Environment-adaptive Learning: How Clustering Helps to Obtain Good Training Data

Shoubhik Debnath[1,2], Shiv Sankar Baishya[1,2], Rudolph Triebel[1], Varun Dutt[2], and Daniel Cremers[1]

[1] Computer Vision Group, Technical University Munich, Germany
[2] Indian Institute of Technology Mandi, India
{debnath,baishya,triebel,cremers}@in.tum.de
varun@iitmandi.ac.in

Abstract. In this paper, we propose a method to combine unsupervised and semi-supervised learning (SSL) into a system that is able to adaptively learn objects in a given environment with very little user interaction. The main idea of our approach is that clustering methods can help to reduce the number of required label queries from user interaction, and at the same time provide the potential to select useful data to learn from. In contrast to standard methods, we train our classifier only on data from the actual environment and only if the clustering gives enough evidence that the data is relevant. We apply our method to the problem of object detection in indoor environments, for which we use a region-of-interest detector before learning. In experiments we show that our adaptive SSL method can outperform the standard non-adaptive supervised approach on an indoor office data set.

Keywords: Semi-supervised learning, active learning

1 Introduction

Current machine perception systems often rely on their capabilities to automatically learn a mapping from the set of potential observations to a set of semantic annotations, for example class labels from a natural language. The biggest challenges for the employed learning algorithms are the large amount of labelled data they usually require, and their potential to adapt to new, unseen environments and situations. In many applications, and particularly in mobile robotics, this adaptability is an important requirement, because it is impossible to anticipate all situations that the robot might encounter before deployment. Therefore, we investigate learning mechanisms that are capable of adapting to new observations by updating their internal representation as new information arrives. This implies that the learning step is performed during operation of the system and not beforehand, and that the data used for training is acquired online. However, the main question is: what are good data to train on? A good answer to this

question directly leads to a shorter training time and in a reduced amount of required human data annotations.

In this paper, we address this question using a simple, but effective idea: Before asking the human supervisor for a semantic label, we group the observed data into clusters using unsupervised learning. Then, our algorithm queries one common label for each cluster from the supervisor and uses the so obtained training data in a semi-supervised learning step. This approach has two major advantages: first, it further reduces the amount of human intervention significantly by asking labels for multiple instances at the same time. And second, it gives us the potential to pre-select interesting data to train on, for example by asking labels only for clusters that are significantly represented. We apply our method to the problem of object detection in indoor office environments, and we show in experiments that this adaptive way of learning can outperform the standard approach, where a purely supervised classifier is learned before observing the actual test data.

2 Related Work

Our work is mostly related to the area of semi-supervised learning (SSL) and transductive learning methods, which have become very popular in the last decade. A good overview of this field is given by Zhu [1,2], who also proposed a graph-based SSL method named Label Propagation. Other methods include the sparse Gaussian Process classifier with null category noise model [3], semi-supervised boosting [4] and the transductive Support Vector Machine (tSVM) [5]. In our work, we also use unsupervised learning as in [6] and combine it with a tSVM to reduce the required interaction with the human supervisor even further. Example applications of SSL in computer vision include image classification from labelled and unlabelled, but tagged images [7], object recognition [8], and video segmentation [9].

Furthermore, our work is also related to the area of active learning, because it involves a user interaction step, for which queries for class labels are actively generated. A good overview on the active learning literature is given by Settles [10]. One interesting example of active learning is the work of Kapoor *et al.* [11] on object categorization using a GP classifier (GPC), where data points possessing large uncertainty (using posterior mean and variance) are queried for labels and used to improve the classification. Triebel *et al.* [12] use active learning for semantic mapping where a sparse GP classifier actively learns to distinguish traffic lights from background. In contrast to classical active learning methods, our approach chooses the data to be asked for labelling based on a relevance criterion rather than, e.g. based on the entropy of the underlying classifier.

