On the Applicability of Unsupervised Feature Learning for Object Recognition in RGB-D Data

Manuel Blum, Jost Tobias Springenberg, Jan Wülfing and Martin Riedmiller
Department of Computer Science, University of Freiburg
{mblum,springj,wuelfj,riedmiller}@tf.uni-freiburg.de

Abstract

We present a feature extraction method for RGB-D data based on k-means clustering that builds on recent work by Coates et al. Using unsupervised learning methods we are able to automatically learn feature responses that combine all available information (color and depth) into one, concise representation. We show that depth information can substantially increase the recognition performance and that the learned features are competitive with previously published results on the *RGB-D Object Dataset*. For object recognition in high resolution RGB-D images we propose to embed the learned features into a local image descriptor, the *convolutional k-means descriptor*. Using this approach recognition accuracy can be increased even further.

1 Introduction

Most prevailing methods for object recognition are still based on hand-designed features, such as SIFT [16] and SURF [2], despite their known limitations, for example the restriction to grayscale images. In order to cope with multi-channel data, these algorithms can be modified manually, but a more general and more flexible strategy is to utilize *learned* features for that purpose. By using unsupervised learning techniques a concise representation of raw input data can be found, that automatically integrates all available information.

Coates et al. [7] showed that very simple unsupervised feature learning algorithms can achieve state-of-the-art performance on the CIFAR [11] and NORB [15] benchmarks, where the best results were accomplished using k-means clustering. In this work we adopt their setting for RGB-D data, where besides the color channels (RGB) an additional depth channel is available. We found that depth information can be seamlessly integrated as an equally important fourth channel to learn features that perform better than those trained on color data only. Moreover, the method was able to outperform modern hand-designed feature descriptors on object recognition tasks using the recently published *RGB-D Object Dataset* [12].

To make the method applicable for high dimensional RGB-D images we combine the learned features into local descriptors, by using a similar feature extraction scheme as in SIFT [16]. In conjunction with feature pooling this gives rise to a local feature descriptor representation. In this paper the descriptors are extracted on a fixed grid, but they could also be extracted around interest points in a bag of words like setting.

Using this descriptor, which is subsequently called the *convolutional k-means descriptor*, we were able to improve on the results of the original learning setup and achieve better classification accuracies than all previously published results [13, 4, 5].

2 Related Work

Hand-designed features based on orientation histograms, such as SIFT [16] and SURF [2], have successfully been used for a variety of applications in computer vision. Previous attempts to adapt these methods to incorporate additional information include the augmentation of the descriptor vector with color histograms [1] and the extraction of SIFT descriptors on grayscale images generated from depth data [12]. More recent methods generalize features based on orientation histograms to a broader class of so called kernel descriptors [5]. Local feature responses are used to extract several kernel responses containing either color or shape information, which are then combined into one descriptor vector. Kernel descriptors can also be extended to capture information present in the depth channel by treating it as a grayscale image and extracting orientation gradient kernels from it. A different approach, considering feature extraction from depth data alone is presented in [18], where feature responses are generated at points where the surface is mostly stable but changes significantly in the immediate vicinity.

Work in the machine learning community gave rise to various different methods for learning low level feature responses from data. Particularly the work on deep belief networks [9] and deep autoencoders [6, 14] resulted in object recognition architectures that achieve excellent performance on important benchmark tasks. Work emphasizing the sparseness of the learned feature representations such as sparse coding [8] and local coordinate coding [19] comprise a related family of algorithms that have been successfully applied to object recognition tasks. While the aforementioned work mostly focuses on improving the unsupervised learning algorithms, it was shown in [7, 8] that the same level of performance can be reached by using simpler algorithms while, at the same time, learning more features. This paper will build on their results, introduce two improvements to the unsupervised learning procedure and propose a local descriptor based on the learned features.

Our approach is similar to work on kernel descriptors [5] in which a local descriptor is built by comparing pixel orientations or color intensities. When using our proposed descriptor we effectively compute a similar feature histogram in the vicinity of interest points. Our feature histogram does, however, not consist of explicitly designed feature responses using pixel comparisons, but is build by comparing a learned representative set of features to image patches around the interest point.

