Information Sciences 320 (2015) 333–345

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Recognizing multi-view objects with occlusions using a deep architecture

Yingjie Xia [a,b], Luming Zhang [c,*], Weiwei Xu [b], Zhenyu Shan [b], Yuncai Liu [b]

[a] College of Computer Science, Zhejiang University, 310027 Zhejiang, China
[b] Hangzhou Institute of Service Engineering, Hangzhou Normal University, 310012 Zhejiang, China
[c] School of Computer and Information, Hefei University of Technology, China

ARTICLE INFO

Article history:
Received 15 June 2014
Received in revised form 18 January 2015
Accepted 30 January 2015
Available online 11 February 2015

Keywords:
Deep learning
Depth data
Multi-view
Object recognition
RGB-D

ABSTRACT

Image-based object recognition is employed widely in many computer vision applications such as image semantic annotation and object location. However, traditional object recognition algorithms based on the 2D features of RGB data have difficulty when objects overlap and image occlusion occurs. At present, RGB-D cameras are being used more widely and the RGB-D depth data can provide auxiliary information to address these challenges. In this study, we propose a deep learning approach for the efficient recognition of 3D objects with occlusion. First, this approach constructs a multi-view shape model based on 3D objects by using an encode–decode deep learning network to represent the features. Next, 3D object recognition in indoor scenes is performed using random forests. The application of deep learning to RGB-D data is beneficial for recovering missing information due to image occlusion. Our experimental results demonstrate that this approach can significantly improve the efficiency of feature representation and the performance of object recognition with occlusion.

1. Introduction

In computer vision applications, object recognition based on multi-view images is essential for extracting semantic information from image pixels. In general, the approaches employed can be categorized into region- and feature point-based recognition methods. In this study, we focus on RGB-D images of indoor scenes where the key task is the recognition of various objects such as desks and chairs. The semantic understanding of RGB-D images can facilitate the 3D modeling of indoor scenes for applications in computer graphics and robotics.

Using RGB-D images, we can extract the color and depth information for an object. Recently, depth information was applied to indoor scene object recognition in [29] using random forests regression. However, this method was only validated with a small-scale database and it was not robust to occlusions in the captured image. Occlusion is one of the major factors that affect the accuracy of object recognition. First, occlusions are difficult to predict during the training stage. Occlusions are determined by the spatial relationships among objects from certain viewpoints, but it is impractical to include all possible occlusions during the training phase of recognition algorithms. In addition, occlusion can significantly decrease the accuracy of the extracted feature vectors because some regions will have missing information.

* Corresponding author.
E-mail address: zglumg@gmail.com (L. Zhang).

http://dx.doi.org/10.1016/j.ins.2015.01.038
0020-0255/

In this study, we propose the construction of multi-view shape models of 3D objects with occlusion using an encode–decode deep learning network. Based on these shape models, we can reconstruct the information embedded in an occluded region for use in the subsequent feature representation process. This approach is based on observations that encode–decode deep learning networks are efficient in recovering missing information from partially observed data. Next, we apply random forests to the models for object recognition.

We tested our pipeline using various occluded indoor scene objects with 10–20% occlusion in the object area. Our experimental results demonstrate that preprocessing input depth data using a deep learning network can improve the recognition performance.

The remainder of this paper is organized as follows. In Section 2, we review related research into object recognition and deep learning. Section 3 introduces our approach to 3D object depth data generation, including the methods for training set generation and test set generation. Section 4 describes feature representation using deep learning and in Section 5, we explain how these features are used for 3D object recognition. In Section 6, we present the results of experiments that demonstrate the accuracy of object recognition with and without occlusions. Finally, we give our conclusions and suggestions for future research in Section 7.

2. Related work

2.1. Object recognition

The general process employed for object recognition can be categorized into five phases: verification, detection and localization, classification, naming, and description [18]. Many previous studies have attempted to address the object recognition problem.

Zerroug and Nevatia [40] proposed a generalized cylinder-based recognition method that uses a subset of generalized cylinders to detect the target object [3]. Brooks [4] proposed a parts-based recognition system for object recognition.

Peng et al. [25] proposed a descriptor that utilizes sectors and contour edges to represent local image features. Pham [26] proposed a recognition algorithm based on template matching strategy, which greatly improved the recognition accuracy in chickens. Lu et al. [21] developed a 3D-model based retrieval and recognition method, which uses a semi-supervised learning method to train a classifier for object recognition.

Around 1990, geometric invariants were introduced to build an efficient indexing mechanism for use in recognition. The geometric invariants-based method was first proposed by Schwartz and Sharir [28], where the basic idea is to obtain a coordinates frame using feature points before utilizing the coordinate frame in the expression of an affine invariant. To integrate this approach with 3D modeling, Flynn and Jain [7] proposed the application of invariant feature indexing in 3D object matching.

Subsequently, researchers showed that it is beneficial to employ local features extracted from images for representation during object recognition. An example is the iconic representation algorithm [27], which extracts local feature vectors to represent the multi-scale local orientation of image locations. The best-known method is SIFT [37], which detects the points of interest in an image. Mikolajczyk and Schmid [22] proposed an affine-invariant interest point detector to improve the reliability of recognition using feature grouping modules. Later, Csurka et al. [5] and Sivic and Zisserman [31] introduced the bag-of-features approach for recognition, which obtains satisfactory results with viewpoint changes and background clutter.

2.2. Deep learning

In machine learning, a single architecture such as kernel machines can only work with a fixed feature layer. At present, deep learning is becoming more prevalent, which is a complex architecture with multiple layers that contain nonlinear components with many trainable parameters, [2]. Convolutional neural networks (CNNs) [16] and deep belief networks (DBNs) [12] are two approaches in this area.

In early trials, CNNs proved to be the most successful in applying multi-layer neural networks. These methods use local connections and shared weights to combine multiple neural units [16]. By reducing the number of training parameters, CNNs can accelerate the learning process during general feed-forward back-propagation training [1].

DBN-based methods were proposed in 2006. DBNs comprise an undirected graph model and a directed graph model, where the former is an associative memory that contains top level units and one level of hidden units, and the latter is a stacked restricted Boltzmann machine (RBM) that contains the remaining layers.

There are many variations of deep learning methods, which are similar to CNNs and DBNs. For example, Ji et al. [14] proposed 3-D CNNs, which use temporal features during network modeling. A convolutional DBN was also proposed by [17].

