

Pick your Neighborhood – Improving Labels and Neighborhood Structure for Label Propagation

Sandra Ebert^{1,2}, Mario Fritz¹, and Bernt Schiele¹

¹ MPI Informatics, Saarbrücken, ² TU Darmstadt

Abstract. Graph-based methods are very popular in semi-supervised learning due to their well founded theoretical background, intuitive interpretation of local neighborhood structure, and strong performance on a wide range of challenging learning problems. However, the success of these methods is highly dependent on the pre-existing neighborhood structure in the data used to construct the graph. In this paper, we use metric learning to improve this critical step by increasing the precision of the nearest neighbors and building our graph in this new metric space. We show that learning of neighborhood relations before constructing the graph consistently improves performance of two label propagation schemes on three different datasets – achieving the best performance reported on Caltech 101 to date. Furthermore, we question the predominant random draw of labels and advocate the importance of the choice of labeled examples. Orthogonal to active learning schemes, we investigate how domain knowledge can substantially increase performance in these semi-supervised learning settings.

1 Introduction

Object recognition and scene classification are frequently addressed in computer vision and state-of-the-art methods are dominated by purely supervised learning methods [8, 7]. Yet, there is common agreement that unlabeled data conveys important information of the global data distribution as well as the structure of the classes themselves. Nevertheless, we rarely find approaches successfully tapping into both types of sources that would be able to challenge the best supervised approaches. In a previous investigation, we show that the success of such methods critically depends on the neighborhood relations in the data [4]. This strongly suggests that learning should start before a neighborhood structure is imposed on the data points in order to surpass the inherent limitations of traditional semi-supervised learning schemes.

One might argue that with the availability of crowd sourcing services like Mechanical Turk the value of unlabeled data has shrunken and will ultimately lose its significance. Evidently, there has been a big impact on the vision community as data and labels seem now available in abundance. But recent data collection efforts at those large scales have their own set of problems due to labeling errors and ambiguities [21]. Also adding label information in an unstructured manner will lead to redundant information yielding an inefficient learning scheme. While

active learning has provided useful insights and improvements in this area, the role of domain knowledge has gone largely overlooked.

Contributions: This paper is concerned with the question of how to make better use of the provided labels already in the early stages of popular semi-supervised learning methods. Therefore, our first main contribution is to employ a metric learning approach to improve the graph construction which leads to a consistent improvement in performance. As second main contribution, we propose methods of querying more informative labels based on domain knowledge that are complimentary to traditional active learning settings. Our semi-supervised learning schemes deliver consistent improvements across 3 dataset and show state-of-the-art performance on Caltech-101.

2 Related work

Graph-based methods are a popular choice for semi-supervised learning (SSL) as they are well understood and easy to implement. The way they exploit neighborhood structure is intuitive and the computational demands are usually moderate. One of the key issue of these methods is the construction of the graph. But this critical aspect is often neglected [24] and meaningful neighborhood relations as well as a class structure is assumed to be encoded in the distances of the raw feature space. We have shown in a recent study [4] that for visual categories those assumptions cannot be taken for granted and that the quality of the graph is in fact highly correlated with performance. Thus it is surprising how little attention graph construction [19, 13] has received in comparison to various algorithmic contributions [22, 23]. In [19], the authors uses the neighboring data points to reconstruct each data point from its neighbors. In [13], they propose a method to balance a graph such that dominant nodes are weighted down. All those methods do not use the information which are contained in the labels itself and they are all based on the limiting assumption that the initial feature representation is sufficient for immediate graph construction.

In contrast, metric learning learns a representation better suited to the task at hand. The proposed methods essentially differ in the parameterization of the learned metric (including regularizers and constraints) and optimization procedures. Some methods learn a Mahalanobis distance [3, 14, 17, 9] often with pairwise constraints, while other approaches maximize the inter-class distance by a large margin approach [20]. Although, there are other works combining SSL with some feature transformation [10, 18, 16], this work tightly interleaves a metric learning scheme with label propagation. We use [3] and the follow-up work [14] that show impressive improvements for Caltech 101. Beside the success, it is scalable to large problems in particular in a high dimensional space and it guarantees convergence to the global maximum.

