2016 IEEE International Conference on Robotics and Automation (ICRA)
Stockholm, Sweden, May 16-21, 2016

Convolutional Hypercube Pyramid for Accurate RGB-D Object Category and Instance Recognition

Hasan F. M. Zaki[1], Faisal Shafait[2] and Ajmal Mian[1]

Abstract— **Deep learning based methods have achieved unprecedented success in solving several computer vision problems involving RGB images. However, this level of success is yet to be seen on RGB-D images owing to two major challenges in this domain: training data deficiency and multi-modality input dissimilarity. We present an RGB-D object recognition framework that addresses these two key challenges by effectively embedding depth and point cloud data into the RGB domain. We employ a convolutional neural network (CNN) pre-trained on RGB data as a feature extractor for both color and depth channels and propose a rich coarse-to-fine feature representation scheme, coined Hypercube Pyramid, that is able to capture discriminatory information at different levels of detail. Finally, we present a novel fusion scheme to combine the Hypercube Pyramid features with the activations of fully connected neurons to construct a compact representation prior to classification. By employing Extreme Learning Machines (ELM) as non-linear classifiers, we show that the proposed method outperforms ten state-of-the-art algorithms for several tasks in terms of recognition accuracy on the benchmark Washington RGB-D and 2D3D object datasets by a large margin (upto 50% reduction in error rate).**

I. INTRODUCTION

Recognizing unseen objects and instances in complex environments is a highly desirable capability for an interactive robot. Developing this capability usually involves training the robot in an off-line mode, where the robot is given a set of training data with corresponding labels and is asked to predict the labels for new instances during test time. To develop a high performance recognition system, several design considerations need to be met. Firstly, labeled training examples must come in abundance to ensure good generalization of the trained model. Secondly, feature descriptors must be both representative and discriminative to mitigate the effect of high intra- and inter-class variability. Additionally, the system must be computationally efficient especially at test time to ensure feasibility for robotics applications.

Traditional object recognition methods rely on features extracted from color images obtained by a typical RGB camera. Recent advances in deep learning methods, especially convolutional neural networks (CNN), have resulted in high accuracy recognition systems for RGB images [1]. The advent of low-cost depth cameras, has opened up a number of possibilities to use depth information to extract more informative features. However, the availability of labeled RGB-D training data is much more restricted compared to the traditional RGB images. Moreover, in contrast to the conventional RGB based recognition, the multi-modal nature of RGB-D images poses additional challenges such as noisy data and input data dissimilarity. Recent research works have aimed at addressing these problems [2], [3], [4], [5], [6], [7], [8], [9], [10], with a particular emphasis towards the unavailability of large scale training datasets due to immensely expensive data capturing and annotation. More importantly, this data deficiency problem has significantly constrained employing powerful but data-hungry deep learning algorithms for model training in the RGB-D domain.

Recently, feature representation techniques have also seen rapid evolution from hand-engineered descriptors to the methods based on learning algorithms that extract semantically descriptive feature sets. The most prevalent techniques are based on the new generation of CNN which have constituted state-of-the-art performance on a wide range of applications [11], [12], [13], [14]. Generally, recognition algorithms in this context rely on the features represented by the fully-connected layers towards the end of the network. While these layers carry rich semantic information, they are too coarse spatially [14] and thus need to be complemented by expensive pre-processing steps such as data augmentation and segmentation [11], [12].

We address these problems by formulating an effective recognition framework based on a deep CNN which has been pre-trained on a large-scale RGB dataset (*i.e.* ImageNet [1]). Precisely, we transfer the knowledge of the CNN model to the domain of depth channel and point cloud by proposing a novel embedding technique that allows a seamless integration of these differing domains. We define a novel Hypercube representation that encodes activations of all convolutional layers to preserve spatially discriminative features, in addition to the semantically-informative global features extracted from the fully connected layer. For multi-scale feature extraction, we devise a coarse-to-fine scheme based on pyramidal re-sampling of the convolutional feature maps. A spatial pyramid pooling scheme is then used at each pyramid level before feature concatenation to produce a compact feature representation. The feature vectors from these pyramids are max pooled to produce a single Hypercube Pyramid descriptor. Finally, these feature vectors from multiple levels and modalities are given as input to Extreme Learning Machine classifiers to perform object category and instance recognition. The core idea is illustrated in Fig. 1. In summary, our core contributions are four-fold:

1) We propose a Hypercube Pyramid descriptor as a dis-

[1]Hasan F. M. Zaki and Ajmal Mian are with the School of Computer Science and Software Engineering, The University of Western Australia, Crawley, WA 6009, Australia `hasan.zaki@research.uwa.edu.au`, `ajmal.mian@uwa.edu.au`

[2]Faisal Shafait is with the National University of Sciences and Technology, Pakistan `faisal.shafait@seeks.edu.pk`

