Active Metric Learning for Object Recognition

Sandra Ebert, Mario Fritz, and Bernt Schiele

Max Planck Institute for Informatics
Saarbrucken, Germany

Abstract. Popular visual representations like SIFT have shown broad applicability across many task. This great generality comes naturally with a lack of specificity when focusing on a particular task or a set of classes. Metric learning approaches have been proposed to tailor general purpose representations to the needs of more specific tasks and have shown strong improvements on visual matching and recognition benchmarks. However, the performance of metric learning depends strongly on the labels that are used for learning. Therefore, we propose to combine metric learning with an active sample selection strategy in order to find labels that are representative for each class as well as improve the class separation of the learnt metric. We analyze several active sample selection strategies in terms of exploration and exploitation trade-offs. Our novel scheme achieves on three different datasets up to 10% improvement of the learned metric. We compare a batch version of our scheme to an interleaved execution of sample selection and metric learning which leads to an overall improvement of up to 23% on challenging datasets for object class recognition.

1 Introduction

Similarity metrics are a core building block of many computer vision methods e.g. for object detection [12] or human pose estimation [15]. Consequently, their performance critically depends on the underlying metric and the resulting neighborhood structure. The ideal metric should produce small intra-class distances and large inter-class distances. But standard metrics often have problems with high dimensional features due to their equal weighting of dimensions. This problem is particularly prominent in computer vision where different feature dimensions are differently affected by noise e.g. due to signal noise, background clutter, or lighting conditions.

A promising direction to address this issue is metric learning [6, 11, 9]. E.g., pairwise constraints from labeled data are used to enforce smaller intra-class distances. But this strategy can be problematic [18, 2] if only few labels are available that might be not informative enough to learn a better metric. For example, outliers may completely distort the metric while redundant samples may have little effect on metric learning. In this paper, we combine active sampling of labels with metric learning to address these problems.

In general, active learning methods [3, 8] use sample selection strategies to request uncertain as well as representative samples so that a higher classification performance can be achieved with only a small fraction of labeled training data. However, the success of active learning critically depends on the choice of the sample selection strategy. Therefore the first main contribution of this paper is to analyze which sampling strategy is best suited to improve metric learning. The analysis is done for three different datasets and in particular for settings where only a small number of labels is available. The second main contribution is to propose two methods that combine active sampling with metric learning leading to a performance improvements of up to 23%.

2 Related work

Supervised metric learning is a promising direction to improve the neighborhood quality of representations for computer vision. Frequently used are methods that learn a global Mahalanobis distance [6, 11, 9] based on pairwise constraints. One advantage of these methods is the kernelized optimization so that multiple kernels can be optimized at the same time [11] and the run time depends only on the number of labels instead of the dimensions. However, a large number of labeled pairs is required to learn a good Mahalanobis distance [10].

In contrast, active learning is a successful strategy to reduce the amount of labels by preserving the overall performance. Methods of active learning can be divided into exploration-driven (density-based), exploitation-driven (uncertainty-based), or a combination of both. Exploitative strategies focus mainly on uncertain regions [14, 16] while explorative methods sample more representative labels by considering the underlying data distribution [13, 5]. But it turns out that a combination of both strategies leads often to a better solution [3, 8].

But there are only few methods that try to reduce the number of required labels [2, 18]. In [18], the authors improve a Bayesian framework for metric learning by a pure exploitation-driven criteria. [2] refines a pairwise constrained clustering by incorporating a pure exploration-driven criteria. However, the previous work lacks an analysis of active sampling methods and there is no attempt to combine active sampling with metric learning in an interleaved framework.

3 Methods

In this section, we introduce the employed metric learning algorithm [6] as well as our active sampling procedure [8] including several criteria for exploration and exploitation. These criteria can be used either separately or in combination within our framework. Finally, we briefly introduce three different classification algorithms that are used with our active metric learning, i.e., k nearest neighbor classifier (KNN), SVM, and the semi-supervised label propagation (LP) [19].