Deep learning is a generic machine learning tool, which can be applied in many areas. For example, during visual document analysis, MNIST can recognize images using CNNs [30]. In face recognition, a deep learning network can learn high-level facial features to narrow the semantic gap between the low level features. Karnowski et al. [15] utilized a deep spatio-temporal inference network during image classification. In general, 3D CNNs are one of the best performing methods and deep learning-based methods are quite useful for solving recognition problems.

3. Multi-view 3D object depth data generation

3.1. Training set generation

By sampling the models in a database from different views and distances, each model can generate several depth images, which can be used as the training set by the proposed method [33]. In order to consider the possible variations in the distance between the camera and objects when taking photographs, we set the translation between 1.3 and 2.3, which allowed 16 depth images of $640 * 480$ pixels to be generated for each 3D model. As shown in Fig. 1, depth images of various models from the database were sampled from different views and distances.

Bilinear interpolation is an extension of linear interpolation for two variables on a 2D grid [20]. The key idea of this algorithm is to perform linear interpolation first in one direction and then in the other direction. Although these two steps are linear, the overall interpolation is nonlinear and quadratic. This algorithm can be utilized to amplify or narrow an image [10]. Thus, in order to boost the training speed, we compressed the image into $128 * 96$ pixels using the bilinear interpolation algorithm according to the following equations:

$$f(x,y_1) = \frac{x_2 - x}{x_2 - x_1} f(x_1,y_1) + \frac{x - x_1}{x_2 - x_1} f(x_2,y_1), \tag{1}$$

$$f(x,y_2) = \frac{x_2 - x}{x_2 - x_1} f(x_1,y_2) + \frac{x - x_1}{x_2 - x_1} f(x_2,y_2), \tag{2}$$

$$f(x,y) = \frac{y_2 - y}{y_2 - y_1} f(x,y_1) + \frac{y - y_1}{y_2 - y_1} f(x,y_2), \tag{3}$$

where (1) and (2) denotes interpolation in the x direction and (3) denotes the interpolation in the y direction. Finally, each compressed depth image is divided into $32 * 32$ patches, with four pixels in each patch. In addition, patches that lack depth information are removed from the training set when training the random regression forests [36]. In Fig. 2, the depth image of a chair is divided into several continuous patches.

3.2. Test set generation

Initially, the depth images of indoor scenes captured by the RGB-D camera were also resized to $128 * 96$ pixels as the training set. We obtained multiple depth images for each object from different angles to obtain better characterizations of the objects, as shown in Fig. 3.

We obtained several test patches by sampling patches of $32 * 32$ with four pixels per patch from the depth images of a chair.

As shown in Fig. 4, the depth image was divided into several continuous patches. When testing, patches without depth data were not passed to the random forests algorithm, thereby reducing the number of false positives.

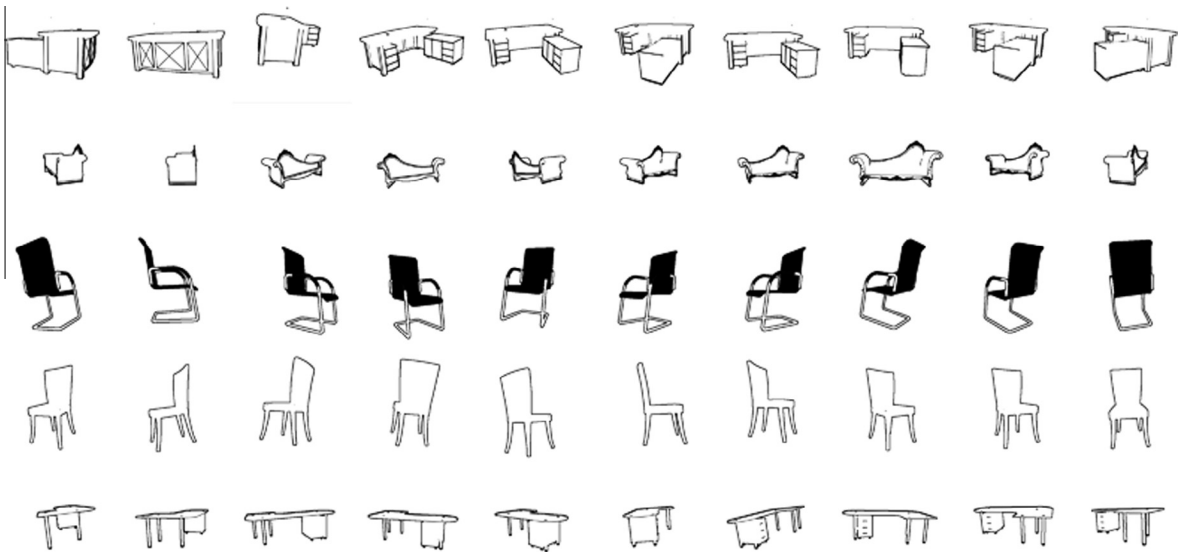


Fig. 1. Depth images sampled from desks, chairs, and sofas in a database.

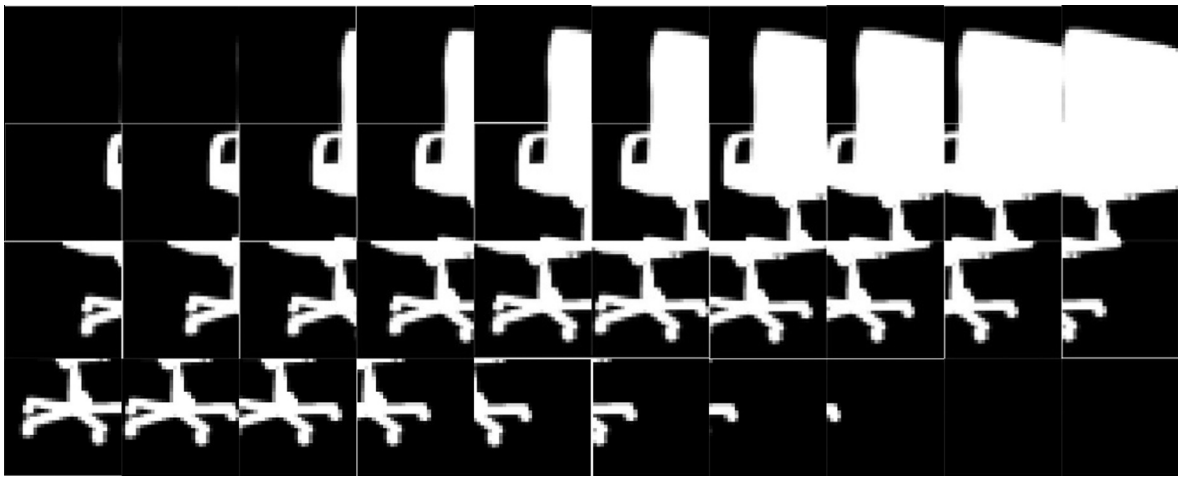


Fig. 2. Continuous sampling of $32 * 32$ patches from a chair.