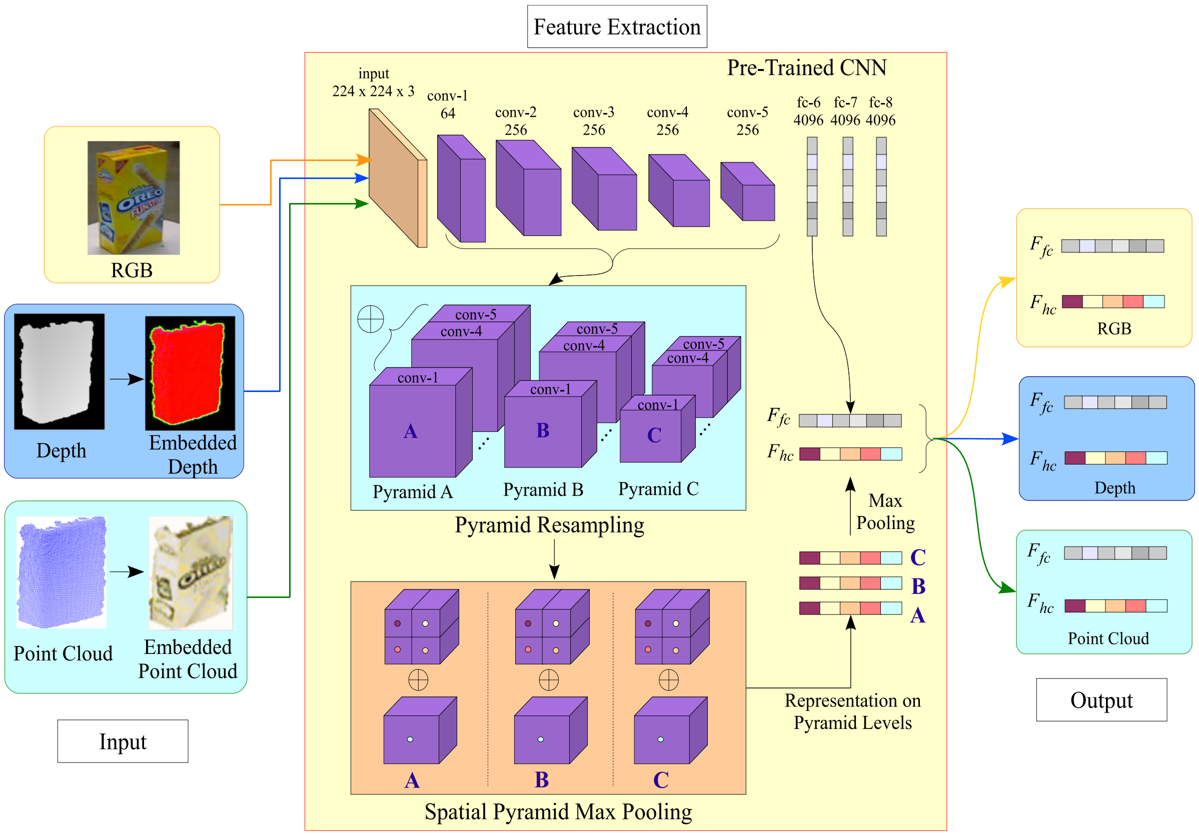

Fig. 1. Illustration of the proposed Convolutional Hypercube Pyramid feature extraction. The feature representation is extracted in a coarse-to-fine manner separately for each RGB, depth and point cloud image. This is done by resampling the convolutional feature maps into three pyramid levels *A*, *B*, and *C* and concatenate all feature maps at each pyramid level separately (refer text for detailed description). The symbol ⊕ denotes concatenation.

criminative feature representation that encodes multi-scale, spatially-relevant information for RGB-D object and instance categorization (Section III-B).

2) We present a novel RGB embedding technique for both depth maps and point cloud data to allow seamless transfer of knowledge between different input modalities (Section III-A).
3) We show that a pre-trained CNN on RGB images can be effectively used for feature extraction from embedded depth data, thereby avoiding the need for large scale labeled RGB-D data to train a deep network.
4) We propose a feature fusion technique based on Extreme Learning Machines to combine features from the Hypercube Pyramid and fully connected neurons leading to a compact but powerful representation for efficient classification (Section III-C).

II. RELATED WORK

Prior works in RGB-D object recognition utilized channel-specific hand-engineered feature descriptors for colour and 3D domains such as SIFT [15] and spin images [2]. The combination of these descriptors is reduced to encoding using the Bag-of-Words (BoW) method [2] or a kernel-based representation [3]. Despite their simplicity, these methods heavily rely on the prior knowledge of the input distribution, which is not readily available in most real-time robotic applications (*e.g.* grasping, navigation). Recent research in feature learning has opened a new perspective in feature extraction techniques where the representation of multiple differing channels can be explicitly learned using a unified learning algorithm. These include the previously proposed Convolutional K-Means (CKM) [16], Convolutional-Recursive Neural Networks (CNN-RNN) [5], Hierarchical Matching Pursuit (HMP) [4], deep Regularized Reconstruction Independent Component Analysis (R^2ICA) [10], Cascaded Random Forests (CaRFs) [8] and Localized Deep ELM (LDELM) [25]. However, these methods do not address the problem of input dissimilarity between RGB and depth channels and thus resort to independently learn from each channel for representation. Besides, these methods learn models from relatively small training datasets which often leads to sub-optimal performance.