3 Method

Our approach uses the unsupervised feature learning framework described in [7] to learn low-dimensional representations of high resolution RGB-D images. Given a set of input vectors $X = \{x^{(1)}, \ldots, x^{(m)}\}$ with $x^{(i)} \in \mathbb{R}^N$, which are obtained by randomly extracting sub-patches from unlabeled images, we apply k-means clustering to discover characteristic features in the training data. Subsequently, the learned features are used to train a classifier capable of solving the object recognition task.

This framework is adapted in three different ways in order to improve classification performance and to optimize the algorithms convergence. (1) We add depth information as a fourth channel to the image representation. (2) The unsupervised learning algorithm is improved by modifying the preprocessing procedure and introducing a bootstrapping approach. (3) In order to cope with the high dimensionality of the input data, we propose to modfiy the framework by learning features only in the vicinity of interest points and by combining their responses into a feature descriptor (see Chapter 4).

3.1 Pre-processing

Before the feature learning algorithm is applied, several pre-processing steps are performed on the input data X. First, all image patches are normalized by subtracting the mean and dividing by the standard deviation. Afterwards a whitening transformation [10] is applied in order to attain uncorrelated data with unit variance. It was shown in [7] that this step is crucial for the quality of the learned feature responses. Instead of the originally proposed ZCA whitening [3] we use PCA whitening, which opens the opportunity to drop insignificant components of the input data, speeding up the feature extraction and leading to an improved feature quality. By keeping only the first n principal components, each patch $x \in R^N$ is projected to a lower dimensional vector $x' \in R^n$.

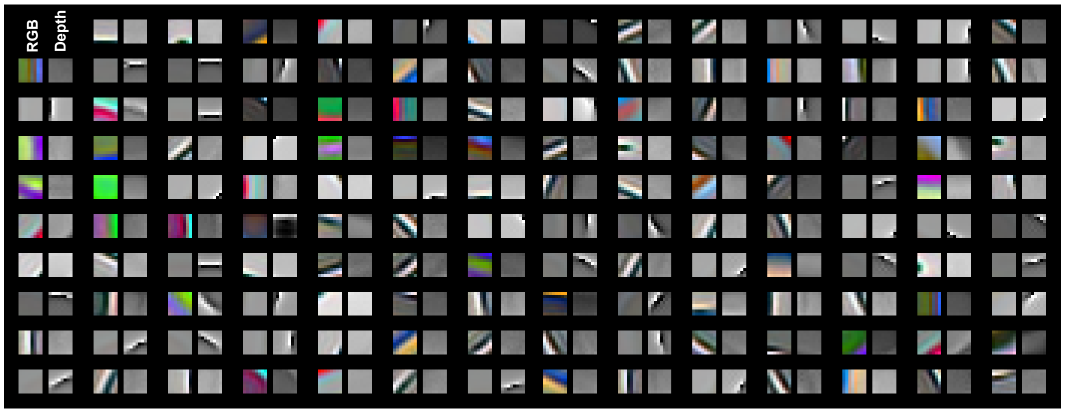


Figure 1: Example features learned by the k-means approach. The displayed 56 features are part of a larger dictionary of 961 features that were learned in our experiments. We visualize the RGB as well as the depth channel for each feature next to each other.

3.2 Unsupervised learning

We use a k-means approach to learn d centroids building the feature response dictionary D by clustering the extracted patches X. Although k-means is a very simple unsupervised learning algorithm that is easy to implement, it has recently been shown that it is competitive to other unsupervised learning algorithms when learning local, low-level features from pre-processed image data [7]. Apart from its simplicity the main advantage of using k-means over other algorithms is that it is very fast and scales well to a large amount of centroids. It can be trivially parallelized on current computer hardware in a map-reduce manner and allows us to learn large, over-complete feature dictionaries that can be expensive to learn using other unsupervised learning approaches.

To further improve upon the feature quality that can be achieved using standard k-means, as well as the required runtime until convergence, we propose to use a bootstrapping learning approach to train the d centroids. This is possible since the data is clustered in PCA whitened space. We first cluster in the subspace spanned by the first p principal components and fill the learned centroids with zeros for all other $c - p$ dimensions. These centroids are then used to *warm start* the clustering procedure in the c dimensional PCA whitened space.