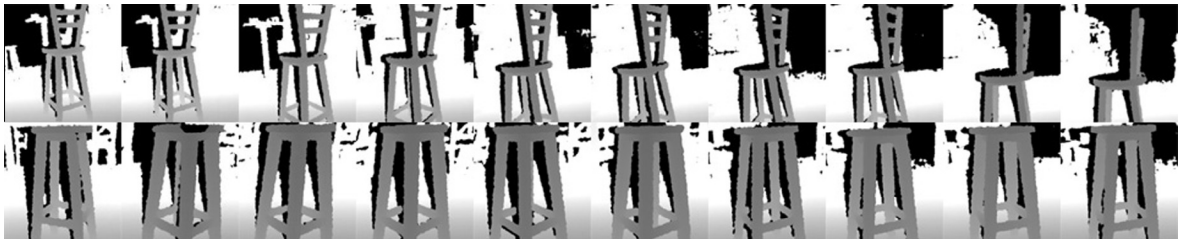


Fig. 3. Depth images of two chairs captured by an RGB-D camera from different angles.

Fig. 4. The $32 * 32$ patches sampled from the depth images of a chair.

In addition, the depth model obtained might fail because the depth data employed different units in the training set and the test set. Thus, data normalization was applied to both the training set and the test set [34,8]. Moreover, the angles of the training set and the test set were controlled to within plus or minus 30 degrees in the horizontal direction as the normal human viewpoint. The distance of the test data was controlled within the range where the RGB-D cameras could capture sufficient depth data from the target. The data obtained might not have been suitable for the algorithm if the bound was outside this range.

4. Feature representation using deep learning

The most important aspect of classification comprises the useful features that are learned. The core of deep learning is the construction of a learning model with several hidden layers to learn more important features related to the targets [6].

Because 3D data are much more complex than 2D data if occlusions are present, deep learning can be an effective tool for learning the features of 3D objects.

4.1. RBMs

An RBM is a generative stochastic neural network, which can learn a probability distribution based on its set of inputs. RBMs have been applied to dimensionality reduction [11], classification [3], collaborative filtering, feature learning [4], and topic modeling [40]. RBMs are a variant of Boltzmann machines, but with the restriction that their neurons must form a bipartite graph. Their input units correspond to the features of their inputs and hidden units are trained, where each connection in an RBM must connect a visible unit to a hidden unit. This restriction enables the production of more efficient training algorithms in the general class of Boltzmann machines, particularly the gradient-based contrastive divergence algorithm [25]. A graphical depiction of an RBM is shown in Fig. 5.

The standard type of RBM has binary-valued hidden and visible units. It comprises weights $W = (w_{i,j})$ associated with the connections between a hidden unit (h_j) and a visible unit (v_j), as well as bias weights a_i for v_i and b_j for h_j. The energy function is defined as

$$E(v,h) = \sum_i a_i v_i - \sum_j b_i h_j - \sum_i \sum_j h_j w_{i,j} v_i \tag{4}$$

or in vector form as,

$$E(v,h) = -a^T V - b^T H - H^T W V. \tag{5}$$

RBMs lack input–input and hidden–hidden interactions, so the energy function $E(v,h)$, which is analogous to that of a Hopfield network, is bilinear [13]. As in general Boltzmann machines, the probability distributions over hidden and/or visible vectors are defined in terms of the energy function [21]:

$$P(v,h) = \frac{1}{Z} e^{-E(v,h)}, \tag{6}$$

where Z is a partition function that is defined as the sum of $e^{-E(v,h)}$ over all possible configurations. Similarly, the probability of a visible vector of Booleans is the sum over all possible hidden layer configurations [21]:

$$P(v,h) = \frac{1}{Z} \sum_h e^{-E(v,h)}. \tag{7}$$

Since the RBM has the shape of a bipartite graph, with no intra-layer connections, the hidden unit activations are mutually independent of the given visible unit activations, whereas the visible unit activations are mutually independent of the given hidden unit activations [25]. Thus, for m visible units and n hidden units, the conditional probability of the visible units v given the hidden units h is

$$P(v|h) = \prod_{i=1}^{m} P(v_i|h). \tag{8}$$

By contrast, the conditional probability of h given v is

$$P(h|v) = \prod_{j=1}^{n} P(h_i|v). \tag{9}$$

The individual activation probabilities are given by $P(h_j = 1|v) = \sigma(b_j + \sum_{i=1}^{m} w_{i,j} v_i)$ and $P(v_i = 1|h) = \sigma\left(a_i + \sum_{j=1}^{n} w_{i,j} h_i\right)$, where σ denotes the logistic sigmoid.

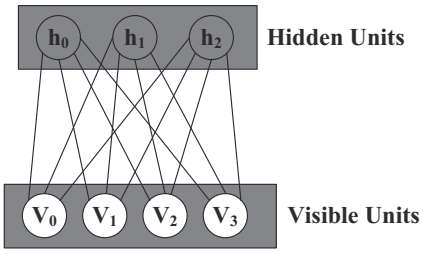


Fig. 5. Graphical depiction of an RBM.

The aim when training RBMs is to maximize the product of the probabilities assigned to some training set V, $\operatorname{argmax}_w \prod_{v \in V} P(v)$. This is equivalent to maximizing the expected log probability of V: $\operatorname{argmax}_w E\left[\sum_{v \in V} \log P(v)\right]$. The algorithm that is used most often to train RBMs is the contrastive divergence algorithm, which was originally developed to train product of experts models [24]. This algorithm employs Gibbs sampling and it is used inside a gradient descent procedure to compute the weight updates. The basic contrastive divergence procedure for a single sample is as follows.