3 Combined Unsupervised and Semi-Supervised Learning

Fig. 1 gives an overview of our proposed semi-supervised learning method. We start with a sequence of input images and determine first an appropriate set of

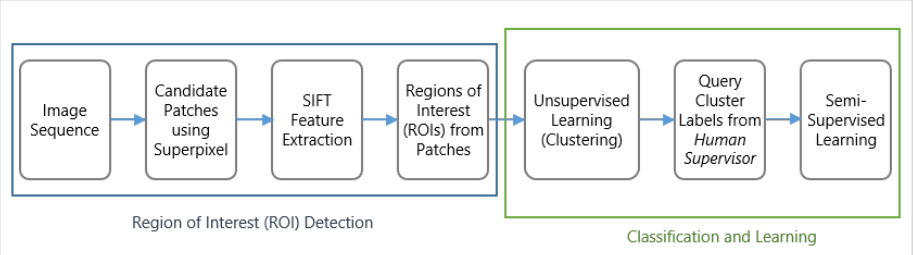

Fig. 1. Flow chart of our proposed system. From a sequence of images, regions of interest are detected using super pixel segmentation and by comparing the segments based on SIFT features. Then the resulting patches are clustered. From each cluster, a subset of patches is used to query object labels from a human supervisor. The resulting hand-labelled data together with some unlabelled samples is then used to train a semi-supervised classifier.

rectangular regions of interest named *patches*. From these patches, we extract SIFT features ("Scale-invariant feature transform", [13]) and use them to define a similarity measure between patches. Based on these similarities, we cluster the patches using spectral clustering. Then, we select a subset of appropriate patches from each cluster and query object labels from a human supervisor as described below. The resulting labelled patches, together with the remaining unlabelled ones are then passed into a multi-class transductive SVM, which then returns predicted labels for the unlabelled patches. In the following sections we describe each step in more detail and give motivations for our algorithm design.

3.1 Region of Interest Detection

Object detection for a given image of a scene is much harder than pure object recognition, because it is not even known to the algorithm *if* the object to be recognized exists in the scene and *where* it is. The common approach to this problem is to determine small sub-windows within the image which potentially contain the object(s) to be classified. In the simplest case, these so-called *regions of interest* (ROI) are obtained using a sliding-window approach. However, to reduce the number of potential ROIs, we use a different method: Given an image sequence, we first compute a superpixel segmentation for each image based on the SLIC algorithm [14]. Then, we compute the bounding box for each segment in every image. For each such resulting candidate patch, we extract SIFT features [13] and compare the patches across the image sequence using a similarity measure s. The motivation for the choice of SIFT descriptor is their high expressive power and their ability to find good matches even under changes of illumination, orientation and scale. In our application, object instances do not vary much in color or texture, which is an ideal condition for the SIFT descriptor. Of course, in a more general setting, where the appearance between the objects of a class

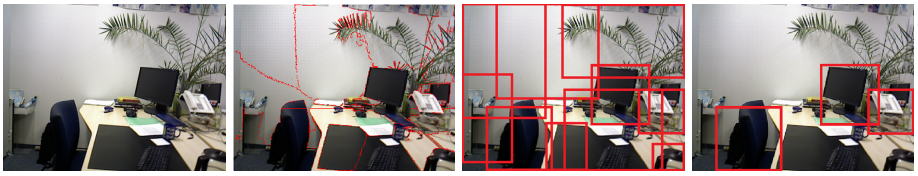

Fig. 2. Example result of our ROI detector. From left to right: 1. Original image 2. SLIC superpixels with boundaries in red, 3. Bounding boxes of the super pixels, 4. Detected ROIs after threshold.

may vary more, other descriptors, for example based on the geometry may be more appropriate.

To compute the similarity measure s, we first define a distance function d between two patches A and B as:

$$d(A,B) = \frac{1}{n}\sum_{i=1}^{n} \|\tilde{\mathbf{x}}_i - \tilde{\mathbf{y}}_i\|^2 , \tag{1}$$

where n is the number of matches found by the SIFT algorithm and i iterates over all these matches. The vectors $\tilde{\mathbf{x}}_i$ and $\tilde{\mathbf{y}}_i$ denote the 128-dimensional descriptor values computed at the key points found by the SIFT method in patches A and B, respectively. From this distance measure, we define the similarity s between two patches as:

$$s(A,B) = 1 - \frac{d(A,B)}{\max\limits_{A',B'} d(A',B')}, \tag{2}$$

thus, s gives values between 0 and 1, where 1 corresponds to maximal similarity. To find patches that contain potentially interesting objects, we compute a similarity score p for patch A as follows:

$$p(A) = \sum_{B \neq A} s(A,B), \tag{3}$$

i.e. the score is defined by the sum of similarities to all other patches. The intuition here is that patches that are very similar to many others more likely contain objects of interest, because they give evidence that there are many instances of the same object class. Note that our formulation implicitly deals with the problem that background patches containing walls, the floor, etc., despite occurring very often will not give a high score, because their appearance is usually much more uniform, which means that much less SIFT key points are detected on them.

Using these score values, an ROI is then detected as the patches A for which $p(A)$ exceeds the average score over an entire image. This simple statistical method finds patches that stick out in terms of their similarities and has the advantage that it does not require to introduce a threshold parameter. In our experiments, this gave good results (see Fig. 2 for an example sequence of our detector), but of course other methods could be used here.

3.2 Clustering of Patches

The main contribution of our work is the idea of using unsupervised learning *before* employing a semi-supervised method for classification. The motivation of this approach is two-fold: first, the number of required user interactions, i.e. label queries, is further reduced compared to standard semi-supervised learning, because we query only one common label for an entire group (cluster) of data instances. And second, the clustering step gives us the opportunity to pre-select interesting data to train on, because typically some clusters can be easily identified as more relevant for the learning task based on simple characteristics such as cluster size or similarities of elements within a cluster. The intuition here is that only those data instances should be learned by the classifier, for which there is enough evidence that they correspond to a meaningful object class. For example, in an office environment, usually there are many instances of classes like telephone, chair or monitor, and the mere fact that there are many very similar instances makes them highly relevant, for example for a mobile robotic system operating in the environment. In contrast, in a home environment, there might be other types of relevant objects, and our approach particularly aims at finding such relevant classes adaptively.

To perform the clustering step, we use the same SIFT descriptors computed earlier for each patch and rely on the same similarity measure s to cluster the patches. We ran experiments with two different standard clustering methods: k-means clustering and spectral clustering. Both methods have been used very successfully in many different kinds of applications, and we found that the difference in performance is not very substantial. We evaluated both methods on our data using the V-measure [15], which is defined as the harmonic mean of homogeneity and completeness of the clustering algorithm. In these experiments, the spectral clustering was slightly better, and it has the further advantage that it does not necessarily require the number of clusters specified as a parameter. The reason is that it is based on the eigen decomposition of the graph Laplacian of the data, and that a method called the *eigen gap heuristic* can be used to determine a good value for the number of clusters. For more details on spectral clustering, we refer to the work of Luxburg [16].

3.3 Querying Object Labels

The next step in our proposed method is to receive class label information from a human supervisor for the patches that have been clustered beforehand. To perform this label query, some important considerations need to be taken into account: On one side, the algorithm should ask the user as few times as possible to give a label input, because this is one of the main motivations of this work. Thus, we want to ask only once for each cluster. On the other side, we need to make sure that the data we provide as training samples to the semi-supervised learning method is as *pure* as possible, i.e. ideally there should be no instances of different objects labelled by the human with the same label. Unfortunately, no clustering algorithm can guarantee complete purity, neglecting of course the

trivial clustering that assigns every data point to its own cluster. Therefore, we propose to use a quality measure q for all patches within a cluster, which is based on the similarities s computed earlier. Concretely, for every patch A of a given cluster $\mathcal{C}$, we compute q as the sum of similarities *within the cluster*:

$$q(A) = \sum_{B \in \mathcal{C}} s(A, B). \tag{4}$$

Note that this is different from the scores computed in Eq. (3), because here, our goal is to find the best cluster representatives. After computing the q-values, we sort all elements within a cluster in descending order of q and ask one common label from the user for the first m such elements of each cluster. This policy gives a good trade-off between the two opposing objectives of generating few label queries and providing pure training data. Of course, this method does not guarantee that there are no instances of different object classes that receive the same label from the supervisor. However, from our experience, the number of cases where queried data points are inconsistent can be reduced substantially using this method.