3 Improving neighborhood structure for SSL

As motivated above, we use metric learning [3] to improve our neighborhood structure and apply a graph-based label propagation algorithm [22] on top of this new metric space. Both methods are briefly explained in the following. As shown in sec. 5 the proposed combination of these two techniques leads to improved results over either technique alone, outperforming previously published results e.g. for Caltech 101 using the same underlying image representation [14].

Information theoretic metric learning (ITML): [3] optimizes the Mahalanobis distance between each point pair $x_i, x_j \in \mathbb{R}^d$

$$d_A(x_i, x_j) = (x_i - x_j)^T A (x_i - x_j) \quad (1)$$

Eq. (1) reduces to a simple euclidean distance if $A = I$. To learn matrix A , the algorithm minimizes the logdet divergence between a matrix A and an initial matrix A_0 with respect to pairwise similarity and dissimilarity constraints:

$$\begin{aligned} \min D_{ld}(A, A_0) \\ \text{s.t. } d_A(x_i, x_j) \leq b_u \quad (i, j) \in \mathcal{S} \\ d_A(x_i, x_j) \geq b_l \quad (i, j) \in \mathcal{D} \end{aligned} \quad (2)$$

b_u and b_l are upper and lower bound of similarity and dissimilarity constraints. \mathcal{S} and \mathcal{D} are sets of similarity and dissimilarity constraints based on the labeled data. To make this optimization feasible, a slack parameter γ is introduced to control the trade-off between satisfying the constraints and minimizing $D_{ld}(A, A_0)$. The larger γ the more constraints are ignored. The optimization is done by repeatedly Bregman projections of a single constraint per iteration.

One benefit of this optimization scheme is the efficient kernelization with $K = X^T A X$. A proof can be found in [3]. The kernel version has several advantages. The run time depends only on the number of constraints n_c and not on the dimensions d that is critical in a high dimensional space. We can subsample the number of constraints such that $n_c \ll d$ which reduces the costs from $O(d^2)$ to $O(n_c^2)$. Finally, we can easily compute the at most violated constraint per iteration since only matrix additions ($K_{ii} + K_{jj} - 2K_{ij}$) is required and no complex multiplications as in eq. (1) leading to faster convergence.

Label propagation (LP): We use the common and robust method by [22]. Given a labeled set $\{(x_1, y_1), \dots, (x_l, y_l)\}$ and an unlabeled set $\{x_{l+1}, \dots, x_{l+u}\}$ with $n = l + u$ data $x_i \in \mathbb{R}^d$ and l labels $y_i \in \mathcal{L} = \{1, \dots, c\}$, we build a k -nearest

$$\text{neighbor graph } \hat{P}_{ij} = \begin{cases} 1 & \text{if } d_A(x_i, x_j) \text{ is one of the smallest } k \text{ distances of } i \\ 0 & \text{otherwise} \end{cases}$$

that is symmetric, e.g., $P_{ij} = \max(\hat{P}_{ij}, \hat{P}_{ji})$, and weighted with a Gaussian kernel $W_{ij} = P_{ij} \exp\left(\frac{-d_A(x_i, x_j)}{2\sigma^2}\right)$. Based on this graph a normalized graph

$$\text{Laplacian } S = I - D^{-1/2} W D^{-1/2} \text{ with } D_{ij} = \begin{cases} \sum_j W_{ij} & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \text{ is built.}$$



Fig. 1. Left: ETH, middle: C-PASCAL (column 5-8), and right: Caltech 101

For the learning, we split our multi-class problem into c binary problems and get a prediction vector for each class by an iterative procedure

$$Y_m^{(t+1)} = \alpha S Y_m^{(t)} + (1 - \alpha) Y_m^{(0)} \quad (3)$$

with $1 \leq m \leq c$ and Y_m^* the limit of this sequence. Parameter $\alpha \in (0, 1]$ controls the overwriting of the original labels. The final prediction is obtained by $\hat{Y} = \operatorname{argmax}_{1 \leq m \leq c} Y_m^*$.