While collecting and annotating 3D datasets are prohibitively expensive, data from the RGB domain are abundantly available (*e.g.* Imagenet [1]) which makes the training of data-hungry algorithms such as CNN more feasible. In recent works, the pre-trained CNN models have proven to be powerful for the task they are trained on as well as for generalizing to other applications [11], [12] without the need for re-training. However, it is still unclear how feasible it is to transfer the models across differing modality domains (*e.g.* RGB to 3D). Gupta *et al.* [13] proposed to encode the depth channel as the combination of height above ground, horizontal disparity and angle with gravity (HHA). This embedding method is strictly geocentric and such information is not always available in object-centric recognition. Schwarz *et al.* [7] demonstrated that the transfer learning can be partially done using a simple colourization scheme of the depth images. Their approach follows conventional methods in using the fully connected layers as feature representation, ignoring the earlier convolutional layers. In contrast, our proposal motivates knowledge transfer in a novel embedding scheme for both depth images and point cloud and utilizes all earlier layers in the deep network.

Following the success of fully connected neurons in recognition tasks, recent studies have shown that convolutional layers in CNN networks also contain a degree of semantically meaningful features. Hariharan *et al.* [14] used the concatenated convolutional layers in local regions as pixel-wise feature representations. Liu *et al.* [17] proposed guided cross-layer pooling to extract sub-array of convolutional layers as local features. In this paper, we consider all convolutional feature maps as global features and devise a novel extraction and pooling scheme to encode a compact, discriminative representation.

III. PROPOSED METHODOLOGY

The overview of our proposed Convolutional Hypercube Pyramid is illustrated in Fig. 1. Throughout this work, we deploy the deep pre-trained CNN model from [11] called VGG-f, which has a similar configuration as that of the AlexNet [1], with five successive convolutional layers followed by three fully connected neural layers. In the first step, our algorithm embeds the depth channel into the RGB domain to be able to perform feature extraction directly from the pre-trained deep network (Sec. III-A). After feed-forwarding each instance through the CNN separately for each input channel, we define the Hypercube Pyramid representation with associative re-sampling and feature pooling. Finally, inference is done with multi-class non-linear Extreme Learning Machine (ELM) classifiers using a late fusion scheme combining both Hypercube Pyramid features and the fully connected neural layer activations.

A. Depth Maps and Point cloud Embedding

RGB-D sensors produce two input channels with complementary and incongruous information. The objective of depth embedding is to render the depth information as RGB in a domain adapted manner, in order to allow knowledge transfer from the pre-trained CNN model. To embed richer depth information, we use both depth images and point cloud data to independently render two RGB images. Given a single channel depth map, $d(u)$, where $u = (x, y)$ and d denotes pixel-wise depth value at the x-y location, we calculate the vertical and horizontal derivative approximations by:

$$\begin{aligned} G_y &= K_y * d(u) \\ G_x &= K_x * d(u), \end{aligned} \tag{1}$$

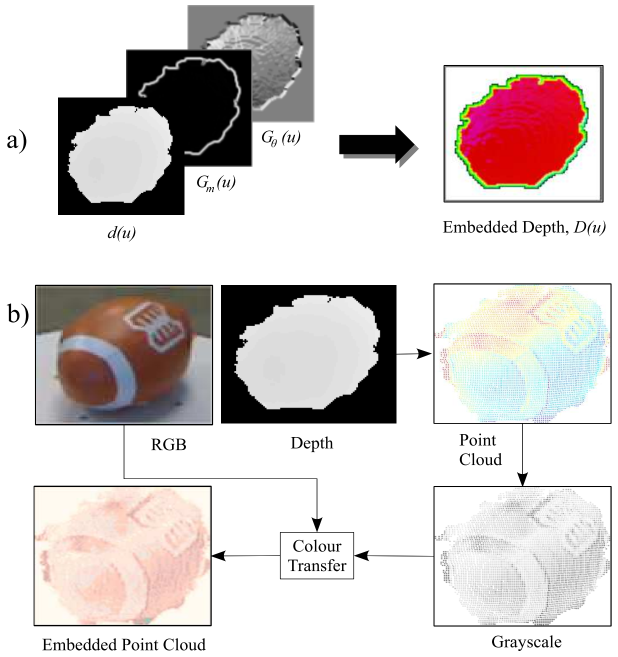

Fig. 2. Illustration of the proposed technique for CNN input embedding a) depth image and b) point cloud object-centric embedding before feature extraction.

where K_y and K_x are the vertical and horizontal Prewitt kernels respectively, and $*$ is a two-dimensional convolution operator. We then compute the gradient magnitude, $G_m = \sqrt{G_y^2 + G_x^2}$ and the gradient direction, $G_\theta = \arctan(G_y, G_x)$. The three-channel depth map is constructed as the concatenation of the original single channel depth map with the gradient magnitude and direction maps, given by $D(u) = \{d(u), G_m, G_\theta\}$. The motivation of this technique is to encode the texture of the object using the gradient direction while gradient magnitude explicitly draws sharp edges at object boundaries. The result of this embedding is depicted in Fig. 2a which shows that the concatenation of the three channels captures rich shape information.