Step 1. Take a training sample v, compute the probabilities of the hidden units, and sample a hidden activation vector h from the probability distribution.
Step 2. Compute the outer product of v and h, and denote this as the positive gradient.
Step 3. From h, sample a reconstruction v' of the visible units, then resample the hidden activations h'.
Step 4. Compute the outer product of v' and h', and denote this as the negative gradient.
Step 5. Let the weight update to $w_{i,j}$ be the positive gradient minus the negative gradient times some learning rate: $\Delta w_{i,j} = \varepsilon\left(vh^T - v'h'^T\right)$.

The updating rule is similar for the biases a_i and b_j.

4.2. DBNs

In machine learning, a DBN is a generative graphical model, or a type of deep neural network, which comprises multiple layers of hidden units that have connections between the layers but not between the units within each layer. After it has been trained using a set of examples in an unsupervised manner, a DBN can learn how to probabilistically reconstruct its inputs. The layers then act as feature detectors on inputs. After this learning step, a DBN can be trained further in a supervised manner to perform classification [9]. The RBM can be used as the building block in a DBN, as shown in Fig. 6, because it shares parameterizations with the individual layers of a DBN.

The principle of greedy layer-wise unsupervised training can be applied to DBNs using RBMs as the building blocks for each layer, as follows. (1) Train the first layer as an RBM that models the raw input $x = h^{(0)}$ as its visible layer. (2) Use the first layer to obtain a representation of the input that will be used as data by the second layer. This representation is selected as the mean activations $P\left(h^{(1)} = 1 | h^{(0)}\right)$. (3) Train the second layer as an RBM by taking the transformed samples

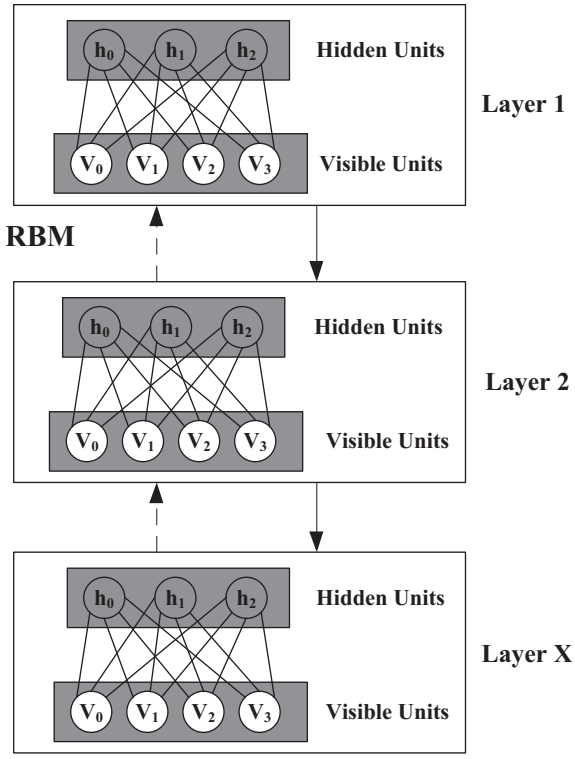


Fig. 6. Graphical depiction of a DBN.

as training examples for the visible layer of the RBM. (4) Iterate (2) and (3) for the desired number of layers by propagating the samples or mean values upward each time. (5) Fine tune all of the parameters of this deep architecture with respect to a proxy for the DBN log-likelihood.

4.3. Data reconstruction

Data reconstruction includes two processes: encoding and decoding [38]. The single layer of an RBM is not the best way to model the structure. Thus, a DBN with RBMs as blocks is used for data reconstruction. An example of the overall process is shown in Fig. 7. The encoding is a process that involves DBN training. After encoding, each layer of features captures the strong, high-order correlations between the activities of the units in the layer below. These data are reconstructed better after fine tuning [32]. For a wide variety of datasets, this is an efficient way for progressively identifying low-dimensional and nonlinear structures. Thus, it is beneficial for data reconstruction when the object is blocked.

RBMs might not be efficient for representing some distributions that could be represented compactly with an unrestricted Boltzmann machine, but RBMs can represent any discrete distribution provided that sufficient hidden units are used. In addition, it can be shown that unless the RBM already models the training distribution perfectly, adding a hidden unit will always improve the log-likelihood. It is not known how many hidden units will be sufficient to represent the data perfectly. After learning one layer of feature detectors, we can treat their activities as data to learn a second layer of features. The first layer of the feature detectors then becomes the visible units used to learn the next RBM. This layer-by-layer learning approach can be repeated as many times as necessary [39]. It can be proved that adding an extra layer always improves the lower bound based on the log probability that the model assigns to the training data, provided that the number of feature detectors per layer does not decrease and their weights are initialized correctly. Thus, the second problem is how many layers are required to obtain the best performance when representing the features. These two problems are discussed in the experimental section.

5. Object recognition

The features of objects are learnt by the DBN and then used as inputs by the random forests to classify the objects according to labels that represent various object classes, which can be formulated as $L = g(f(i), V)$, where g refers to the random forests classifier, $f(i)$ is the feature vector learnt by DBN, and V is the vector of the parameter configurations for the nodes used to construct the decision trees in the random forests. Compared with other classification algorithms, training and testing are rapid with random forests, and this method outperforms others when handling extremely large volumes of training data. The random forests method can balance the error during the classification of unbalanced datasets. Furthermore, this method always obtains accurate classifications with high efficiency.

In the training phase, the random forests classifier is trained with the features learned from the training set. The random split selection strategy is used during the training of the random forests classifier [41]. During random selection, simple decision stumps are tested on a feature channel, which is selected randomly when splitting non-leaf nodes [35]. In addition, a large number of thresholds are generated randomly and then tested on the feature channel F_c. After testing the conditional formulae $F_c > \tau$, the qualified patches comprise the patch set of its right child and the remainder are grouped into the left child. The node is split into its left and right child nodes based on the threshold τ, which is a final parameter that maximizes the information gain (IG). IG can be calculated by Eqs. (10) and (11):

$$\mathrm{IG} = H(C) - \sum_{i\in\{L,R\}} \omega_i H_i(C), \tag{10}$$

$$H(C) = \sum_{c} -p(c)\ln p(c), \tag{11}$$

where $H(C)$ refers to the entropy and $p(c)$ is the probability of the object class label, which is calculated as the percentage of class c in the number of patches in the node. The leaf node is created when the number of patches is below 20 or the tree reaches the maximum depth [19].