4 Datasets and representation

We analyze three datasets with increasing number of object classes and different difficulty. Some of the images are shown in Fig. 1.

ETH-80 (ETH) [15] contains 3,280 images divided in 8 object classes and 10 instances per class. Each instance is photographed from 41 viewpoints in front of a uniform background.

We propose Cropped PASCAL (C-PASCAL) in [4] where we use the bounding box annotations of the PASCAL VOC challenge 2008 training set [5] to extract the objects such that classification can be evaluated in a multi-class setting. The resulting data set contains 4,450 images of aligned objects from 20 classes but with varying object poses, challenging appearances, background clutter, and truncation. For the data representation of both datasets, we also use a HOG [2] representation with cells of 8×8 pixels.

Caltech 101 [6] is a dataset with 9,144 images and 101 object classes. Objects are located in the middle of the image, but there is still background clutter and a large intra-class variability. As a representation we use the same kernel as in [12] (obtained from the authors), which uses an average of four kernels: two kernels based on the geometric blur descriptor, Pyramid Match Kernel (PMK) and the Spatial PMK using SIFT features [11].

5 Evaluation of metric learning for label propagation

In this section, we show first the performance on all three datasets and compare our results with the k-nearest neighbors results (KNN) given by [14]. Next, we give some insight in the learned metric and the resulting neighborhood structure. Based on these observations, we propose a new propagation scheme – *Interleaved Metric Learning and Propagation (IMLP)* – by continuously adding unlabeled data. Finally, we show results on Caltech 101 that outperform the state-of-the-art for 5 training samples.

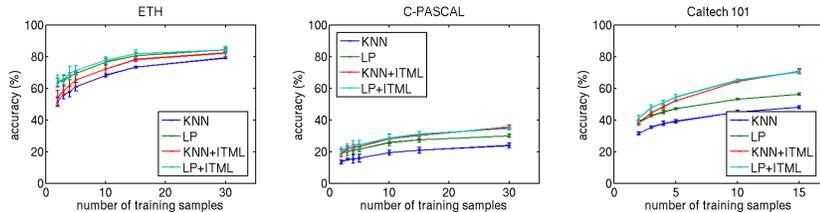


Fig. 2. Overall accuracy for different number of training samples. left: ETH, middle: C-PASCAL, and right: Caltech 101

Metric learning for label propagation: In all experiments, we use the kernelized version of ITML with a gaussian kernel. Only for ETH we report plain metric learning as we didn’t observe any increased performance. Parameter σ for the kernel and the slack parameter γ are set empirically. For the number of nearest neighbor k we choose always the best for each algorithm. All experiments were repeated 5 times with random splits. In tab. 1, results for k nearest neighbor classifier before (KNN) and after (KNN+ITML) metric learning, and label propagation before (LP) and after (LP+ITML) are shown for 5 training samples per class. First, KNN+ITML (col. 3) is always better than KNN (col. 2). Moreover, there is an increase from 39.1% to 52.2% for Caltech 101. Second, LP (col. 4) is consistently improved by LP+ITML (col. 5), i.e., for Caltech 101 from 47.1% to 54.5%. Finally, all LP+ITML results are better than KNN+ITML due to the additional information from the unlabeled data. This leads to an improvement of 2.3% for Caltech 101 in comparison to [14]. The same observation holds true when we vary the number of training examples as in fig. 2. The light blue curve (LP+ITML) is for all 3 datasets above all other curves. It is noteworthy to mention that both LP curves are above all KNN results for ETH.