For the embedding of the point cloud $p^{(i)} = \{a^{(i)}, b^{(i)}, z^{(i)}\}$, $i \in 1, \ldots, P$, we first project the point cloud onto its canonical view[1] and then apply a colour map along the direction of the depth axis. The colourized point cloud is then converted to gray-scale and a colour transfer algorithm [18] is applied with the corresponding colour image to approximate the RGB values at each pixel. This technique transfers the chromatic information from the source image (RGB) to the target image (gray-scale) by matching the luminance and texture between these images. The main advantage of this technique over existing embedding methods [7], [13] is that the colourization scheme is closely guided by the RGB images and it is fully automatic. The depiction of this technique in Fig. 2b suggests that the resulting image closely resembles the corresponding RGB channel except that it has additional

[1]In practice, we define the canonical view as the -27.5°and 20°off the azimuth and elevation angles

depth and shape information.

B. Convolutional Hypercube Pyramid

Suppose we have a CNN with L convolutional layers for each RGB, depth and point cloud channels. Following a rescaling of the input image to the desired model input (*i.e.* $m \times m$; $m = 224$ for our model) and mean normalization by subtracting each image from the average image of the ImageNet [1] database, the feed-forward of the network consists of consecutive operations of convolution, pooling, and local contrast normalization (LCN) modules in the convolutional layers, followed by fully connected layers towards the end of the network. Most conventional methods [12], [13], [7] only consider the fully connected layers for classification purpose since these layers contain rich semantic information. However, earlier convolutional layers carry locally-activated features [14], [17], which are largely ignored in methods using only the fully connected layers. To encode spatially-relevant information into the feature representation, we formulate a novel framework to extract features from all convolutional layers, which are used together with the holistic features extracted from the fully connected layers to construct a powerful representation for object recognition.

Note that our description of Hypercube Pyramid only considers one input domain (*e.g.* RGB) since we perform feature extraction for other domains (embedded depth image and point cloud) using the same procedure. For each convolutional layer $l^{(i)} = \{l^{(1)}, \ldots, l^{(L)}\}$, the feature maps activation at the convolutional node can be mathematically expressed as

$$a^{(l)}_{i,j,n^{(l)}} = \sigma(\sum_{w,h,c} k_{w,h,c,n^{(1)}} * a^{(l-1)}_{i-w,j-h,c} + b^{(l)}_{i,j}). \quad (2)$$

In Eq. 2, $\sigma(.)$ denotes the Rectified Linear Unit (ReLU) non-linear function, and b is a bias term. k indicates the three-dimensional w-by-h-by-c learned filter kernels such that it convolves the c feature maps at previous layer ($l-1$) to produce n feature maps with dimension i-by-j at the current layer l. As depicted in Fig. 1, the number of feature maps (*i.e.* the depth of the convolutional layers) in each convolutional layer is $n^{(l)} = \{64, 256, 256, 256, 256\}$, resulting in a total of $N = 1088$ feature maps.

To convert the feature maps into the Hypercube representation that encodes multi-scale information, first each convolutional feature map is sub-sampled into three pyramid levels. Specifically, we sub-sample the spatial dimension (i, j) of each feature map in all convolutional layers into $p^{(1)} = m \times m, p^{(2)} = 2m \times 2m$ and $p^{(3)} = 0.5m \times 0.5m$ respectively using bilinear interpolation in order to capture distinctive features of the convolutional layers at multiple scales [15]. Then, we concatenate them together along the depth dimension separately at each pyramid level to produce a pyramid of Hypercube descriptors (see Fig. 1 for illustration). Concretely, our Hypercube at each pyramid level P is given by

$$H_P = [a^{(1)}_{p,n^{(1)}}, a^{(2)}_{p,n^{(2)}}, \ldots a^{(L)}_{p,n^{(L)}}], \quad (3)$$

with $H_k \in \mathbb{R}^{p^{(k)} \times N}$ where $k = 1, \ldots, P$.

This operation produces three Hypercube of different spatial sizes. Then, in order to enhance the discriminative characteristics of the descriptors as well as to reduce the dimensionality of the Hypercube, we perform spatial pyramid max pooling (SPM) [4] where the Hypercube at each of the pyramid levels are divided into two (SPM) levels. The whole Hypercube is used as one cell for SPM level one, whereas the Hypercube are partitioned into four equi-sized cells for SPM level two. Then, the pooled feature vectors for each cell can be calculated simply as the component-wise maxima over all feature maps within that cell. Note that the dimension of feature vectors extracted from each cell is equal to the depth (N) of the respective Hypercube. This step generates five equal-dimensional feature vectors which are then concatenated to create a single vector for each of the three pyramid levels. Finally, max pooling is performed again to combine the three feature vectors to produce a compact discriminative representation $F_{hc} \in \mathbb{R}^{5N}$ of the pyramidal Hypercube for classification.