The impact of this procedure is presented in Fig. 2. Without bootstrapping some features are badly localized, which is an artifact of clustering in a high dimensional space (e.g. 144 dimensions if image patches of 6×6 pixel are used). This effect is visible in Fig. 2 where affected features are marked red. When the bootstrapping procedure is used the features are well distributed over the whole feature space by pre-training the centroids on the major principle components. The consecutive clustering procedure in the complete feature space is thus simplified and the badly localized features disappear.

An example set of features that is learned using data from the *RGB-D object dataset* is depicted in Fig. 1.

3.3 Feature response

After the features are learned, different possibilities exist to compute the feature responses $f(x) \in \mathbb{R}^d$ of a given $w \times w$ image patch. The importance of the encoding scheme has been thoroughly discussed in [8] were different encodings schemes for sparse coding and vector quantization were evaluated. In this paper we restrict ourselves to a soft variant of the standard k-means response. The standard hard k-means response $f(x)$ is a sparse vector indicating the closest centroid

$$f_i(x) = \begin{cases} 1 & \text{if } c^i = \arg\min_{c^i \in D} \|c^i - x\| \\ 0 & \text{else} \end{cases} . \tag{1}$$

The soft variant of this response keeps the information about the distance of the current patch to all centroids $c^i \in D$ that are closer than the average distance $\mu(z) = \frac{\sum_{i=1}^{d} z_i}{d}$ where $z \in R^d$ with

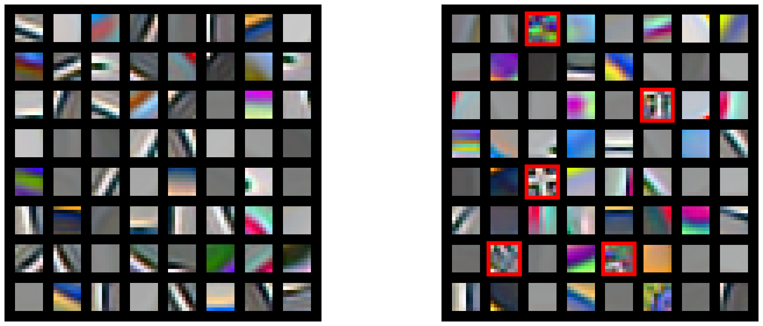

(a) with bootstrapping (b) without bootstrapping

Figure 2: Comparison of features learned from the *RGB-D object dataset* with and without bootstrapping. (a) When bootstrapping is enabled all learned centroids correspond to nicely localized features. (b) Without bootstrapping several cluster centers, marked in red, do not represent good feature responses due to the high dimensionality of the space in which k-means clustering is performed. Note that the depth channel is omitted in this picture to make visual comparison of the feature quality easier. Patches corresponding to features with depth selective responses thus appear in gray.

$z_i = \|c^i - x\|$. This ensures that the vector is still sparse but contains more information than in the hard case. In [7] this is called the triangular response, where $f(x)$ is defined as

$$f_i(x) = \max(0, \mu(z) - z_i). \tag{2}$$

3.4 Feature extraction

In the feature extraction step the input image is processed in a convolutional manner. First the feature response $f(x) \in \mathbb{R}^d$ of each $w \times w$ sub-patch $x \in \mathbb{R}^{w \times w \times 4}$ in the image is computed. We chose $w = 6$ if not denoted otherwise. The feature response is computed by stepping over the image with a fixed stride of s computing $f(x)$ for the image sub-patch centered at the current image position in each step. A schematic of the feature extraction process is depicted in Fig. 3.

After the feature extraction step the image is split into four quadrants and the feature responses within the quadrants are sum-pooled to reduce the dimensionality of the feature representation. If the input images are not of equal size this sum-pooling will result in differently scaled feature representations. One way to account for this problem is to rescale all images to be of equal size. Another possibility which we employ in our experiments is to normalize the sum-pooled feature response of each quadrant by the number of image patches extracted from it.

