dataset saved us a lot of computational time.

We report the mean accuracy and the standard deviation of the classifiers.

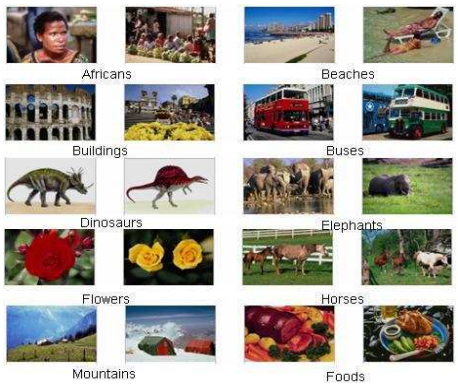

Figure 4. *Image examples of Corel with ground truth for different groups, namely Africans, beaches, buildings, buses, dinosaurs, elephants, flowers, horses, mountains and foods.*

Table I shows the first comparison between the standard SVM and the deep SVM with two layers consisting of RBF kernels. Here, the fixed partitioning cluster correlogram with different MPEG-7 descriptors is used. The table shows that the deep SVM gives some improvement on all independent descriptors, although the differences are not quite significant (according to the student t-test). Table II shows the accuracies using the three different evaluated architectures: the naive SVM, the ensemble of SVMs (E-SVM), and the ensemble of deep SVMs (E-D-SVM). Combining multiple descriptors using the ensemble of deep SVMs significantly outperforms the standard SVM ($p < 0.05$) and also performs slightly better than the ensemble of standard SVMs. The performance with a 4.8% error-rate on Corel is the best result reported in literature to the best of our knowledge. Also note that although the differences between the naive approach with an error-rate of 6.1% and the ensemble of D-SVMs with an error-rate of 4.8% does not seem large, it is significant and the reduction of the error is more than 20%. The ensemble of SVMs does not perform significantly better than the naive approach. Finally, note that combining all descriptors works much better than using a single descriptor alone.

Table I
THE AVERAGE CLASSIFICATION ACCURACY (MEAN AND SD) OF THE FIXED PARTITIONING CLUSTER CORRELOGRAM USING DIFFERENT MPEG-7 DESCRIPTORS. FP1 = COLOR LAYOUT, FP2 = COLOR STRUCTURE, FP3 = SCALABLE COLOR, AND FP4 = EDGE HISTOGRAM.

Method	$FP1$	$FP2$	$FP3$	$FP4$
SVM	80.9 ± 2.8	83.4 ± 4.4	76.9 ± 4.1	65.9 ± 3.7
D-SVM	82.7 ± 2.8	83.7 ± 4.1	77.4 ± 4.1	67.1 ± 3.6

Table II
THE AVERAGE CLASSIFICATION ACCURACY (MEAN AND SD) OF THE DIFFERENT COMBINATION ARCHITECTURES.

Naive SVM	E-SVM	E-D-SVM
93.1 ± 2.9	94.4 ± 2.3	95.2 ± 1.9

C. Results on the Caltech-101 dataset

The Caltech-101 dataset is one of the most popular and widely used datasets to demonstrate the performance of object recognition systems [9]. The images with different sizes were categorized into 101 classes, however we used only the first 20 classes as shown in Fig. 5 for computational reasons. Furthermore, we used the regions of interest (ROIs) of the images as obtained by the research described in [4]. For evaluating the SVM classifiers, we used 15 training and 15 testing images for each image class. We chose 5 times different training and test images randomly taken from the dataset to evaluate the performances of the different methods. We used the accuracy measure to optimize the learning parameters of the standard SVMs and the first layer of the D-SVM, but used the F1-measure to optimize the second layer of the D-SVM, which gave slightly better results than accuracy. We used the RBF kernels for all SVMs and report the mean accuracies and standard deviations.

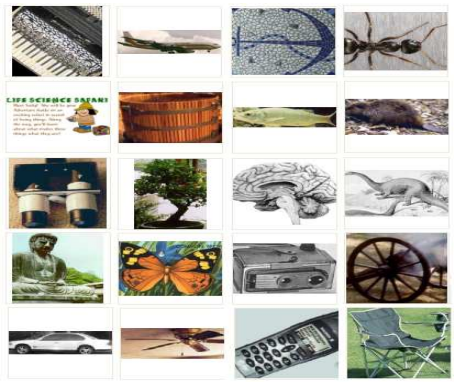

Figure 5. *Image examples of Caltech with ground truth for 20 different groups, namely accordion, airplane, anchor, ant, background, barrel, bass, beaver, binocular, bonsai, brain, brontosaurus, Buddha, butterfly, camera, cannon, car side, ceiling fan, cell phone and chair.*

We first tested the spatial pyramid with edge and gradient based histogram descriptors separately. Table III shows that for most descriptors there is a slight improvement when the deep SVM is used instead of the standard SVM, although the differences are not quite significant (for some descriptors p is around 0.1, though). Finally, we tested the three combination methods using 5 times different training and test image datasets. Table IV shows the performances using the three different combination approaches. Similar to the results with Corel, the ensemble of deep SVMs significantly outperforms the naive approach and performs better (although not significantly) than the ensemble of SVMs. Here, the ensemble of standard SVMs also significantly outperforms the naive approach. Finally, note that all combination methods perform again much better than using a single descriptor.