Discussion and analysis: In fact, the precision of our neighborhood structure increases. This is also illustrated in fig. 3 for C-PASCAL. The first nearest neighbors of a query image (1st col.) are shown before ITML (first row) and after ITML (second row). True positives are outlined with green. Indeed, training examples of a class are pushed close together. But ITML tends to overfit to the training samples. This is more obvious when we split the quality of k nearest neighbors into labeled (NN_L) and unlabeled (NN_U) quality, i.e., the number of true positives within the k nearest neighbors. Fig. 4 shows these qualities for different number of neighbors k . In particular for C-PASCAL and Caltech 101, where we use a Gaussian kernel, NN_L increases up to 95% – 100% for $k = 1$ while the effect on NN_U is substantially smaller.

dataset	KNN	KNN+ITML	LP	LP+ITML
ETH	61.0 ± 2.6	69.3 ± 0.8	65.3 ± 4.7	71.4 ± 3.0
C-PASCAL	15.8 ± 2.6	23.0 ± 1.6	21.5 ± 1.6	24.2 ± 2.7
Caltech 101	39.1 ± 1.1	52.2 ± 0.5	47.1 ± 0.6	54.5 ± 1.7

Table 1. Overall accuracy for all datasets and 5 training samples

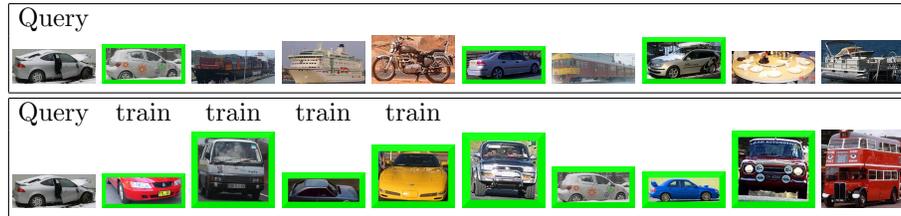


Fig. 3. First nearest neighbors of a query image (1st column) of C-PASCAL. Top: before ITML and bottom: after ITML. True positives are outlined with green and training samples are marked with “train”.

Interleaved Metric Learning and Propagation (IMLP): Based on this observation, we address the lack of generalization by incorporating few predictions from unlabeled data. We propose an iterative procedure with interleaved metric learning and label propagation. This improves incrementally the nearest neighbor precision with the condition that the manifold structure given by the unlabeled data is taken into account. The resulting procedure is as follows:

1. metric learning to get kernel K
2. label propagation with kernel K to obtain predictions \hat{Y} of unlabeled data
3. choose $m = m + n_s$ data points x_i such that $|\tilde{y}_1| \geq \dots \geq |\tilde{y}_i| \geq |\tilde{y}_{i+1}| \geq \dots \geq |\tilde{y}_m|$ with $\tilde{Y} = \max_{1 \leq j \leq c} Y_j^*$ and $l < i \leq u$
4. construct new sets of similarities \mathcal{S} and dissimilarities \mathcal{D} from l labels and m predicted labels, and go to step 1.

Tab. 2 shows results for Caltech 101, 5 training samples, and $n_s = 200$. We improve our results of LP+ITML to 58.7% that goes beyond existing best known numbers of 56.9% by Boiman[1] and 54.2% by Gehler[8]. Also, the performance of KNN+ITML increases to 59.1%. The better performance in comparison to LP+ITML can be explained by incorporating more structure from unlabeled data. Finally, we also get a small improvement for C-PASCAL even though not as much as for Caltech 101 due to lower prediction quality, and almost no improvement for ETH.

6 Selection of training data based on domain knowledge

While the previous section was concerned with algorithmic improvements, we now want to shift the focus to the importance of selecting good training examples

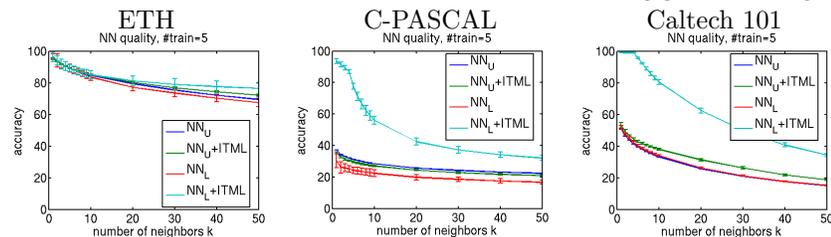


Fig. 4. Nearest neighbor quality splitted into labeled and unlabeled quality

dataset	KNN+ITML	LP+ITML
original	52.2 ± 0.5	54.5 ± 1.7
with predictions	59.1 ± 0.7	58.7 ± 0.8