4 Convolutional k-means descriptor

Image recognition algorithms applied to standard benchmark problems in the machine learning community use small RGB or grayscale images (32×32 pixel for CIFAR [11], 96×96 pixel for NORB [15]) in which objects are centered and images are scaled to equal dimensions. Features are then commonly extracted in a convolutional manner.

In contrast to this the extraction of image descriptors based on local patches around interest points is a successfully used approach in many computer vision tasks. It is especially useful when one wants to process large images (e.g. images of up to 640×480 pixel containing additional depth information in the case of Kinect RGB-D data) as well as images of varying sizes which may contain objects that are partially occluded or appear in cluttered scenes.

It is however not straightforward to modify these approaches to effectively use richer data representations. The idea of the *convolutional k-means descriptor* is to remedy this shortcoming by explicitly learning feature responses around detected interest points which are then used to build a local feature histogram in a small region around interest points.

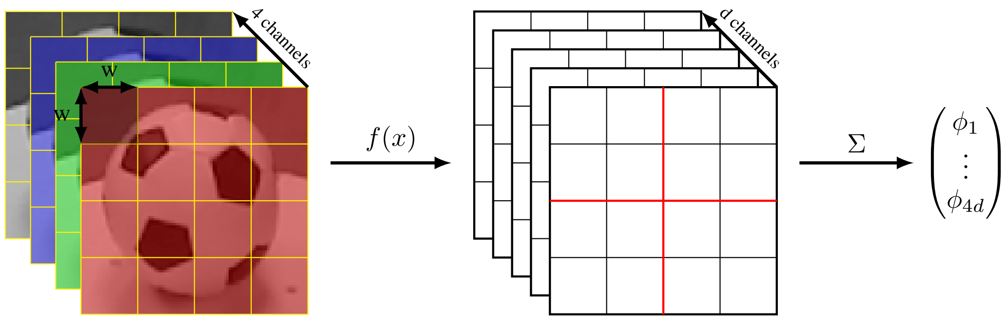


Figure 3: The extraction procedure for the learned features. Feature responses $f(x)$ are computed for each $w \times w$ patch x in the image and then pooled in four quadrants.

The procedure necessary to extract *convolutional k-means descriptors* from a given input image can be divided into three steps.

1. A set of feature responses must be learned capturing the image structure of patches around interest points.
2. An interest point detector is run on the input image.
3. *Convolutional k-means descriptor* responses are extracted around each detected interest point.

To learn the feature responses the approach described in Section 3 is used with the modification that image patches are only extracted in the vicinity of interest points.

For the interest point detection any standard detector such as SIFT [16], SURF [2] and FAST [17] can be used. Alternatively, if computation time is not the limiting factor it is also possible to extract feature descriptors at fixed grid positions.

The extraction of the *convolutional k-means descriptor* around a given set of interest points is a locally pooled variant of the feature extraction step described in 3.4. It is similar to the extraction of SIFT/SURF descriptors. First a $k \times k$ pixel region around the interest point is extracted, we chose $k = 16$. Then the feature response $f(x) \in \mathbb{R}^d$ of each $w \times w$ sub-patch $x \in \mathbb{R}^{w \times w \times 4}$ in the region of interest is computed.

The computation of the feature histogram at pixel position $p = (p_x, p_y)^T$ used as the descriptor can then be formalized as a vector $F(p_x, p_y) \in \mathbb{R}^{4 \cdot d}$ with

$$F_i(p_x, p_y) = \sum_{(p_{x'}, p_{y'})^T \in P} f_i(x_{(p_{x'}, p_{y'})}) \cdot \delta(p_x, p_y, p_{x'}, p_{y'}, i), \tag{3}$$

where P is the set of all pixels within the $k \times k$ region of interest, $x_{(p_{x'}, p_{y'})}$ is the $w \times w$ patch around the pixel $(p_{x'}, p_{y'})$ and δ is an indicator function computing the corresponding descriptor region of the pixel given as

$$\delta(p_x, p_y, p_{x'}, p_{y'}, i) = \begin{cases} 1 \text{ if} & \begin{aligned} &\max(0, \operatorname{sgn}(p_{x'} - p_x)) + \\ &\max(0, \operatorname{sgn}(p_{y'} - p_y)) \cdot 2 \\ &= \lfloor i/d \rfloor \end{aligned} \\ 0 \text{ else} & \end{cases}. \tag{4}$$

Since the number of learned features d is typically high (e.g. 1200) the size of the descriptor vector also tends to be large. If a smaller feature vector is required its dimensionality can be reduced by performing a PCA over all descriptors extracted from the training images, dropping all but the most significant principal components.