To evaluate the classifier, the features of an object are learnt by the DBN and fed into the random forests, after which the results from the random trees will be integrated according to Eq. (12):

$$p(c|P_i) = \frac{1}{K}\sum_{l=1}^{K} p_l\left(c|\hat{P}_i\right), \tag{12}$$

where K is the number of trees generated in the forest and $p_l\left(c|\hat{P}_i\right)$ refers to the probability of class c, i.e., the percentage of patches with label c in the leaf node of tree l.

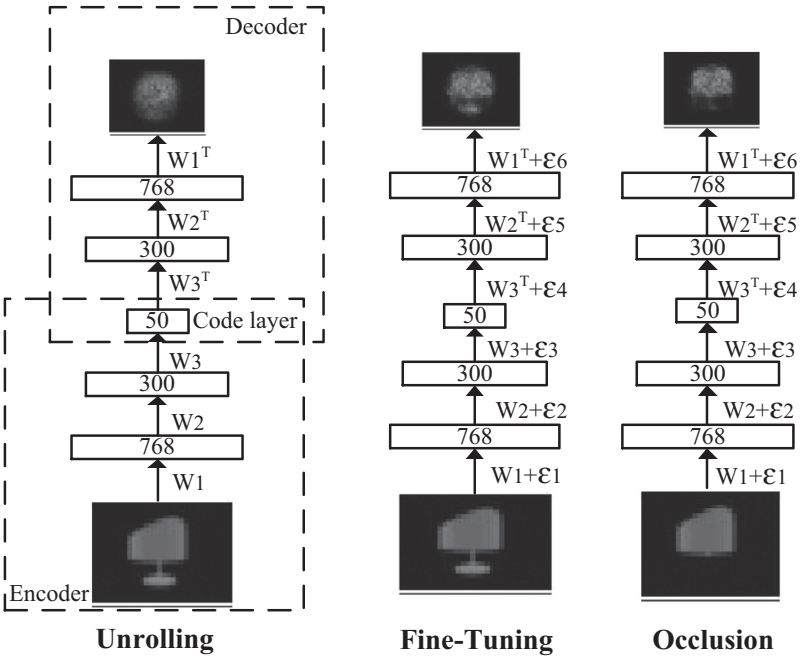


Fig. 7. Example showing the data reconstruction process.

6. Results and analysis

6.1. Experimental setup

In the experiments, we evaluated the accuracy of 3D object recognition using different numbers of DBN layers and different numbers of units for feature training. We also included objects without occlusions and with occlusion areas of different sizes based on different sized patches and intervals to evaluate the recognition accuracy. By default, we set the size of the patch to $32 * 32$ and the interval size as 10. The training set was prepared by collecting $16 * 722$ models, which were captured from 16 views and categorized according to five categories: chairs, tables, cabinets, sofas, and tea tables [23]. The test set was prepared by capturing 10 objects in the five categories from six views. Therefore, the test set contained 60 depth images. We compared our approach with support vector machine (SVM) and random forests algorithms using 2D and 3D features, including histogram of gradients, normal structure tensor, geometry moments, and spin image, where the latter three types of features are extracted from depth data.

The hardware used in the evaluation comprised a desktop PC with a 2.6 GHz Dual core Intel i5 CPU and 4 GB memory. The RGB-D images were captured by a Microsoft Kinect camera at a resolution of $640 * 480$. The background was removed from those images using an adaptive Gaussian-mixture model. We use the Labelme program to label the training set. All of the experiments used indoor scenes.

6.2. Results and analysis

6.2.1. Recognition accuracy with different number of DBN layers and training circulation times

In this experiment, we tested the recognition accuracy of 3D objects using two, three, or four DBN layers and 500, 1000, 2000, or 3000 training circulation times. There were 600 or 200 units in each layer with two layers; 600, 200, or 100 with three layers; and 600, 300, 200, or 100 with four layers. The accuracy of the proposed algorithm was defined as the percentage of correctly labeled depth images, which was defined by the following equation:

$$Accuracy = \frac{correct\_Image}{all\_Image} * 100\%, \tag{13}$$

where *correct_Image* denotes the number of depth images where the class calculated by the proposed classification algorithm was correct compared with the ground truth, and *all_Image* denotes the number of all depth images obtained from various 3D objects. The results are shown in Fig. 8.

The data that we used were not obtained from a standard dataset, so we had to manually assign each test object with the label of the most similar training object. The experimental results showed that the recognition accuracy increased with the

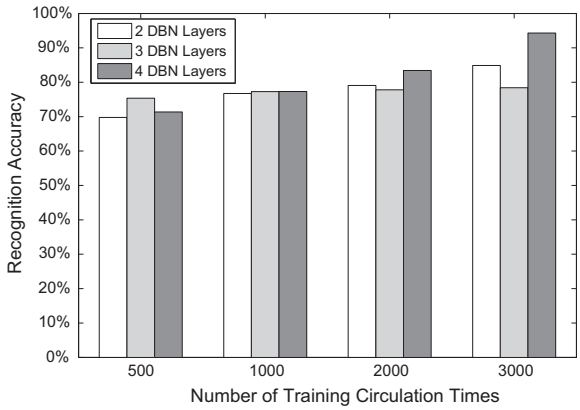


Fig. 8. Recognition accuracy with different numbers of DBN layers and training circulation times.