3.1 Metric learning

We use the information-theoretic metric learning (ITML) proposed by [6]. ITML learns a global metric by optimizing the Mahalanobis distance,

$$d_A(x_i, x_j) = (x_i - x_j)^T A(x_i - x_j), \tag{1}$$

between two labeled points $x_i, x_j \in \mathbb{R}$ with a Mahalanobis matrix A such that intra-class distances are small and inter-class distances are large, i.e.,

$$\begin{aligned} &\min D_{ld}(A, A_0) \\ &s.t.\ d_A(x_i, x_j) \le u \quad (i,j) \in \mathcal{S} \\ &\qquad d_A(x_i, x_j) \ge l \quad (i,j) \in \mathcal{D} \end{aligned} \tag{2}$$

with LogDet loss D_{ld} and the original data space A_0. u and l are upper and lower bounds of similarity and dissimilarity constraints. $\mathcal{S}$ and $\mathcal{D}$ are sets of similarity and dissimilarity constraints based on the labeled data. This linear optimization can be easily transformed into a kernelized optimization by $K = X^T AX$ to speed up the learning.

3.2 Active sample selection

In this work, we explore two exploration and two exploitation criteria. Let us assume, we have $n = l+u$ data points with l labeled examples $L = \{(x_1, \hat{y}_1), ..., (x_l, \hat{y}_l)\}$ and u unlabeled examples $U = \{x_{l+1}, ..., x_n\}$ with $x_i \in \mathbb{R}^d$. We denote $\hat{y} \in \mathcal{L} = \{1, ..., c\}$ the labels with c the number of classes.

Exploitation. *Entropy* (Ent) is the most common criteria for exploitation [1] that uses the class posterior:

$$Ent(x_i) = -\sum_{j=1}^{c} P(y_{ij}|x_i) \log P(y_{ij}|x_i) \tag{3}$$

where $\sum_j P(y_{ij}|x_i) = 1$ are predictions of a classifier. This criteria focuses more on examples that have a high overall class confusion.

Margin (Mar) computes the difference between best versus second best class prediction [14]:

$$Mar(x_i) = P(y_{ik_1}|x_i) - P(y_{ik_2}|x_i) \tag{4}$$

such that $P(y_{ik_1}|x_i) \ge P(y_{ik_2}|x_i) \ge ... \ge P(y_{ik_c}|x_i)$. In each iteration, label $x^* = \text{argmin}_{x_i \in U} Mar(x_i)$ is queried. In contrast to *Ent*, this criteria concentrates more on the decision boundaries between two classes.

Exploration. These criteria are often used in combination with exploitation criteria as they do not get any feedback about the uncertainty during the active sample selection so that more labels are required to obtain good performance.

Kernel farthest first (Ker) captures the entire data space by looking for the most unexplored regions given the current labels [1, 2] by computing the minimum distance from each unlabeled sample to all labels

$$Ker(x_i) = \min_{x_j \in L} d(x_i, x_j), \tag{5}$$

and then requesting the label for the farthest sample $x^* = \text{argmax}_i Ker(x_i)$. This criteria samples evenly the entire data space but often selects many outliers.

Graph density (Gra) [8] is a sampling criteria that uses a k-nearest neighbor graph structure to find highly connected nodes, i.e.,

$$Gra(x_i) = \frac{\sum_i W_{ij}}{\sum_i P_{ij}}. \tag{6}$$

with the similarity matrix $W_{ij} = P_{ij} \exp\left(\frac{-d(x_i,x_j)}{2\sigma^2}\right)$ and the adjacency matrix P_{ij}. After each sampling step, the weights of direct neighbors of sample x_i are reduced by $Gra(x_j) = Gra(x_j) - Gra(x_i)P_{ij}$ to avoid oversampling of a region.

Active sampling. We use our time-varying combination of exploration and exploitation introduced in [8], i.e.,

$$H(x_i) = \beta(t)r(U(x_i)) + (1 - \beta(t))r(D(x_i)) \tag{7}$$

with $U \in \{Ent, Mar\}$, $D \in \{Ker, Gra\}$, $\beta(t) : \{1, ..., T\} \to [0, 1]$, and a ranking function $r : \mathbb{R} \to \{1, ..., u\}$ that uses the ordering of both criteria instead of the values itself. We set $\beta(t) = log(t)$ that means more exploration at the beginning followed by exploitation at the end of the sampling process. Finally, we request the label for the sample with the minimal score $\text{argmin}_{x_i \in U} H(x_i)$.

3.3 Classification algorithms

In the following, we explain the use of three different classifier in our active sampling framework because not all classifier provide a class posterior that can be immediately used for *Ent* or *Mar*.

1) KNN. Similar to [11], we show results for the k nearest neighbor classifier with $k = 1$ because it shows consistently best performance. For the class posterior $p(y_{ij}|x_i)$, we use the confusion of the 10 nearest labels for each unlabeled data point weighted by their similarity and finally normalized by the overall sum.