5 Evaluation

To evaluate the performance of the learned features, we test the convolutional k-means method (CKM) described in Section 3 and the proposed convolutional k-means descriptor (CKM Desc.) in an object recognition setting on the *RGB-D Object Dataset* [12]. We focus our experiments on the influence of the PCA dimensionality reduction, investigate the contribution of depth information and compare the object recognition performance with several state-of-the-art algorithms.

5.1 RGB-D Object Dataset and experimental setup

The *RGB-D Object Dataset* which is presented in [12] is a novel, large scale RGB-D dataset containing image sequences of 300 objects, grouped in 51 categories. Images are captured with a Kinect style camera at a resolution of 640×480 pixel. Each object is recorded from three viewing heights ($30°$, $45°$ and $60°$ above the horizon) while it rotates on a turntable resulting in approximately 150 views per object and 207920 RGB-D images in total. The images are already cropped and segmented, obviating the need for object detection and segmentation.

For our experiments, we use the same setup as in [12], distinguishing between category and instance recognition. For object recognition on the instance level, the task for the recognition algorithm is to detect the exact physical instance of an object that was previously presented as a training image. The object may appear in a different setting as the training instance, i.e. in a different scene, different lighting conditions and changing perspective. In the object recognition setting on the category level a set of training images with corresponding labels is presented. The labels group object instances into a set of categories (e.g. all instances of chairs are assigned the label *chair*). The objects presented in the recognition phase are new instances of a given category. The task for the algorithm then is to assign the correct label to the presented object.

To generate training and test sets, first, the data is subsampled by taking every fifth video frame resulting in 41942 RGB-D images. For category recognition, from each category one object is selected randomly for testing and the classifier is trained on all remaining objects. For instance recognition there are two scenarios:

- Alternating contiguous frames: Each video is divided into 3 contiguous sequences of equal length. For each object there are three heights, giving a total of 9 video sequences for each instance. These are split randomly in 7 parts for training and 2 parts for testing.
- Leave-sequence-out: The classifier is trained on the video sequences from the $30°$ and $60°$ perspective and evaluated on the $45°$ video sequence.

If not stated otherwise, the following experiments were conducted in the leave-sequence-out scenario. For all experiments a set of features was learned on a representative subset of the training data, feature responses were extracted on all training images and a linear SVM was used as the classifier.

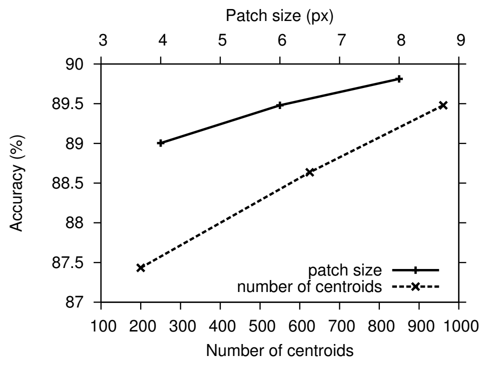
5.2 Contribution of depth information

We show that depth information improves recognition accuracy by applying the convolutional k-means approach on the dataset in two experimetal conditions, one using all available 4 channels (RGB-D) and one using only color information (RGB) from the images. In both experiments, 961 features with a patch size of 6 pixel were learned while keeping the first 60 principal components. Patches were extracted with a stride of 1. Neglecting the depth channel and learning features solely on the RGB images (CKM RGB) we achieved an accuracy of 80.2%. In the second condition, including the depth channel (CKM RGB-D), the accuracy could be increased to 87.3% (see Fig. 4a).

We conducted the same evaluation for the proposed *convolutional k-means descriptor* using the same parameters as above, but extracting descriptors on fixed grid positions with a grid size of 4×4 pixel. Ignoring depth information (CKM Desc. RGB), the recognition accuracy was 82.9%. Using all available channels, accuracy was significantly higher (89.5%) (CKM Desc. RGB-D).