Table 2. Overall accuracy for Caltech 101 and 5 training samples on our original setting and with predictions

for semi-supervised learning algorithms. As those methods tend to operate in a regime where only a few labels are available, a random strategy can easily pick a set of atypical examples or simply provide poor coverage of the class and/or viewpoint variation. We illustrate these issues in fig. 5, where we provide a more detailed analysis for the class “car” from our C-PASCAL experiment. The first column shows the best random draw w.r.t. average precision (PASCAL VOC criteria) of the retrieved unlabeled examples. We observe a good coverage of intra-class variation and view-points. The next column shows the worst draw. Atypical examples, less viewpoint variation, and truncation have led to a drop in precision from 28.6% to 13.7%! Next we selected 5 prototypical examples by hand to convey our domain knowledge of cars, which results in a performance of 22.4%, right in between the best and worst results of a random draw. To take a step towards an automated approach, we also seek prototypical examples in a statistical sense by finding modes in the distribution of the car examples. Please note that this is best-case type analysis as we are finding the modes for the cars isolated from the other classes. However, this leads to a performance of 35% which is over 6% better than the best random draw we have and over 20% better than the worst one. This large margin emphasizes the potential of selecting appropriate labeled examples opposed to a random draw. The last two columns represent draws from a method we are going to present in this section, that almost recover the best-case performance. In the following, we address the sampling process in an unsupervised manner by using graph properties. The results on all 3 datasets show an improvement for both precision and robustness. In the last experiment, we look at ETH where we use the viewpoint information to obtain better distributed and more representative training examples.

Towards indentifying prototypical instances: In our first experiment, we build a graph based on our kernels and use the intrinsic graph structure to identify highly connected nodes or nodes with a high weight. The intuition behind is that representative images for a class are usually well integrated in the graph and form almost a clique with other similar images, e.g., these nodes have many edges ($\gg k$) with high weights. Our goal is to find such key images. To eliminate images that have many neighbors but only with low weights, we normalize this term by the number of edges:

$$\frac{\sum_i W_{ij}}{\sum_i P_{ij}} > thresh \quad (4)$$

We set *thresh* in our experiments to 0.6. For C-PASCAL, this reduces the number of possible selected images from 222 on average to 136 images per class. Tab. 3 shows the performance with and without thresholding for all three datasets (row 1-2) and 5 training examples. Again, we have an improvement for

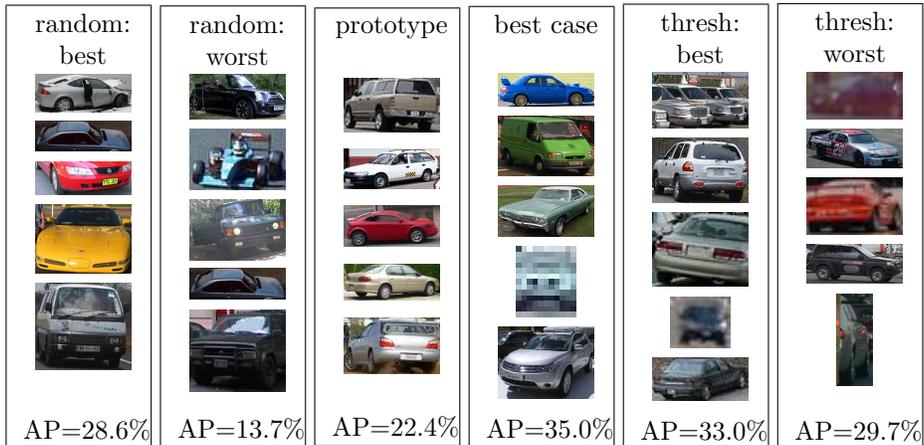


Fig. 5. Training samples of C-PASCAL: random best and worst seed (column 1-2), prototypical selection, and best case estimation (4th column), and with threshold on the graph structure for best and worst seed (column 5-6). AP is the average precision for this class calculated by the PASCAL VOC criteria.

all datasets. For C-PASCAL, we increase the performance of LP+ITML from 24.2% to 25.6% while decreasing the standard deviation from 2.7% to 2.0%. Fig. 5 shows the according training samples for the best and the worst seed in the last two columns. It stands out that the average precision (AP) of the worst seed of thresholded sampling is higher with 29.7% than the best random sampling AP with 28.6%.