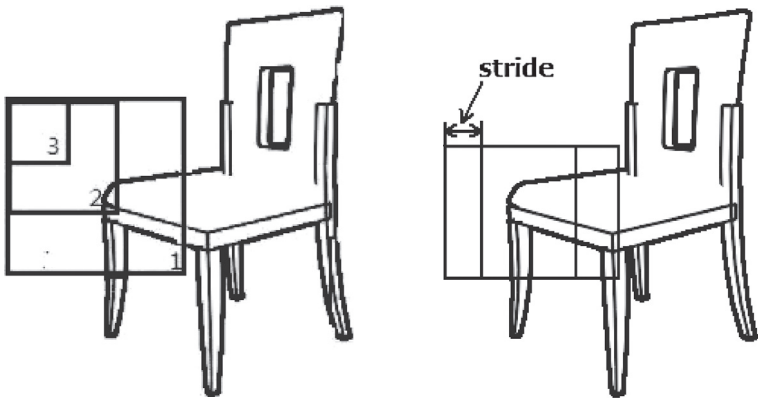
number of training circulation times. This was because a higher number of training circulations could improve the convergence of the average reconstruction error, the mini-batch mean squared error based on the training set, and the full-batch training error. However, increasing the number of DBN layers did not increase the recognition accuracy. The design of DBN is application-oriented, which means that the number of DBN layers and the number of units in each layer should match the specific requirements of applications. According to the results of this test, the 3D object recognition accuracy was highest with four DBN layers and 3000 training circulation times.

6.2.2. Recognition accuracy for 3D objects without occlusions

In this experiment, we evaluated the recognition accuracy for 3D objects without occlusions. This evaluation used different patch sizes that ranged from $16 * 16$ to $48 * 48$, and different interval sizes, i.e., from 10 to 30, as shown in Fig. 9. In Fig. 9(a), the squares of different sizes indicate different patch sizes, while in Fig. 9(b), the distance between two neighboring squares indicates the interval.

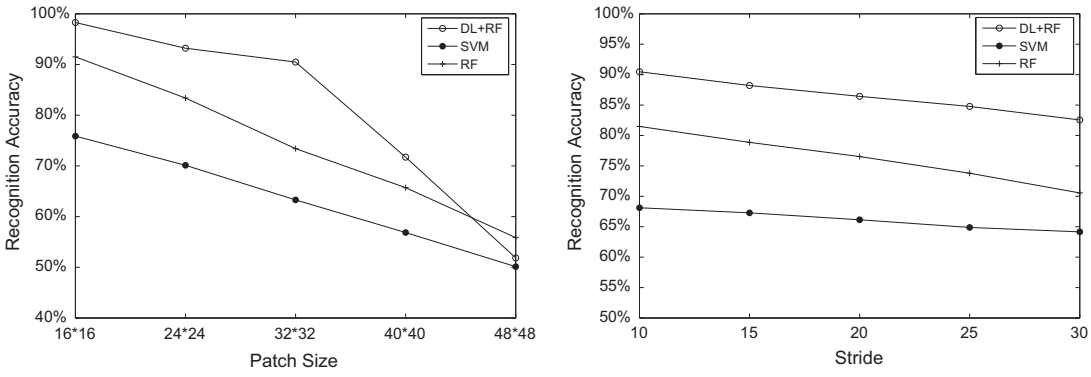
Fig. 10 shows the experimental results obtained employing our approach (deep learning plus random forests) based on the recognition accuracy for 3D objects using different patch sizes with a fixed interval size of 10, as well as different interval sizes with a fixed patch size of $32 * 32$, which are compared with the results obtained using the SVM and random forests.

The results show that our approach always performed better than SVM and random forests, regardless of whether we fixed the interval size and changed the patch size, or if we fixed the patch size and changed the interval size. This is because deep learning can extract features that facilitate the recognition of 3D objects. Fig. 10(a) shows that the recognition accuracy decreased significantly as the patch size increased, which was due to the reduction in the training set size and the smaller number of patches in the test set because the size of the depth images was fixed. Fig. 10(b) shows that the recognition accuracy also decreased as the interval size increased, which is also explained in the same manner as the changes shown in Fig. 10(a). However, if the patch size or interval size was excessively small, such as pixel-level patches, the training set was too large and the training efficiency degraded greatly.



(a) patches with different sizes (b) stride

Fig. 9. Patches and intervals in depth images.



(a) Stride size = 10 (b) Patch size = 32*32

Fig. 10. Recognition accuracy using different patch and interval sizes.

6.2.3. Recognition accuracy of 3D objects with occlusions

In the depth images, the 3D objects were not always clear due to occlusions in the images. Therefore, we tested the recognition accuracy of 3D objects using occlusion areas of different sizes and different number of occlusions. The size of the occlusion areas varied from $4 * 4$ to $32 * 32$, and the number of occlusions ranges from 1 to 5. Some test cases are shown in Fig. 11.

In the first test, the image included only one occlusion but its size varied, as shown in Fig. 12. The recognition accuracy using our approach was compared with that obtained using the SVM and random forests, where the patch size and interval size were fixed at $32 * 32$ and 10, respectively. The results obtained are shown in Fig. 13.

The recognition accuracy of our approach was always higher than that of the other two algorithms, thereby demonstrating that the encode–decode deep learning network could efficiently recover missing information from occlusion areas based on partially observed data. Our approach remained stable when the occlusion size was less than $20 * 20$, and thus $20 * 20$ was set as the size threshold for occlusion areas during 3D object recognition. The performance of random forests was much lower than that of the SVM because occlusions could easily affect the voting mechanism based on all the patches in each image when using the random forests algorithm.

In the second test, we used images that included multiple occlusions but where the size was fixed at $8 * 8$. These occlusions were located in random positions. The recognition accuracy of our approach based on different numbers of occlusions is shown in Fig. 14.

The recognition accuracy of our approach was higher than that of the other two algorithms for the same reason as the trends illustrated in Fig. 13. When the number of occlusions was less than four, the recognition accuracy using the SVM was less than that when using random forests, whereas the opposite was the case when the number exceeded four. This is because the SVM algorithm uses both global and patch level features, which are not affected greatly as the number of occlusions increases, whereas the random forests algorithm only uses patch level features. When the number of occlusions increased to five, the recognition accuracy with our approach was still over 70%, whereas the recognition accuracy using the SVM and random forests was below 60%.