2) SVM. We apply libSVM [4] with our own kernels in a one-vs-one classification scheme. The accumulated and normalized decision values are used as the class posterior. Parameter C is empirically determined but is quite robust.

3) Label propagation (LP). For semi-supervised learning, we use [19] that propagates labels through a k nearest neighbor structure, i.e.,

$$Y_j^{(t+1)} = \alpha S Y_j^{(t)} + (1 - \alpha) Y_j^{(0)} \tag{8}$$

with $1 \leq j \leq c$, the symmetric graph Laplacian $S = D^{-1/2} W D^{-1/2}$ based on the similarity matrix W from above, the diagonal matrix $D_{ii} = \sum_j W_{ij}$, the original label vector $Y_j^{(0)}$ consisting of $1, -1$ for labeled data and 0 for the unlabeled data. Parameter $\alpha \in (0, 1]$ that controls the overwriting of the original labels. The final prediction is obtained by $\hat{Y} = \text{argmax}_{j \leq c} Y_j^{(t+1)}$. For the class posterior, we use the normalized class predictions $P(y_{ij}|x_i) = \frac{y_{ij}^{(t+1)}}{\sum_{j=1}^{c} y_{ij}^{(t+1)}}$.

4 Active metric learning

By requesting more informative and representative training examples, we expect that metric learning achieves better performance given the same amount of training data or – respectively – achieve equal performance already with significantly less annotated data. Therefore, we explore two different ways to combine active sampling with metric learning.

4.1 Batch active metric learning (BAML)

Our first approach starts by querying the desired number of labeled data points according to the chosen sample selection strategy and learns a metric based on this labeled data. As the metric is learnt only once across the whole pool of labeled data points, we call this approach *batch active metric learning (BAML)*. While this method obtains good performance, it does not get any direct feedback involving the learnt metric during sampling. To improve the coupling between the two processes we propose a second version of our method which interleaves active sampling and metric learning.

4.2 Interleaved active metric learning (IAML)

The second active metric learning approach alternates between active sampling and metric learning. We start with active sampling in order to have a minimum of similarity constraints for metric learning. In our experiments, we apply metric learning each mc iterations with $2 \leq m \leq |L|$, c the number of classes, and $|L|$ the average number of requested labels per class. After metric learning we use the learned kernel to request the next batch of labels with active sampling. In each iteration we learn the metric based on the original feature space with the current available labels and all pairwise constraints. We found experimentally that using the original feature space is less susceptible to drift than incrementally updating the learnt metric.

5 Datasets and representation

In our experiments, we analyze three different datasets for image classification with increasing number of classes and difficulty. Fig. 1 shows sample images.

ETH-80 consists of 8 classes (*apple*, *car*, *cow*, *cup*, *dog*, *horse*, *pear*, and *tomato*) photographed from different viewpoints in front of a uniform background. This dataset contains $3,280$ images.

C-PASCAL is subset of the PASCAL VOC challenge 2008 data used in [7] in a multi-class setting. Single objects are extracted by bounding box annotations. The resulting dataset consists of $4,450$ images of aligned objects from 20 classes but with varying object poses, background clutter, and truncations.

IM100 is a subset ImageNet 2010 that consists of 100 classes similar to Caltech 101. IM100 contains 100 images per class resulting in a dataset with

Fig. 1. Sample images for ETH (left), C-PASCAL (middle), and IM100 (right).

10,000 images. Objects can be anywhere in an image and images often contain background clutter, occlusions, or truncations.

Representation. In the experiments, we show results for a dense SIFT Bag-of-Words representation. SIFT-features are extracted using the implementation by [17], sampled on a regular grid, and quantized into 1,000 visual words.

6 Experiments

In our experimental section, we first analyze in Sec. 6.1 different sampling criteria and their combinations in terms of representativeness for metric learning. We focus on the 1-NN classification performance as it reflects the change of the underlying metric. Then, we explore in Sec. 6.2 if these insights transfer also to other algorithms. Finally in Sec. 6.3, we show further improvements by applying our interleaved active metric learning (IAML) framework.

6.1 Different sampling criteria for metric learning

In this subsection, we analyze several sampling criteria and mixtures of those in comparison to random sampling and their influence on the entire metric. For this purpose, we look at the 1-NN accuracy as this measure gives a good intuition about the learned neighborhood structure. Tab. 1 shows results before and after metric learning for different average number of labels per class $|L|$. We request at most 10% labels, i.e., for ETH we vary $|L|$ from 5 to 25 and for IM100 from 3 to 10. *Rand* is our baseline using random sampling where we draw exactly $|L|$ labels per class with a uniform distribution. Last line in each table is the average performance over the whole column. All results are averaged over 5 runs.