To illustrate this step, Fig. 3 shows an example result of the clustering step, where each row corresponds to a different cluster and only the first 3 elements according to the quality measure q are shown. As we can see, in two out of four cases the first three cluster elements only contain objects of the same class, and in the other two cases the mistakes made by the algorithm are completely comprehensible. We also note that the clustering result yields more clusters than there are actual classes, i.e. we have an over-clustering. This is only a problem in the sense that it requires the user to give more class labels than actually needed, but this effect was only minor in our experiments.

3.4 Training a Classifier

As a final step in our approach, we use the labelled data obtained from the previous step to learn a classifier for the objects discovered in the environment. Here, we considered three different strategies. First, we investigated the use of a standard supervised learning method using a linear Support Vector Machine (SVM). Then, we evaluated two semi-supervised learning techniques, where the first was a simple nearest neighbour rule, i.e. each unlabelled sample was assigned the label of the closest labelled sample according to our similarity measure. And finally, we used a transductive SVM [5] with an RBF kernel. Thus, in addition to the labelled training set $\mathcal{D}$ of size l, the algorithm is also given an unlabelled set $\mathcal{D}^\star = \{\mathbf{x}_i^\star \in \mathbb{R}^p\}_{i=1}^k$ of test examples to be classified. Formally, a transductive SVM is defined by the following primal optimization problem:

Find $(\mathbf{y}_1^\star, \mathbf{y}_2^\star, \ldots, \mathbf{y}_n^\star, \mathbf{w}, \mathrm{b})$ so that

$$\begin{aligned} & \min \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{subject to} \quad & y_i[\mathbf{w} \cdot \mathbf{x}_i - b] \geq 1, \quad y_j^\star[\mathbf{w} \cdot \mathbf{x}_j^\star - b] \geq 1, \\ & y_j^\star \in \{-1, 1\} \quad \forall i = 1, \ldots l, \forall j = 1, \ldots, k \end{aligned} \tag{5}$$

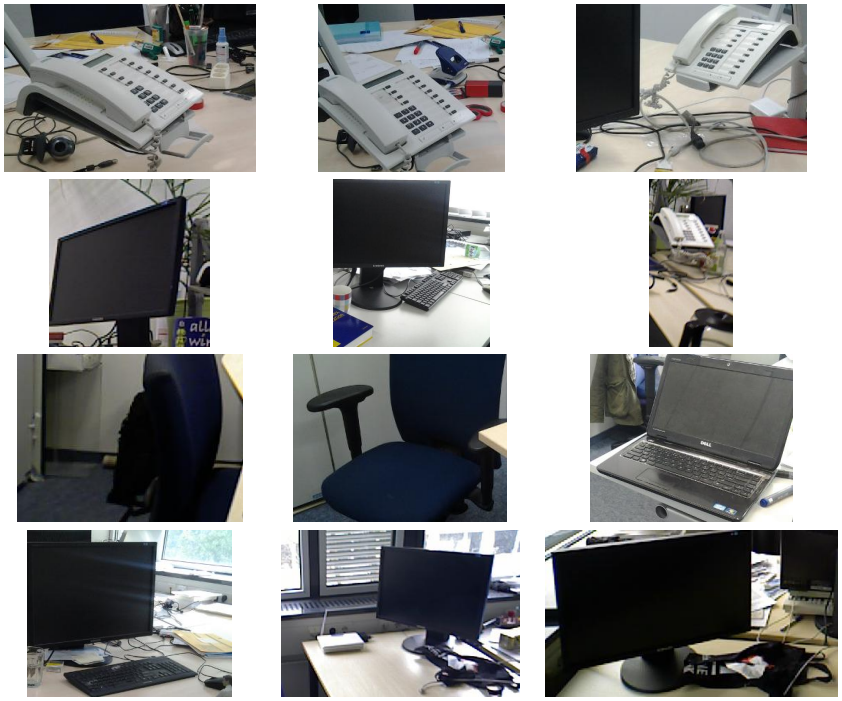

Fig. 3. Examples of clusters obtained from the clustering algorithm (every row corresponds to a different cluster). For each cluster, we show the first three elements according to the quality measure defined in (4).

where $(\mathbf{x}_i, \mathbf{y}_i)$ are the training examples, $\mathbf{y}_i^\star$ are the predicted labels for the unlabelled test example and $\mathbf{w}$ is the weight vector. This means, that the transductive SVM learns from both the labelled and the unlabelled examples, and it returns label predictions for the unlabelled ones. In that sense, the training and the inference step are contained within the same common procedure.