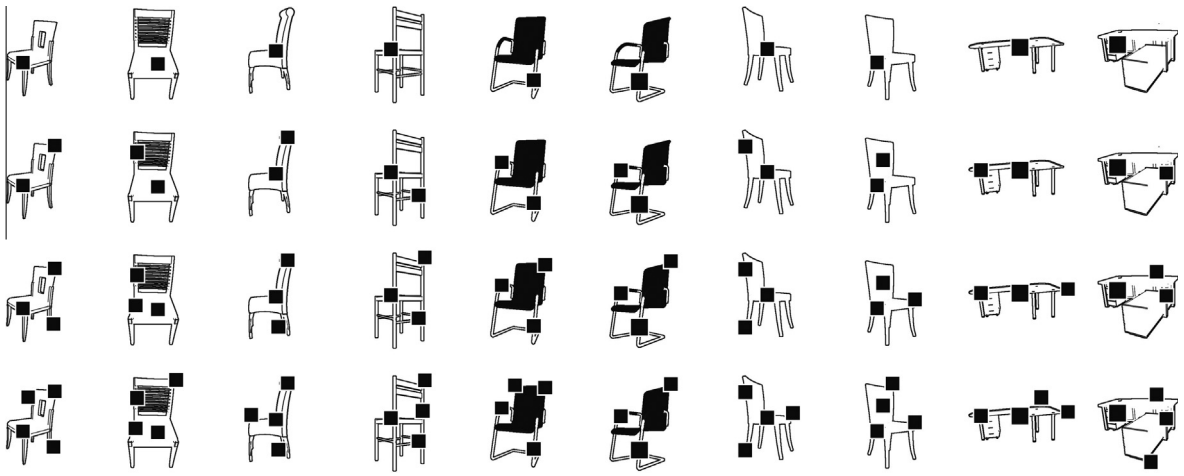


Fig. 11. Test cases of 3D objects with different numbers of occlusions.

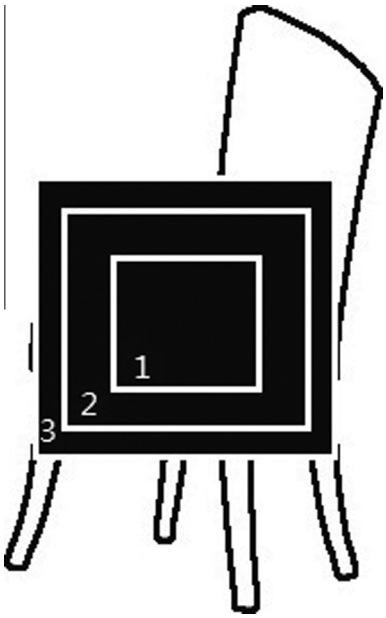


Fig. 12. Depth image with only one occlusion but variable size.

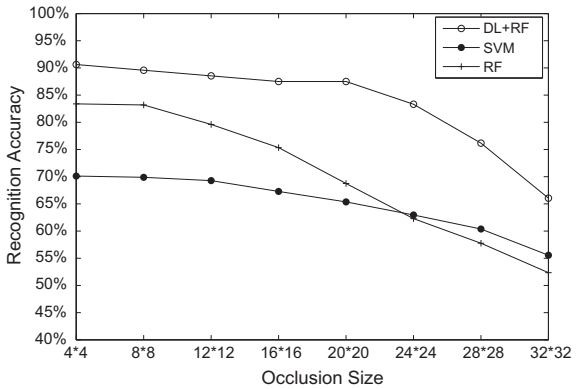


Fig. 13. Recognition accuracy for 3D objects with only one occlusion.

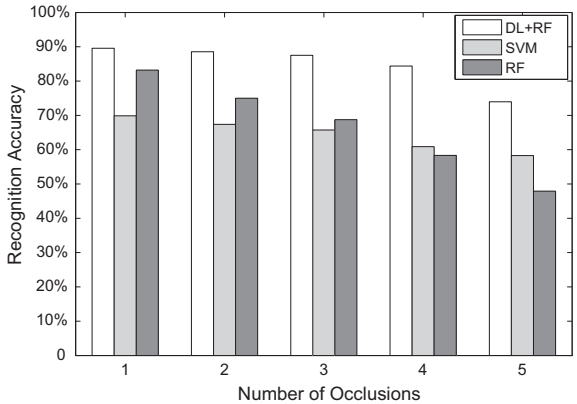


Fig. 14. Recognition accuracy of 3D objects with multiple occlusions.

7. Conclusions and future work

In this study, we developed a multi-view-based 3D object recognition approach for objects with occlusions using deep learning. In our proposed method, we integrate the 3D structures and sample depth images of objects from multiple views, before segmenting the depth images into patches to generate the training set used for deep learning. Next, a DBN with blocks of RBMs is used to train the features of 3D objects. These features are fed into a random forests algorithm to obtain the object classification for recognition. Our results showed that the proposed approach allowed the robust and accurate recognition of 3D objects even when the depth images of objects included occlusion.

In our future research, we plan to include more discriminative depth features by increasing the number of layers in the DBN. We will also investigate the use of pixel-level object classification instead of patch-level object classification to facilitate even more accurate object recognition. Finally, Microsoft Kinect can only capture short-range depth information, so we will test other types of long-range RGB-D cameras.

Acknowledgments

This research was jointly funded by the following funds: National Natural Science Foundation of China under Grant Nos. 61472113 and 61304188, Zhejiang Provincial Natural Science Foundation of China under Grant Nos. LZ13F020004 and LR14F020003, and Zhejiang Xinmiao talent projects under Grant No. 2014R421061.