VI. CONCLUSION

We have introduced the deep support vector machine that can build multi-layer support vector machines where kernel

Table III
THE AVERAGE CLASSIFICATION ACCURACY (MEAN AND SD) OF INDEPENDENT DESCRIPTORS.

Descriptor	SVM (%)	D-SVM (%)
EH_G	62.0 ± 1.4	62.1 ± 0.5
EH_C	64.1 ± 2.1	64.6 ± 1.4
S_180_G	72.7 ± 1.4	72.9 ± 1.7
S_180_C	71.1 ± 1.0	70.9 ± 0.5
S_360_G	65.4 ± 2.6	67.7 ± 1.2
S_360_C	66.4 ± 2.3	68.3 ± 0.6
HG_180_G	70.1 ± 2.5	69.1 ± 0.7
HG_180_C	69.1 ± 2.0	70.8 ± 0.5
HG_360_G	63.5 ± 2.3	64.3 ± 5.0
HG_360_C	65.5 ± 2.8	67.3 ± 0.9

Table IV
THE AVERAGE CLASSIFICATION ACCURACY (MEAN AND SD) OF THE DIFFERENT COMBINATION ARCHITECTURES.

Naive SVM	E-SVM	E-D-SVM
77.4 ± 0.9	82.1 ± 3.0	83.1 ± 2.2

activations of prototype examples can be mixed in non-linear ways. We combined the deep SVM with a product rule ensemble for combining multiple image descriptors and have evaluated our approach on the Corel and Caltech datasets. The results show that the deep SVM architecture with the product rule handles multiple features efficiently and performs significantly better than a standard SVM.

There are several ways to extend this research. Instead of using the RBF-RBF kernel combination, other combinations can be researched. Furthermore, it may be worthwhile to also use the support vector coefficients to scale the kernel activations. Finally, now we used the same training data for the different layers in the D-SVM. We want to study the effect of using different datasets for training different layers.

ACKNOWLEDGEMENTS

The first author wants to thank the government of Malaysia for the Ph.D. grant.

REFERENCES

[1] A. Abdullah, R. C. Veltkamp, and M. Wiering. Spatial pyramids and two-layer stacking SVM classifiers for image categorization: A comparative study. In *International Joint Conference on Neural Network (IJCNN 09)*, 2009.

[2] A. Abdullah and M. Wiering. CIREC: Cluster correlogram image retrieval and categorization using MPEG-7 descriptors. In *IEEE Symposium on Computational Intelligence in Image and Signal Processing*, pages 431 – 437, 2007.

[3] Y. Bengio and Y. Lecun. Scaling learning algorithms towards AI. In *Large-Scale Kernel Machines*. MIT Press, 2007.

[4] A. Bosch, A. Zisserman, and X. Muñoz. Image classification using ROIs and multiple kernel learning. Submitted to the International Journal of Computer Vision, June 2008.

[5] L. Breiman. Bagging predictors. *Machine Learning*, Vol. 24(2):123–140, 1996.

[6] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Volume 1*, pages 886–893, 2005.

[8] T. G. Dietterich. Ensemble methods in machine learning. In *Multiple Classifier Systems*, pages 1–15. Springer-Verlag, 2000.

[9] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2004)*, page 178, 2004.

[10] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *European Conference on Computational Learning Theory*, pages 23–37, 1995.

[11] G. E. Hinton. To recognize shapes, first learn to generate images. *Progress in brain research*, 165:535–547, 2007.

[12] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, July 2006.

[13] C. W. Hsu, C. C. Chang, and C. J. Lin. A practical guide to support vector classification. Technical report, Department of Computer Science and Information Engineering, National Taiwan University, 2003.

[14] M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6:181–214, 1994.

[15] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, pages 2169–2178, 2006.

[16] D. G. Lowe. Distinctive image features from scale-invariant keypoints. In *International Journal of Computer Vision*, volume 60, pages 91–110, 2004.

[17] B. Schölkopf, C. Burges, and V. Vapnik. Extracting support data for a given task. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, pages 252–257. AAAI Press, 1995.

[18] D. M. Tax, R. P. Duin, and M. V. Breukelen. Comparison between product and mean classifier combination rules. *In Proceedings of the Workshop on Statistical Pattern Recognition*, pages 165–170, 1997.

[19] G. W. Taylor, G. E. Hinton, and S. Roweis. Modeling human motion using binary latent variables. *Advances in Neural Information Processing Systems*, 2007.

[20] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.

[21] J. Z. Wang, J. Li, and G. Wiederhold. SIMPLIcity: Semantics-sensitive integrated matching for picture Libraries. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 23(9):947–963, 2001.

[22] I. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques, 2nd Edition*. Morgan Kaufmann, San Francisco, 2005.

[23] D. H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.