Before metric learning (Tab. 1, top), we notice large differences between several sampling criteria. In average, we observe a performance of 29.7% for random sampling while for single active sampling criteria the accuracy vary from 26.2% for *Ker* to 31.4% for *Mar*. Both *Mar* and *Gra* are better than *Rand*. *Ent* and *Ker* are worse than *Rand* due to their tendency to focus more on low density regions. Then we look at each specific dataset, *Mar* performs best for ETH that contains a smooth manifold structure. In contrast, *Gra* tends to oversample dense regions, e.g., *pear*, leading to worse performance in comparison to *Ker*. On more complex datasets such as C-PASCAL or IM100, *Gra* clearly

Accuracy before metric learning									
		Single criteria				Mixture of two criteria			
$\lvert L\rvert$	Rand	Ent	Mar	Gra	Ker	M+G	M+K	E+G	E+K
ETH									
5	50.6	45.9	57.0	51.1	46.0	**59.8**	43.3	55.0	49.1
15	69.1	59.7	69.7	62.6	64.0	**71.0**	65.1	62.0	60.5
25	74.2	62.7	74.4	69.8	72.4	**77.3**	72.1	66.2	66.4
C-PASCAL									
5	12.6	11.3	16.1	17.8	9.8	19.1	11.1	17.1	10.3
15	17.5	19.8	21.0	**24.1**	12.4	23.2	14.9	21.8	17.5
25	19.3	21.8	23.4	**27.5**	13.9	24.8	17.7	24.5	19.7
IM100									
3	6.3	5.1	5.6	**8.2**	5.1	**8.2**	5.4	7.2	5.2
5	7.6	6.0	6.8	**9.3**	5.6	**9.3**	6.2	8.1	5.9
10	9.8	7.3	8.6	10.5	7.0	**10.6**	7.9	9.0	7.0
Overall average									
	29.7	26.6	31.4	31.2	26.2	**33.7**	27.1	30.1	26.8

Accuracy after metric learning									
		Single criteria				Mixture of two criteria			
$\lvert L\rvert$	Rand	Ent	Mar	Gra	Ker	M+G	M+K	E+G	E+K
ETH									
5	61.6	59.3	67.7	52.7	67.5	**70.0**	63.3	62.7	65.8
15	79.8	67.9	82.2	69.1	80.0	**83.0**	82.0	70.7	76.3
25	82.8	74.6	84.5	78.1	83.5	**86.3**	86.1	73.3	79.4
C-PASCAL									
5	16.9	19.4	22.4	23.5	17.1	25.7	20.0	**26.2**	18.9
15	25.2	32.5	32.6	34.4	18.5	**34.5**	22.4	33.2	29.1
25	28.8	37.9	**39.0**	36.9	22.5	38.4	29.6	38.4	36.6
IM100									
3	6.7	6.4	7.4	9.3	6.8	**10.6**	7.0	9.6	6.8
5	11.4	8.6	9.6	10.7	8.0	**13.0**	9.2	11.7	8.6
10	15.9	12.6	14.6	12.5	11.1	**16.3**	14.6	15.3	12.4
Overall average									
	36.6	35.5	40.0	36.4	35.0	**42.0**	37.1	37.9	37.1

Table 1. 1-NN accuracy before (1st table) and after (2nd table) metric learning for single criteria and the mixtures Ent+Gra (E+G), Ent+Ker (E+K), Mar+Gra (M+G), and Mar+Ker (M+K).

outperforms all other single criteria. For C-PASCAL with 25 labels per class we achieve a performance of 27.5% for *Gra* while *Mar* shows a performance of 23.4% and *Ker* achieves only 13.9% accuracy. Finally, the combination *Mar+Gra* outperforms with 33.7% in average the best single criteria with 31.2%. All other combinations are strongly limited to the power of the combined criteria that means using *Gra* shows better performance than using *Ker*, and mixtures with *Mar* are in average better than mixtures with *Ent*.