C. Feature Fusion and Inference Using Extreme Learning Machines

Existing methods which formulate feature representation based on CNN directly use the feature vectors [7], [12] or use simple concatenation of feature vectors from convolution layers and fully connected neurons, F_{fc} [14], [17] as input to the classifiers (*e.g.* Support Vector Machines) for class inference. Although these methods are straight-forward in implementation, simple concatenation for example, comes at the expense of producing long feature vectors which will increase the computational complexity at test time, especially if used in conjunction with more powerful classifiers such as those with non-linear kernels [19].

Conversely, we would want our classifier input to be compact in the feature space without sacrificing the discriminative properties of the feature representation. This can be achieved by employing the Extreme Learning Machines classifier [19], [20]. We use ELM not only for multi-class object classification, but also as the feature fusing engine to combine our Hypercube representation F_{hc} with the fully connected neurons F_{fc}. We investigate both early fusion and late fusion strategies to identify the most accurate classification scheme. Assume $F_c = \{f_c^{(i)}, t^{(i)}\}, \in \mathbb{R}^D, i = 1, 2, \ldots, N$ represents the feature vectors used as input for classification, where N is the total number of RGB-D images with target labels t. In the early fusion, F_c is simply the concatenation of both F_{hc} and F_{fc} (*i.e.* $F_c = [F_{hc}, F_{fc}]$). The ELM begins by mapping the feature vectors onto the hidden layer to output $h = \sigma(\sum_{i=1}^{N} W_{in} f_c^{(i)} + b_{in}) \in \mathbb{R}^H$, where $\sigma(.)$, $W_{in} \in \mathbb{R}^{H \times D}$ and b_{in} are the piecewise sigmoidal activation function, randomized orthogonal input weight matrix and the bias vector, respectively. Subsequently, the hidden variables are mapped onto the target labels, parametrized by the output weight W_{out} and bias b_{out} which gives the output variables $y = \sigma(\sum_{i=1}^{N} W_{out}t^{(i)} + b_{out})\beta$.

Hence, this leaves β as the only parameter to be tuned. The tuning process is done by optimizing an objective function of ELM which simultaneously minimizes the norm of the output weight and the loss between the actual output and the target labels which can be compactly written as

$$\min_{\beta} \mathcal{J}_{ELM} = \frac{1}{2}\|\beta\|_F^2 + \frac{\lambda}{2}\|h\beta - T\|_2^2. \quad (4)$$

The closed-form solution of Eq. 4 follows the linear least square, $\beta = h^{\dagger}T$, where $h^{\dagger}$ is the generalized Moore-Penrose pseudo-inverse of h. Based on orthogonal projection method, we compute $h^{\dagger}$ as $h^{\dagger} = (\mathrm{I}\lambda^{-1} + h^T h)^{-1}h^T$ or $h^{\dagger} = h^T(\mathrm{I}\lambda^{-1} + hh^T)^{-1}$ with the condition that $h^T h$ is non-singular (if $H > N$) or hh^T is non-singular (if $H < N$) [20]. Here, $\mathbf{I}$ is an identity matrix. The regularization coefficient λ enhances the generalization ability of ELM by acting as a solution stabilizer against overfitting [20].

For the case of late fusion, we first independently feed F_{hc} and F_{fc} as input to the ELM classifiers and optimize Eq. 4. The outputs of this operation are the class probabilities, y_{hc} and y_{fc}, for the Hypercube Pyramid and fully connected neurons independently. Then, the concatenation of these new vectors (*i.e.* $F_c = [y_{hc}, y_{fc}]$) is used as an input vector for another ELM for final classification. The advantages of this method are two-fold: First, the feature vector dimension at the final classification stage is reduced to only double the number of classes. Second, the learning and inference time of ELM is substantially reduced. Fig. 3 provides a depiction of the proposed late fusion scheme.

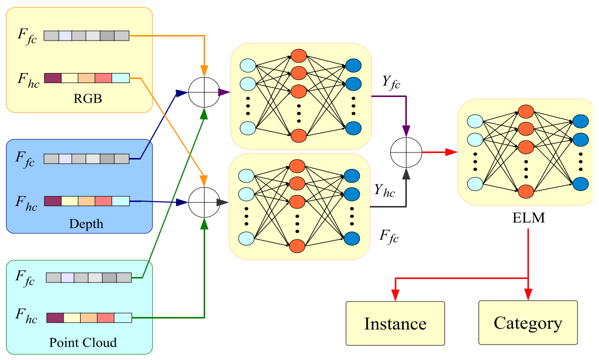

Fig. 3. Illustration of the proposed late fusion scheme to combine Hypercube Pyramid descriptor F_{hc} and the fully connected neurons F_{fc} for the combined RGB-D channels. $\oplus$ means concatenation.