From these three methods the worst in our experiments was the standard supervised SVM, and we did not consider this further. The highest classification performance was obtained with the transductive SVM, and we give more details in the experimental section. As feature vectors for training, we compute for every patch the Hierarchical Matching Pursuit (HMP) descriptor introduced by Bo *et al.* [17]. The HMP features are calculated in a multi-layer process where each layer is computed on a different scale, containing the same three steps: Matching Pursuit, Pyramid Max Pooling and Contrast Normalization. The key element in this process is the Matching Pursuit step, which is based on a sparse coding algorithm known as K-SVD. Given a set of h-dimensional observations $Y = [y^1, ..., y^n] \in R^{h \times n}$ (image patches in our case), K-SVD learns

a dictionary $D = [d^1, ..., d^n] \in R^{h \times m}$, and an associate sparse code matrix $X = [x^1, ..., x^n] \in R^{m \times n}$ by minimizing the following reconstruction error,

$$\min_{x_i} \|y_i - Dx_i\|^2 \text{ s.t. } \|x_i\|_0 \leq K, \tag{6}$$

where x_i are the columns of X, the zero-norm $\|x_i\|_0$ counts the non-zero entries in the sparse code x_i, and K is the sparsity level, which bounds the number of the non-zero entries. The Matching Pursuit step finds an approximate solution to the optimization problem mentioned above using a greedy approach. Pyramid Max Pooling is a non-linear operator that generates higher level representations from sparse codes of local patches which are spatially close. And Contrast Normalization turns out to be essential for good recognition performance, since the magnitude of sparse codes varies over a wide range due to local variations in illumination and foreground-background contrast. Bo *et al.* [17] used a linear SVM in combination with HMP features and reported very good classification results. We verified these results using data from the Caltech 101 benchmark, and we show them in the results section. From this, we conclude that HMP features exhibit a high amount of expressiveness, because they give very good classification results for a comparably simple classifier such as the linear SVM.

In practice, the use of HMP features consists of two phases: one where the dictionaries are learned from some given training data, and one where feature vectors are computed for new test data based on the sparse codes with respect to the learned dictionaries. While the first phase can require huge computation time, as it usually uses a large training data set, the online phase is comparably fast, as it only requires the computation of a sparse representation for a given dictionary. We note however, that the dictionary learning step is completely unsupervised, as it does not require any human-labelled data.

4 Experiments and Results

To measure the performance of our approach, we performed several experiments. First, we evaluated our method to detect regions of interest. Then, we evaluated two different semi-supervised learning methods on a benchmark and on our own data. And finally, we verified experimentally the benefits of using our adaptive, semi-supervised learning method over a standard non-adaptive supervised strategy. More details about all experiments are given in the following.

4.1 Evaluating the ROI Detector

As mentioned above, our ROI detector finds patches that occur often with high similarity across images. Therefore, to assess this method quantitatively, we first created ground truth data for the objects that occurred most frequently in our data. Concretely, we labelled those ROIs as correct detections, which contained chairs, monitors or telephones. Results on 7 different images in terms of precision and recall are given in table 1. We see that our detector tends to find more

	Image 1	Image 2	Image 3	Image 4	Image 5	Image 6	Image 7
Actual ROIs	2	3	2	1	1	2	2
Predicted ROIs	4	4	2	3	3	4	4
Recall	1	0.67	1	1	1	1	1
Precision	0.5	0.5	1	0.33	0.33	0.5	0.5

Table 1. Evaluation of the ROI detector on 7 input images. While the precision is comparably low, recall is good, which is the main purpose of this step.

ROIs than there actually are, and the recall is much better than the precision. However, for ROI detection we are actually more interested in recall than in precision, because missing a candidate for classification is worse than reporting a background patch as a ROI, as the latter can be handled by the classifier.