Method	Accuracy (%)
CKM (RGB)	80.2
CKM Desc. (RGB)	82.9
CKM (RGB-D)	87.3
CKM Desc. (RGB-D)	**89.5**

(a)



(b)

Figure 4: (a) Comparison of classification accuracies on the RGB-D Object Dataset in the leave-sequence-out setting (961 features, patch size of 6 pixel, keeping the first 60 principal components). (b) Effect of patch size (using 961 features) and number of centroids (using patch size of 6 pixel) on classification accuracy tested on the *RGB-D Object Dataset*. All experiments were conducted in the leave-sequence-out scenario, keeping the first 60 principal components.

5.3 Effect of patch size and number of centroids

As pointed out in [7], two important parameters influencing the performance of the learned features are the size of the extracted patches as well as the number of centroids that constitute the feature dictionary. They reported that patch sizes of $6 - 8$ pixel performed best and that performance improved with increasing numbers of features. To test whether those findings also hold when the features are used to build the *convolutional k-means descriptor*, we conducted a similar parameter evaluation where we tested patch sizes of 4, 6 and 8 (using 961 features) and learned dictionaries with 400, 625 and 961 features (using a patch size of 6 pixel). In all experiments we kept the first 60 principal components. The results are shown in Figure 4b, indicating that recognition accuracy increases linearly with the number of centroids. It should however be noted that there is only a small difference of 2% between the top performing result (961 features) and the one using 200 features. The same holds for the patch size experiment. Larger patches help to improve results, the variance however is even smaller (0.8%).

5.4 Effect of dimensionality reduction

It remains to evaluate the effect of the changes to the feature learning procedure which were proposed in Section 3.1 in order to improve the feature quality.

We first use a representative subset of the training images to train the features and vary the number of principle components that are kept. The maximum number of components is the original dimensionality of the patches extracted for learning the features, which is $N = w \cdot w \cdot 4$, where w represents the patch size in pixel. For this experiment, we chose the patch size to be $w = 6$ pixel and hence $N = 144$. We therefore conduct the experiment keeping all principal components $c = 144$ as well as keeping $c = 100$ and $c = 60$ components. In all three settings the k-means algorithm for learning the feature dictionaries is run for a maximum of 300 episodes.

Table 1 depicts the results of the experiments for both, the convolutional k-means approach (using 961 features) and the proposed descriptor (using 1200 features). It can be clearly seen that the projection of the input data to the first principle components improved the quality of the resulting feature dictionary as it resulted in a significant improvement of recognition performance. It should be noted that apart from this performance increase a more practical measure, the computation time required for clustering and descriptor extraction is also reduced by the dimensionality reduction.

		Number of principal components kept		
		144	100	60
Accuracy (%)	CKM RGB-D (961 features)	85.0	85.9	**87.3**
	CKM Desc. RGB-D (1200 features)	86.8	87.2	**89.3**

Table 1: Influence of the PCA dimensionality reduction evaluated in the leave-sequence-out setting.

Method	Instance (Leave sequence out)	Instance (Alternating contiguous frames)	Category
Linear SVM [12]	73.9	90.2 ± 0.6	81.9 ± 2.8
Nonlinear SVM [12]	74.8	90.6 ± 0.6	83.8 ± 3.5
Random Forest [12]	73.1	90.5 ± 0.4	79.6 ± 4.0
IDL [13]	-	91.3 ± 0.3	85.4 ± 3.2
HKDES [4]	82.4	-	84.1 ± 2.2
Kernel Desc. [5]	84.5	-	86.2 ± 2.1
CKM Desc. (this work)	**89.3**	$\mathbf{92.1} \pm 0.4$	$\mathbf{86.4} \pm 2.3$

Table 2: Classification accuracy of the proposed descriptor evaluated on the RGB-D Object Dataset using 1200 learned features. Accuracies are averaged over 10 trials.

5.5 Performance on the *RGB-D Object Dataset*

Finally, we conducted another experiment to compare the performance of the learned features on the RGB-D Object Dataset [12] to several state-of-the-art algorithms in all three settings leave-sequence-out, alternating contiguous frames and category recognition.