To get an idea what we can achieve in an almost best case scenario, we build a graph with $k = 50$, and calculate for each node the number of correct neighbors. We apply k-means clustering for each class to get 5 clusters. Finally, we choose for each cluster the image with the highest nearest neighbor accuracy. This procedure ensures both class coverage and high precision. The results are shown in tab. 3 last row and the corresponding training examples for C-PASCAL are in

dataset	sampling	KNN	KNN+ITML	LP	LP+ITML
ETH	random	61.0 ± 2.6	65.3 ± 4.7	69.3 ± 0.8	71.3 ± 3.0
	threshold	63.8 ± 2.0	67.4 ± 1.0	72.9 ± 2.8	73.0 ± 3.3
	best case	68.5	73.5	81.1	82.6
C-PASCAL	random	15.8 ± 2.6	21.5 ± 1.6	23.0 ± 1.6	24.2 ± 2.7
	threshold	19.0 ± 1.1	23.5 ± 1.0	24.7 ± 1.9	25.6 ± 2.0
	best case	30.1	30.4	36.2	36.4
Caltech 101	random	39.1 ± 1.1	47.1 ± 0.6	52.2 ± 0.5	54.5 ± 1.7
	threshold	40.3 ± 0.6	47.3 ± 1.1	53.3 ± 1.2	55.5 ± 0.9
	best case	45.7	53.9	57.9	59.9

Table 3. Overall accuracy of different sampling methods – random sampling, with threshold, and a best case estimate – for 5 training samples.

long.	width	KNN	KNN+ITML	LP	LP+ITML
random	random	61.0 ± 2.6	65.3 ± 4.7	69.3 ± 0.8	71.3 ± 3.0
90°	360°/5	64.0 ± 2.2	65.5 ± 3.5	74.9 ± 2.6	74.2 ± 3.6
68° – 90°		63.7 ± 2.8	66.2 ± 2.3	73.9 ± 2.5	74.4 ± 3.1
45° – 90°		65.4 ± 2.3	68.3 ± 2.7	73.3 ± 1.7	75.2 ± 2.4
35° – 90°		64.3 ± 1.9	68.2 ± 1.5	73.6 ± 2.2	76.5 ± 2.0
22° – 90°		62.2 ± 3.5	67.1 ± 3.1	69.8 ± 3.1	71.8 ± 3.6

Table 4. Overall accuracy of ETH for different viewpoint sampling methods in comparison to our random baseline (first line)

fig. 5 (col. 4). It is obvious that there is a huge potential in selecting the “right” training samples. While we improve the performance of LP+ITML of C-PASCAL from 24.2% to 36.4%, we also increase the difference to KNN+ITML from 2.7% to 6% that suggests a large unused potential in the underlying structure.

Towards indentifying prototypical viewpoints: For our second experiment, we use domain knowledge in terms of viewpoint information. Each object in ETH is captured from 41 different viewpoints with varying degrees on both the longitudinal axis from 0° to 90° and the width axis with 360°. We split the width axis with 360° into 5 parts and sample one example from each of these areas. Additional, we increase the radius on the longitudinal axis starting from 90° to 22°. The larger the range the more objects from above are sampled.

In Tab. 4 are the results in comparison to our random baseline (first line). All sampling methods based on domain knowledge (row 2-6) lead to a higher performance and a lower standard deviation in comparison to the baseline with 71.3% that contains many images photographed from above. Our best result for LP+ITML with viewpoint information is 76.5%.