For a qualitative evaluation, we show an example result of the ROI detector in Fig. 4. As we can see here, the detector found the two regions of actual interest, i.e. the chair and the monitor, and it only returned one false positive.

4.2 Comparison of Adaptive Semi-Supervised Learning and Standard Supervised Learning

To measure the performance of our adaptive semi-supervised learning method, we ran experiments on a subset of the standard benchmark data set Caltech 101, and on our own data. The subset consisted of 10 classes (see Fig. 5), for the Caltech 101 and 3 classes for our data. For both experiments, we used dictionaries for the HMP features that were learned from 10 images per class from the benchmark set. For the Caltech 101 we did not employ the ROI detector, because these images already contain one major object and not much background. Thus, we only clustered the data, computed HMP features for each image and trained a semi-supervised learner on a mixture of labeled and unlabeled images, where the labels were obtained from querying the best 3 representatives of each cluster. The results for the k-nn method and the transductive SVM with RBF kernel are given in the left column of Table 2. As we can see, the transductive SVM performs much better than the k-nn approach, and the final accuracy is

Fig. 4. Example result of our ROI detector. The ground-truth ROIs are shown on the left and the predicted ROIs on the right.

Fig. 5. Examples from each of the 10 classes in the Caltech 101 data set which were used for the experiments.

comparably high, given that only very few data samples used for training were actually labeled.

The same conclusion we can draw for our indoor office data set (see right column of Table 2). Here, we used 25 ROIs for evaluation, consisting of 6 chairs, 13 monitors and 6 telephones. Again, the transductive SVM performs better than the naive k-nn approach. Also, it is interesting to see that supervised learning works well when trained and tested on the same kind of data, but when tested on data from a different environment, it may fail as in our example. To overcome such problems our adaptive SSL method seems to be an appropriate approach.

Note that our adaptive TSVM approach gives somewhat worse results than the standard SVM method on Caltech101. This is because the clustering step for this data set had to be done using the HMP features and not SIFT, as for our own data: the appearances of the objects in Caltech 101 are simply too diverse to compare them using SIFT. However, we experienced that spectral clustering works worse on HMP features, which means that for Caltech 101 the training data provided to TSVM was of less quality than if we had chosen standard supervised learning. For our evaluation, this is however of little importance, as our method anyhow aims at adapting to a given environment with no previously labelled data where objects of the same class are not very diverse. An application of our method to an environment-independent, pre-labelled data set such as Caltech101 is therefore not very meaningful.

4.3 Number of Generated Label Queries

In another experiment, we investigated the correspondence of the number of label queries made by the algorithm and the classification accuracy. There are two parameters that can be set: the number of clusters c and the number m of patches per cluster, which receive a label after the query (see above). On one side, we want to have few clusters, i.e. c should be low. However, if there are more clusters, then the clusters are smaller and therefore *purer*, i.e. there are more elements that agree on the true class label. Purer clusters means that we

Learning method	**Caltech 101**	**Our data**
standard SVM	95.25%	52.00%
adaptive k-nn SSL	55.86%	58.00%
adaptive TSVM	81.43%	88.00%

Table 2. Classification accuracy of standard SVM learning and adaptive SSL methods on different data sets. The standard SVM was trained on a subset of Caltech 101 in both cases. Thus, while standard supervised learning gives good results when training and test data are similar, it can perform badly when they are dissimilar. However, our adaptive SSL performs much better, because it queries the relevant class labels from the data before learning the classifier. From the two considered methods, transductive SVMs perform better than the k-nearest neighbour method.

can increase m, without assigning wrong labels to patches, thus we obtain better training data. This relationship is shown in Fig. 6. If the number of clusters is small, we get the best accuracy for $m = 1$. But for more clusters, $m = 2$ is better, because by assigning the same label to the first m elements of each cluster, we get fewer wrong labels. In general we found that having less labels for training is better than having more, but wrong labels.