IV. Experimental Evaluation

The presented method is evaluated on the publicly available Washington RGB-D (WRGB-D) [2] and 2D3D [21] datasets that are widely used for benchmarking RGB-D object recognition algorithms. We use the MatConvNet toolbox [22] from the open-source VLFeat library [23] to get the pre-trained CNN model as described in Sec. III. Note that for this paper, we do not fully optimize the parameters of ELM including the number of hidden neurons H and regularization coefficient λ. We provide the empirically chosen parameter values used in all experiments for the sake of reproducibility of the presented results. Next, we will briefly describe the datasets, discuss several model ablations and finally compare our method against several state-of-the-art algorithms.

A. Washington RGB-D Object Dataset

WRGB-D dataset contains 300 household object instances which were organized into 51 categories. Each instance was captured using an ASUS Xtion Pro Live camera on a revolving turntable from three elevation angles ($30°, 45°$ and $60°$). Following the experimental setup of Lai *et al.* [2], we conducted two sets of experiments, namely category recognition and instance recognition. In category recognition, a "leave-one-instance-out" procedure is carried out over ten trials. To ensure a fair comparison with other methods, we use the same training/ testing splits and the cropped images as suggested by Lai *et al.* [2][2]. For instance recognition, the images captured at 45° for each object instances are used as validation sets and the training is done on the rest.

1) Model Ablation Study: Firstly, we examine different modules of our proposed method to find the best-performing architecture. For that purpose, we compare the early and late fusion schemes of the Hypercube Pyramid representation as described in Section III-C, with the baseline of our model which is the fully connected neurons (FC-6). For category recognition, we set the parameters $\{H, \lambda\}$ as $\{5000, 1e5\}$ and $\{13000, 1e13\}$ for FC-6 and early fusion respectively. As for the late fusion, the parameters are fixed at $\{10000, 1e10\}$ for the first two ELMs to get the class probabilities y_{hc} and y_{fc} and $\{1000, 1e6\}$ is used for the final classification. For instance recognition, $\{H, \lambda\}$ are set as $\{13000, 1e13\}$ for all subtasks, with the exception of the late fusion final classification where we set them at $\{1000, 1e6\}$.

As depicted in Table I, the combination of our Hypercube Pyramid with late fusion technique consistently outperforms the other two alternative modules by a significant margin. This is mainly credited to the fusion technique which uses the class probability distribution obtained from different feature channels as the new feature vectors for classification. The testing time for late fusion features is only 6.3×10^{-5} seconds for one image using MATLAB on a 64-bit, 2.5 GHz machine. The accuracy for FC-6 and early fusion are similar for category recognition, which shows that conventional fusion schemes using a simple concatenation (early fusion) are less effective for combining the features originating from different sources. This is probably because of the difficulty faced by the classifier in suitably weighing the cross-channel inputs.

To show that the use of point clouds in addition to depth images improves object recognition accuracy, we conducted a separate experiment in which we performed object category recognition using depth images and point clouds in isolation.

[2]Available: http://rgbd-dataset.cs.washington.edu/

TABLE I

MODEL ABLATION OF THE PROPOSED METHOD FOR OBJECT AND INSTANCE RECOGNITION IN WASHINGTON RGB-D OBJECT DATASET [2]. THE REPORTED ACCURACY (IN %) IS THE AVERAGE ACCURACY OVER TEN SPLITS.

	Category Recognition			Instance Recognition			2D3D		
Model	RGB	D	RGB-D	RGB	D	RGB-D	RGB	D	RGB-D
FC-6	85.5±2.1	79.6±1.8	87.6±1.7	95.1	48.0	94.6	86.1	88.1	88.5
Hypercube Pyramid (Early Fusion)	85.9±1.9	81.2±2.1	87.9±1.8	94.8	28.1	86.7	86.0	87.4	89.5
Hypercube Pyramid (Late Fusion)	**87.6±2.2**	**85.0±2.1**	**91.1±1.4**	**95.5**	**50.2**	**97.2**	**90.6**	**91.6**	**94.3**

Using Hypercube Pyramid representation with the late fusion scheme, the accuracy of using only the depth images or point clouds was 79.4% and 70.3% respectively. However, when both channels were fused together, the accuracy increased to 85% as depicted in Table I. This indicates that depth images and point clouds contain complementary information and augment each other to provide richer 3D information resulting in improved recognition performance.

For instance recognition, we observe an interesting pattern in the classification accuracy where the performance drops when the Hypercube Pyramid representation is combined with the FC-6 using early fusion. This trend shows that while the Hypercube Pyramid is powerful for categorical classification, it is less effective for more fine-grained tasks such as instance recognition. Moreover, when we individually classify objects using the Hypercube Pyramid, we obtain accuracy similar to the FC-6. Nevertheless, the accuracy increases when the late fusion scheme is used to combine the features, showing that the two representations contain complementary information.

2) Comparative Study: To compare the accuracy of our algorithm with the state-of-the-art, we benchmark against ten related algorithms including EMK-SIFT [2], Depth Kernel [3], CNN-RNN [5], CKM [16], HMP [4], semi-supervised learning (SSL) [6], subset-based deep learning (subset-RNN) [24], CNN-colourized [7], CaRFs [8] and LDELM [25]. All results are taken from the original publications and included in Table II.