7 Conclusion

In this work, we use metric learning to enhance our nearest neighborhood structure that is key for graph-based algorithms and their performance. We show a consistent and significant improvement on three different datasets, and give insights into the learned metric space. We propose a second label propagation scheme – Interleaved Metric Learning and Propagation (IMLP) – that leads to the best published performance on Caltech 101 to date. Finally, we use domain knowledge to sample training data for the semi-supervised framework, and point out the potential in comparison to the common random sampling strategy.

In future work, we intend to make this approach scalable to large image collections like ImageNet since we have not yet exploited all information contained in massive data sets. It would also be interesting to explore other domain-specific or structure knowledge to get better and more representative training samples that require less supervision.

References

1. Boiman, O., Shechtman, E., Irani, M.: In defense of Nearest-Neighbor based image classification. In: CVPR (2008)
2. Dalal, N., Triggs, B.: Histograms of Oriented Gradients for Human Detection. In: CVPR (2005)
3. Davis, J.V., Kulis, B., Jain, P., Sra, S., Dhillon, I.S.: Information-theoretic metric learning. In: ICML (2007)
4. Ebert, S., Larlus, D., Schiele, B.: Extracting Structures in Image Collections for Object Recognition. In: ECCV (2010)
5. Everingham, M., Van Gool, L., Williams, C.K.: The PASCAL VOC (2008)
6. Fei-Fei, L., Fergus, R., Perona, P.: One-shot learning of object categories. PAMI 28(4), 594–611 (2006)
7. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. PAMI 32, 1627–1645 (2010)
8. Gehler, P., Nowozin, S.: On feature combination for multiclass object classification. In: ICCV (2009)
9. Goldberger, J., Roweis, S., Hinton, G., Salakhutdinov, R.: Neighbourhood components analysis. In: NIPS (2005)
10. Grabner, H., Leistner, C., Bischof, H.: Semi-supervised On-line Boosting for Robust Tracking. In: ECCV (2008)
11. Grauman, K., Darrell, T.: The Pyramid Match Kernel : Discriminative Classification with Sets of Image Features. In: ICCV (2005)
12. Jain, P., Kapoor, A.: Active learning for large multi-class problems. In: CVPR (2009)
13. Jebara, T., Wang, J., Chang, S.F.: Graph construction and b -matching for semi-supervised learning. In: ICML (2009)
14. Kulis, B., Jain, P., Grauman, K.: Fast Similarity Search for Learned Metrics. PAMI 31(12), 2143–2157 (2009)
15. Leibe, B., Schiele, B.: Analyzing Appearance and Contour Based Methods for Object Categorization. In: CVPR (2003)
16. Lu, Z., Jain, P., Dhillon, I.S.: Geometry-aware metric learning. In: ICML (2009)
17. Saenko, K., Kulis, B., Fritz, M., Darrell, T.: Adapting visual category models to new domains. In: ECCV (2010)
18. Teramoto, R.: Prediction of Alzheimer’s diagnosis using semi-supervised distance metric learning with label propagation. *Comp. Biol. and Chem.* 32(6), 438–441 (2008)
19. Wang, F., Zhang, C.: Label propagation through linear neighborhoods. *KDE* 20(1), 55–67 (2008)
20. Weinberger, K.Q., Saul, L.K.: Distance Metric Learning for Large Margin Nearest Neighbor Classification. *JMLR* 10, 207–244 (2009)
21. Welinder, P., Branson, S., Belongie, S., Perona, P.: The multidimensional wisdom of crowds. In: NIPS (2010)
22. Zhou, D., Schölkopf, B., Bousquet, O., Lal, T.N., Weston, J.: Learning with Local and Global Consistency. In: NIPS (2004)
23. Zhu, X., Lafferty, J.: Harmonic mixtures: combining mixture models and graph-based methods for inductive and scalable semi-supervised learning. In: ICML (2005)
24. Zhu, X.: Semi-supervised learning literature survey. Tech. rep., University of Wisconsin-Madison (2006)