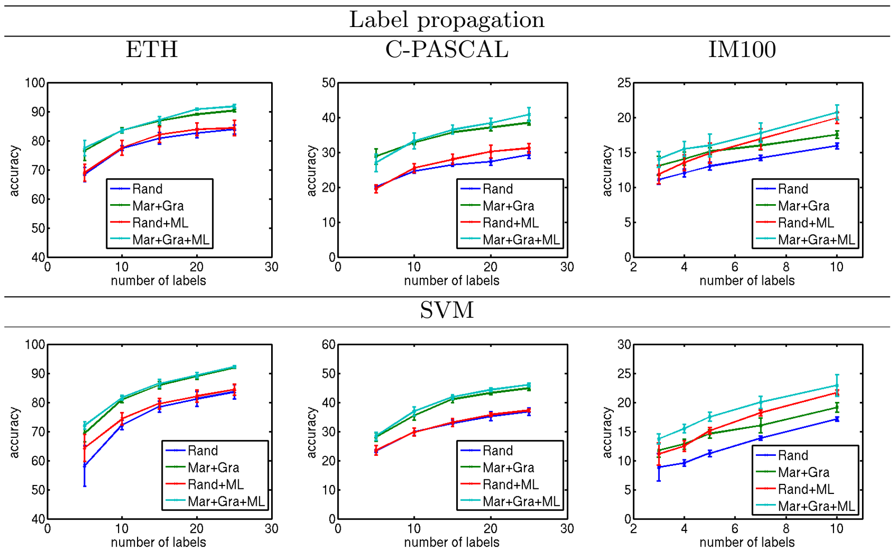

Fig. 2. LP and SVM accuracy of all three datasets and different number of labels for random sampling and the mixture Mar+Gra with and without metric learning.

After metric learning (Tab. 1, bottom), we observe a consistent improvement to the previous table that means metric learning always helps. For example, *Rand* is overall improved by 6.9% from 29.7% without metric learning to 36.6% with metric learning and our best combination *Mar+Gra* is increased in average by 8.3% from 33.7% to 42.0%. From these improvements we see also that there is a larger benefit when using our BAML in comparison to *Rand* with metric learning. This observation also holds true for most other active sampling selection methods, e.g., *Ent+Ker* is improved by 10.3% from 26.8% to 37.1% that is better than *Rand* after metric learning. Another important insight results from the comparison of the influence of active sample selection on metric learning. Obviously, metric learning has a larger impact on the overall performance than active sample selection that means *Rand* is improved from 29.7% to 33.7% with *Mar+Gra* and to 36.6% with metric learning alone. But if we combine both strategies we achieve a final performance of 42.0% that corresponds to an overall increase of 12.3% across three datasets.

To conclude this subsection, metric learning benefits significantly from labels that are more representative. In average, *Mar+Gra* is the best sampling strategy for our BAML. Finally, metric learning combined with active sample selection achieves consistent improvements over random sampling of up to 12.3%.

6.2 BAML on LP and SVM

In this subsection, we explore if our insights from the previous subsection translate to more complex classification schemes such as label propagation (LP) or

	ETH			C-PASCAL			IM100		
$\|L\|$	BAML	IAML	diff	BAML	IAML	diff	BAML	IAML	diff
5	70.0	68.0	-2.0	25.7	23.3	-2.4	10.6	10.5	-0.1
10	77.4	79.8	+2.4	30.6	32.1	+1.5	11.5	12.0	+0.5
15	83.0	82.6	-0.4	34.5	40.7	+6.2	13.0	13.0	0.0
20	85.1	87.2	+2.1	36.7	41.7	+5.0	14.2	14.9	+0.7
25	86.3	90.3	+4.0	38.4	43.5	+5.1	16.3	17.1	+0.8

Table 2. Interleaved active metric learning (IAML) in comparison to the batch active metric learning (BAML) both for Mar+Gra sampling.

SVM. Fig. 2 shows accuracy for random sampling (*Rand*) and *Mar+Gra* – the best sampling strategy from Sec. 6.1 – before and after metric learning. The first row contains results of LP and the second row for SVM. Again, we show the average over 5 runs including standard deviation for different number of labels.

We also observe a consistent improvement for LP and SVM when applying BAML. For IM100 with 10 labels per class, we increase our performance with LP from 15.9% (*Rand*) to 17.5% (*Mar+Gra*) to 19.9% (*Rand+ML*) to 20.7% (*Mar+Gra*), and with SVM from 17.1% (*Rand*) to 19.2% (*Mar+Gra*) to 21.7% (*Rand+ML*) to 23.3% (*Mar+Gra+ML*). For datasets with a small number of classes, i.e., ETH and C-PASCAL, active sampling is more important than metric learning that is contrary to the previous subsection. The reason is that these methods benefit from their regularization during the learning while the KNN performance is directly connected to the neighborhood structure. But for datasets with a large number of classes like IM100, metric learning is still more important because there are more constraints to fulfill. Another interesting point turns out when looking at the SVM results. For a small number of labels, SVM benefits more from metric learning although this algorithm learns a metric by itself. This can be seen in particular for ETH and IM100.