The results depict the superiority of our proposed method which constitutes state-of-the-art for several subtasks for WRGB-D. For object category recognition, our method outperforms other methods by a significant margin. Our method achieves 91.1% accuracy which is 1.7% more than the closest competitor CNN-colourized [7]. Other methods which extract features from additional derivative channels such as surface normals [4] and point cloud surfels [8] do not perform as good as our three-channel feature extraction. Additionally, our choice of features substantially reduces the processing time needed to extract them from depth and point cloud channels. Our method also outperforms other methods for channel-specific category recognition. The accuracy of our RGB-only recognition improves state-of-the-art by 4.5%, which can be attributed to our novel CNN-based Hypercube Pyramid representation. The significant

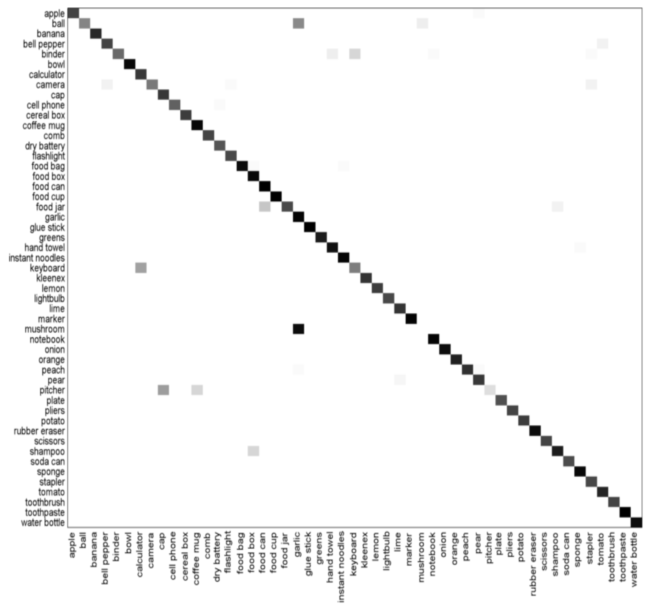

Fig. 4. Confusion matrix for one of the object categorization trials using the proposed Hypercube Pyramid representation and late fusion scheme on the Washington RGB-D object dataset [2]. This figure is best viewed with magnification.

performance improvement for depth-only recognition is an interesting result. It shows that the features extracted from a pre-trained CNN on RGB-only images were powerful enough to achieve high accuracy even when the underlying data was coming from a different domain. Hence, using appropriate embedding and rendering techniques, such as our proposed depth and point cloud embedding (Sec. III-A), seamless transfer of knowledge from other domains is possible.

Our technique also outperforms other methods for instance recognition by a large margin, except for depth-only recognition in which LDELM [25] descriptor wins with a reported accuracy of 54.3%. While this can be attributed to the heavily tuned deep networks from different derivative depth channels, the accuracy is largely inferior for RGB and RGB-D recognition compared to our proposed algorithm. We observe that for instance recognition, colour information provides better discrimination across intra-class instances while they generally share very similar shapes (*e.g.* balls are spherical, soda cans are cylinderical). Nonetheless,

TABLE II
PERFORMANCE COMPARISON IN TERMS OF RECOGNITION ACCURACY (IN %) OF THE PROPOSED HYPERCUBE PYRAMIDS WITH STATE-OF-THE-ART METHODS ON WASHINGTON RGB-D OBJECT DATASET [2]. THE ACCURACY IS REPORTED IS AN AVERAGE OVER 10 TRIALS. THE METHODS WITH $\star$ LABEL INDICATES THE EXPERIMENTS WERE CONDUCTED USING THE SAME TRAINING/ TESTING SPLITS.

Recognition Type		Category Recognition			Instance Recognition		
Method		RGB	D	RGB-D	RGB	D	RGB-D
EMK-SIFT [2] $\star$	ICRA '11	74.5 ± 3.1	64.7 ± 2.2	83.8 ± 3.5	60.7	46.2	74.8
Depth Kernel [3] $\star$	IROS '11	77.7 ± 1.9	78.8 ± 2.7	86.2 ± 2.1	78.6	54.3	84.5
CNN-RNN [5]	NIPS '12	80.8 ± 4.2	78.9 ± 3.8	86.8 ± 3.3	–	–	–
CKM [16]	ICRA '12	–	–	86.4 ± 2.3	–	–	90.4
HMP [4] $\star$	ISER '13	82.4 ± 2.1	81.2 ± 2.3	87.5 ± 2.9	92.1	51.7	92.8
SSL [6]	ICPR '14	81.8 ± 1.9	77.7 ± 1.4	87.2 ± 1.1	–	–	–
subset-RNN [24]	Neurocomp.'15	82.8 ± 3.4	81.8 ± 2.6	88.5 ± 3.1	–	–	–
CNN-colourized [7]	ICRA '15	83.1 ± 2.0	–	89.4 ± 1.3	92.0	45.5	94.1
CaRFs [8] $\star$	ICRA '15	–	–	88.1 ± 2.4	–	–	–
LDELM [25] $\star$	DICTA '15	78.6 ± 1.8	81.6 ± 0.7	88.3 ± 1.6	92.8	**55.2**	93.5
Hypercube Pyramid $\star$	this work	**87.6 ± 2.2**	**85.0 ± 2.1**	**91.1 ± 1.4**	**95.5**	50.2	**97.2**