For this purpose we use the best performing configuration from the previous experiments. Hence the descriptor extraction scheme on a fixed grid is used and 1200 features are learned using the first 60 principal components. Again, we choose $w = 6$ and run the k-means algorithm until convergence.

As can be seen in Table 2, using the learned descriptor we are able to improve on the best currently known approaches. In the leave-sequence-out setting the previous best result is achieved using Kernel Descriptors [5] which reach an accuracy of 84.5%. Using the proposed *convolutional k-means descriptor* this result can be improved by a large margin of 4.8%. In the alternating contiguous frame and the category recognition setting our approach can also slightly improve on previous results reaching 92.1% and 86.4% respectively.

6 Conclusion

In this work we have shown the applicability of an unsupervised feature learning approach for the task of object recognition in RGB-D data. In contrast to traditional hand-designed features, our approach is able to automatically combine different channels of information into one concise representation.

Its performance was evaluated on the *RGB-D Object Dataset*, where it was able to improve on previously published results. Especially in the instance recognition problem, the best performing algorithms to date were outperformed by a large margin. To the best of our knowledge our descriptor therefore has the highest accuracy on this dataset in all three recognition settings. We conclude that a learned feature descriptor is a valuable tool for efficiently combining different sensory modalities.

Future work will focus on learning rotation and scale invariant descriptors which will help to improve the object recognition performance in real world applications.

References

[1] Alaa E. Abdel-Hakim and Aly A. Farag. CSIFT: A SIFT Descriptor with Color Invariant Characteristics. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[2] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Proceedings of the 9th European Conference on Computer Vision (ECCV)*, 2006.

[3] Anthony J. Bell and Terrence J. Sejnowski. The ”independent components” of natural scenes are edge filters. *Vision research*, 37(23):3327–3338, December 1997.

[4] Liefeng Bo, Kevin Lai, Xiaofeng Ren, and Dieter Fox. Object Recognition with Hierarchical Kernel Descriptors. In *IEEE Computer Vision and Pattern Recognition (CVPR)*, 2011.

[5] Liefeng Bo, Xiaofeng Ren, and Dieter Fox. Depth Kernel Descriptors for Object Recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011.

[6] Dan C. Ciresan, Ueli Meier, Jonathan Masci, Luca M. Gambardella, and Jürgen Schmidhuber. High-Performance Neural Networks for Visual Object Classification. Technical report, Dalle Molle Institute for Artificial Intelligence, Manno, Switzerland, 2011.

[7] Adam Coates, Honglak Lee, and Andrew Y. Ng. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[8] Adam Coates, Andrew Y. Ng, and Serra Mall. The Importance of Encoding Versus Training with Sparse Coding and Vector Quantization. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2011.

[9] Geoffrey E Hinton. Learning multiple layers of representation. *Trends in cognitive sciences*, 11:428–34, October 2007.

[10] Aapo Hyvärinen and Erkki Oja. Independent component analysis: algorithms and applications. *Neural networks*, 13(4-5):411–30, 2000.

[11] Alex Krizhevsky. *Learning Multiple Layers of Features from Tiny Images*. Master thesis, Department of Computer Science, University of Toronto, 2009.

[12] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. A Large-Scale Hierarchical Multi-View RGB-D Object Dataset. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[13] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. Sparse Distance Learning for Object Recognition Combining RGB and Depth Information. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[14] Quoc V Le, Jiquan Ngiam, Zhenghao Chen, Daniel Chia, Pang Wei Koh, and Andrew Y. Ng. Tiled convolutional neural networks. *Advances in Neural Information Processing Systems*, 2010.

[15] Yann LeCun, Fu Jie Huang, and Bottou Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 97–104, Washington, DC, USA, 2004. IEEE.

[16] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, November 2004.

[17] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*, 2006.

[18] Bastian Steder, R.B. Rusu, Kurt Konolige, and W. Burgard. Point Feature Extraction on 3D Range Scans Taking into Account Object Boundaries. In *Proc. of IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[19] Kai Yu and T. Zhang. Improved local coordinate coding using local tangents. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2010.