References

[1] I. Arel, D. Rose, T. Karnowski, Deep machine learning—a new frontier in artificial intelligence research [research frontier], Comput. Intell. Mag. 5 (2010) 13–18.
[2] Y. Bengio, Y. LeCun, Scaling learning algorithms towards ai, Large-Scale Kernel Mach. 34 (2007) 1–41.
[3] T. Binford, Visual perception by computer, in: Proceedings of IEEE Conference on Systems and Control, 1971, p. 262.
[4] R. Brooks, Symbolic reasoning among 3-d models and 2-d images, Artif. Intell. 17 (1981) 285–348.
[5] G. Csurka, C. Dance, L. Fan, J. Willamowski, C. Bray, Visual categorization with bags of keypoints, in: Workshop on Statistical Learning in Computer Vision, ECCV, 2004, pp. 1–2.
[6] S. Fidler, S. Dickinson, R. Urtasun, 3d object detection and viewpoint estimation with a deformable 3d cuboid model, in: Advances in Neural Information Processing Systems, 2012, pp. 611–619.
[7] P. Flynn, A. Jain, 3d object recognition using invariant feature indexing of interpretation tables, CVGIP: Image Underst. 55 (1992) 119–129.
[8] Y. Gao, M. Wang, R. Ji, X. Wu, Q. Dai, 3d object retrieval with hausdorff distance learning, IEEE Trans. Ind. Electron. (2014).
[9] Y. Gao, M. Wang, R. Ji, Z. Zha, J. Shen, k-partite graph reinforcement and its application in multimedia information retrieval, Inform. Sci. 194 (2012) 224–239.
[10] Y. Gao, M. Wang, D. Tao, R. Ji, Q. Dai, 3d object retrieval and recognition with hypergraph analysis, IEEE Trans. Image Process. 21 (9) (2012) 4290–4303.
[11] A. Hanson, Computer Vision Systems, Academic Press, 1977.
[12] G. Hinton, S. Osindero, Y. Teh, A fast learning algorithm for deep belief nets, Neural Comput. 18 (2006) 1527–1554.
[13] M. Hofmann, D. Gavrila, Multi-view 3d human pose estimation in complex environment, Int. J. Comput. Vision 96 (1) (2012) 103–124.
[14] S. Ji, W. Xu, M. Yang, K. Yu, 3d convolutional neural networks for human action recognition, IEEE Trans. Pattern Anal. Mach. Intell. 35 (2013) 221–231.
[15] T. Karnowski, I. Arel, D. Rose, Deep spatiotemporal feature learning with application to image classification, in: Proceedings of 9th International Conference on Machine Learning and Applications, IEEE, 2010, pp. 883–888.
[16] Y. LeCun, Y. Bengio, Convolutional networks for images, speech, and time series, in: The Handbook of Brain Theory and Neural Networks, MIT Press, 1998, pp. 255–258.
[17] H. Lee, R. Grosse, R. Ranganath, A. Ng, Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations, in: Proceedings of the 26th Annual International Conference on Machine Learning, ACM, 2009, pp. 609–616.
[18] J. Li, W. Huang, L. Shao, N. Allinson, Building recognition in urban environments: a survey of state-of-the-art and future challenges, Inform. Sci. 277 (2014) 406–420.
[19] P. Li, M. Wang, J. Cheng, C. Xu, H. Lu, Spectral hashing with semantically consistent graph for image indexing, IEEE Trans. Multimedia 15 (1) (2013) 141–152.
[20] R. Li, T. Zickler, Discriminative virtual views for cross-view action recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2012, pp. 2855–2862.
[21] K. Lu, Q. Wang, J. Xue, W. Pan, 3d model retrieval and classification by semi-supervised learning with content-based similarity, Inform. Sci. (2014).
[22] K. Mikolajczyk, C. Schmid, An affine invariant interest point detector, in: Computer Vision, Springer, 2002, pp. 128–142.
[23] L. Nie, M. Wang, Y. Gao, Z. Zha, T. Chua, Beyond text QA: multimedia answer generation by harvesting web information, IEEE Trans. Multimedia 15 (2) (2013) 426–441.
[24] D. Pelli, B. Farell, D. Moore, The remarkable inefficiency of word recognition, Nature 423 (2003) 752–756.
[25] S. Peng, D. Kim, S. Lee, C. Chung, A visual shape descriptor using sectors and shape context of contour lines, Inform. Sci. 180 (2010) 2925–2939.
[26] T. Pham, Pattern recognition by active visual information processing in birds, Inform. Sci. 270 (2014) 134–142.
[27] R. Rao, D. Ballard, An active vision architecture based on iconic representations, Artif. Intell. 78 (1995) 461–505.
[28] J. Schwartz, M. Sharir, Identification of partially obscured objects in two and three dimensions by matching noisy characteristic curves, Int. J. Robot. Res. 6 (1987) 29–44.
[29] T. Shao, W. Xu, K. Zhou, J. Wang, D. Li, B. Guo, An interactive approach to semantic modeling of indoor scenes with an rgbd camera, ACM Trans. Graph. 31 (2012) 136.
[30] P. Simard, D. Steinkraus, J. Platt, Best practices for convolutional neural networks applied to visual document analysis, in: Proceedings of 7th International Conference on Document Analysis and Recognition, IEEE, 2003, pp. 958–963.
[31] J. Sivic, A. Zisserman, Video Google: a text retrieval approach to object matching in videos, in: Proceedings of 9th IEEE International Conference on Computer Vision, IEEE, 2003, pp. 1470–1477.
[32] M. Song, C. Chen, J. Bu, T. Sha, Image-based facial sketch-to-photo synthesis via online coupled dictionary learning, Inform. Sci. 193 (2012) 233–246.
[33] M. Wang, Y. Gao, K. Lu, Y. Rui, View-based discriminative probabilistic modeling for 3d object retrieval and recognition, IEEE Trans. Image Process. 22 (4) (2013) 1395–1407.
[34] M. Wang, X. Hua, R. Hong, J. Tang, G. Qi, Y. Song, Unified video annotation via multigraph learning, IEEE Trans. Circ. Syst. Video Technol. 19 (5) (2009) 733–746.
[35] M. Wang, H. Li, D. Tao, K. Lu, X. Wu, Multimodal graph-based reranking for web image search, IEEE Trans. Image Process. 21 (11) (2012) 4649–4661.

[36] M. Wang, B. Ni, X. Hua, T. Chua, Assistive tagging: a survey of multimedia tagging with human-computer joint exploration, ACM Comput. Surv. 44 (4) (2012) 25.
[37] Z. Wang, P. Cui, F. Li, E. Chang, S. Yang, A data-driven study of image feature extraction and fusion, Inform. Sci. (2014).
[38] Y. Xia, X. Ren, Z. Peng, J. Zhang, L. She, Effectively identifying the influential spreaders in large-scale social networks, Multimedia Tools Appl. (2014) 1–13.
[39] Y. Xia, Y. Zhou, Synchronization induced by disorder of phase directions, Int. J. Mod. Phys. C 25 (05) (2014).
[40] M. Zerroug, R. Nevatia, Three-dimensional descriptions based on the analysis of the invariant and quasi-invariant properties of some curved-axis generalized cylinders, IEEE Trans. Pattern Anal. Mach. Intell. 18 (1996) 237–253.
[41] L. Zhang, Y. Gao, Y. Xia, K. Lu, J. Shen, R. Ji, Representative discovery of structure cues for weakly-supervised image segmentation, IEEE Trans. Multimedia 16 (02) (2014) 470–479.