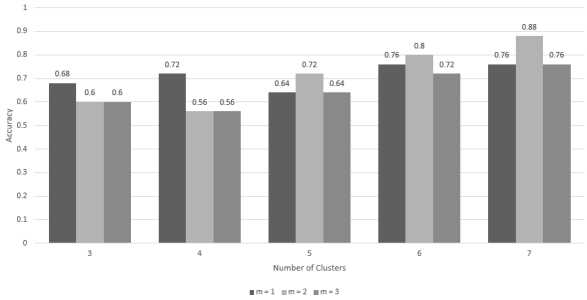

Fig. 6. Accuracy vs. number of clusters and number m (m = 1, 2, 3) of patches receiving a label from the query. More clusters lead to a higher cluster purity. Then, higher values of m are more effective, because the tSVM receives better training data.

5 Discussion and Conclusions

Our proposed approach for adaptive semi-supervised learning for object detection in indoor environments has two major advantages over standard supervised learning methods: first, it is able to select informative data to learn from and to adapt to a given environment by only querying labels for currently observed, situation-relevant data and using them to train a classifier. And second, it reduces the number of required user interactions by making more informed

questions about the data based on a pre-clustering step. Our experiments show that the proposed approach can outperform standard non-adaptive supervised learning when applied to environment-dependent data.

Acknowledgment This work was funded by the EU project SPENCER (ICT-2011-600877).

References

1. X. Zhu, "Semi-supervised learning literature survey," Computer Sciences, University of Wisconsin-Madison, Tech. Rep. 1530, 2005.
2. ——, "Semi-supervised learning with graphs," Ph.D. dissertation, Carnegie Mellon University, 2005.
3. N. D. Lawrence, J. C. Platt, and M. I. Jordan, "Extensions of the informative vector machine," in *Proc. of the First Intern. Conf. on Deterministic and Statistical Methods in Machine Learning*. Springer-Verlag, 2004, pp. 56–87.
4. A. Saffari, C. Leistner, and H. Bischof, "Regularized multi-class semi-supervised boosting," in *Conf. on Comp. Vision & Patt. Recog. (CVPR)*, 2009.
5. T. Joachims, "Transductive inference for text classification using support vector machines," 1999, pp. 200–209.
6. R. Triebel, R. Paul, D. Rus, and P. Newman, "Parsing outdoor scenes from streamed 3d laser data using online clustering and incremental belief updates," in *Robotics Track of AAAI Conference on Artificial Intelligence*, 2012.
7. M. Guillaumin, J. Verbeek, and C. Schmid, "Multimodal semi-supervised learning for image classification," in *Conf. on Comp. Vision & Patt. Recog. (CVPR)*, 2010.
8. S. Ebert, D. Larlus, and B. Schiele, "Extracting structures in image collections for object recognition," in *European Conf. on Comp. Vision (ECCV)*, 2010.
9. I. Budvytis, V. Badrinarayanan, and R. Cipolla, "Semi-supervised video segmentation using tree structured graphical models," *Trans. on Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2751–64, Nov 2013.
10. B. Settles, "Active learning literature survey," Tech. Rep., 2010.
11. A. Kapoor, K. Grauman, R. Urtasun, and T. Darrell, "Gaussian processes for object categorization," *Intern. Journal of Computer Vision*, vol. 88, no. 2, pp. 169–188, 2010.
12. R. Triebel, H. Grimmett, R. Paul, and I. Posner, "Driven learning for driving: How introspection improves semantic mapping," in *The International Symposium on Robotics Research (ISRR)*, 2013.
13. D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. of the Intern. Conf. on Computer Vision (ICCV)*, 1999, pp. 1150–1157.
14. R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "Slic superpixels compared to state-of-the-art superpixel methods," *Trans. on Pattern Analysis and Machine Intelligence*, vol. 34, no. 11, pp. 2274–2282, Nov 2012.
15. A. Rosenberg and J. Hirschberg, "V-measure: A conditional entropy-based external cluster evaluation measure," in *Proc. of the Joint Conf. on Empirical Methods in Natural Language Proc. and Comp. Natural Language Learning(EMNLP-CoNLL)*, 2007, pp. 410–420.
16. U. Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, no. 4, pp. 395–416, 2007.
17. L. Bo, X. Ren, and D. Fox, "Hierarchical matching pursuit for image classification: Architecture and fast algorithms," in *In NIPS*, 2011.