6.3 Interleaved active metric learning (IAML)

In this subsection, we show 1-NN results in Tab. 2 for the interleaved active metric learning (IAML) when using our best active sampling strategy *Mar+Gra*. In average, we observe an additional improvement that tends to be higher the more labels we use. For example, C-PASCAL with 15 labels is increased by 6.2% from 34.5% (BAML) to 40.7% (IAML). In few cases, we also observe a decrease in performance in particular for a small number of labels that can be explained by a drifting effect. In all experiments we recover from these issues for $|L| > 15$.

7 Conclusion

We present an active metric learning approach that combines active sampling strategies with metric learning. While a first version (BAML) of the approach operates in batch mode and already allows to learn better metrics from fewer training examples, our second version (IAML) interleaves active sampling and metric learning even more tightly which leads to further performance improvements by providing better feedback to the active sampling strategy. Our analysis of different sampling criteria and their influence on the KNN performance shows the importance of choosing an appropriate sampling scheme for metric learning. While we show consistent improvements over a random sample selection baseline, a combination of density and uncertainty-based criteria performs best on average. Finally, we improve also results for different supervised as well as semi-supervised classification algorithms. All our experiments are carried out on three challenging object class recognition benchmarks, where our new approaches consistently outperform random sample selection strategies for metric learning leading to improvements of up to 23% for KNN.

References

1. Baram, Y., El-Yaniv, R., Luz, K.: Online Choice of Active Learning Algorithms. JMLR 5, 255–291 (2004)
2. Basu, S., Banerjee, A., Mooney, R.: Active Semi-Supervision for Pairwise Constrained Clustering. In: SIAM (2004)
3. Cebron, N., Berthold, M.R.: Active learning for object classification: from exploration to exploitation. DMKD 18(2), 283–299 (2009)
4. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2(3), 1–27 (2011)
5. Dasgupta, S., Hsu, D.: Hierarchical sampling for active learning. In: ICML (2008)
6. Davis, J., Kulis, B., Jain, P., Sra, S., Dhillon, I.: Information-theoretic metric learning. In: ICML (2007)
7. Ebert, S., Fritz, M., Schiele, B.: Pick your Neighborhood Improving Labels and Neighborhood Structure for Label Propagation. In: DAGM (2011)
8. Ebert, S., Fritz, M., Schiele, B.: RALF : A Reinforced Active Learning Formulation for Object Class Recognition. In: CVPR (2012)
9. Goldberger, J., Roweis, S., Hinton, G., Salakhutdinov, R.: Neighbourhood Components Analysis. In: NIPS (2005)
10. Guillaumin, M., Verbeek, J., Schmid, C.: Multiple Instance Metric Learning from Automatically Labeled Bags of Faces. In: ECCV (2010)
11. Kulis, B., Jain, P., Grauman, K.: Fast Similarity Search for Learned Metrics. PAMI 31(12), 2143–2157 (2009)
12. Malisiewicz, T., Gupta, A., Efros, A.: Ensemble of Exemplar-SVMs for Object Detection and Beyond. In: ICCV (2011)
13. Nguyen, H.T., Smeulders, A.: Active learning using pre-clustering. In: ICML (2004)
14. Settles, B., Craven, M.: An analysis of active learning strategies for sequence labeling tasks. In: Emp. Meth. in NLP (2008)
15. Straka, M., Hauswiesner, S., Ruether, M., Bischof, H.: Skeletal Graph Based Human Pose Estimation in Real-Time. In: BMVC (2011)
16. Tong, S., Koller, D.: Support Vector Machine Active Learning with Applications to Text Classification. JMLR 2, 45–66 (2001)
17. Vedaldi, A., Fulkerson, B.: VLFEAT: An Open and Portable Library of Computer Vision Algorithms (2008), http://www.vlfeat.org/
18. Yang, L., Jin, R., et.al.: Bayesian active distance metric learning. In: UAI (2007)
19. Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B.: Learning with Local and Global Consistency. In: NIPS (2004)