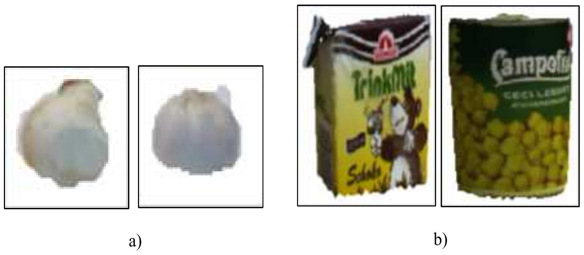

Fig. 5. Selected outliers for a) WRGB-D [2] (*mushroom* misclassified as *garlic*) b) 2D3D [21] (*drink carton* misclassified as *can*).

this problem can be effectively mitigated by considering colour and depth features in unison. Figure 4 visualizes the confusion matrix for one of the category recognition trials on the WRGB-D dataset. The strongest off-diagonal element shows the mis-classification of *mushroom* which is labelled as *garlic*. As depicted in Fig. 5a, this is due to both instances having similar appearance and shape which makes the recognition task difficult even for human experts. In addition, the category *mushroom* has a very low number of examples in the dataset, which makes it hard for the classifier to construct a good model for inference. We conjecture that the performance can be further improved by performing data augmentation techniques such as jittering [11], [12] to increase the number of training samples and the accuracy is taken as an average prediction from all augmented images.

B. 2D3D Object Dataset

In this experiment, we examine the performance of the proposed Hypercube Pyramid on 2D3D dataset [21]. Contrary to the WRGB-D, this dataset has a relatively lower number of instances (163 objects organized into 16 categories) and consists of highly textured common objects (*e.g.* drink cartons, computer monitors). We carefully replicate the procedure set forth by Browatzki *et al.* [21] to ensure a fair comparison with other state-of-the-art methods. Specifically, we combine the classes *fork*, *knife* and *spoon* into a joint class of *silverware* and exclude *phone* and *perforator* due to

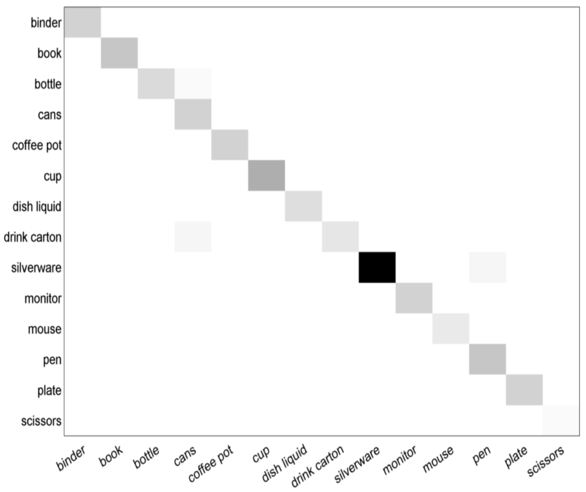

Fig. 6. Confusion matrix for object categorization using our proposed Hypercube Pyramid on the 2D3D object dataset [21]. Superior recognition accuracy is recorded as indicated by the strong diagonal entries.

their small number of examples. This makes a final dataset of 156 instances and 14 classes for category recognition. For evaluation, six instances per class are randomly chosen for training. Validation is done on the remaining instances, while only 18 RGB-D frames per instance are randomly selected for both sets. However, for categories that have less than six instances (*e.g. scissors*), we ensure that at least one instance is used in the validation set. The parameters of ELM for this dataset are set as follows: $\{H, \lambda\} = \{5000, 1e5\}$ for FC-6 and the same $\{H, \lambda\} = \{10000, 1e10\}$ are used for early fusion and for class probabilities generation for late fusion. As the number of categories is very small, we set the parameters as $\{70, 2\}$ for the final classification ELM.

The depiction of model ablation and comparison to other existing methods are reported in Table I and Table III respectively. In Table I, the results for FC-6 and early fusion are comparable, which supports our hypothesis made in Section IV-A.1 regarding the low effectiveness of simple concatenation for object recognition. However, when late fusion is performed, the accuracy shows significant improvement for RGB, depth, and combined channels which confirms the efficacy of the proposed method. Without fully optimizing the parameters of the ELM classifiers, the accuracy is higher than previous state-of-the-art methods (Table III). The closest competitor, subset-RNN, which is based on expensive subset generation and recursive neural network at feature extraction, lags 1.5% from the performance of our proposed Hypercube Pyramid. We credit this result mainly to the effectiveness of the depth and point cloud embedding which is also reflected in the higher accuracy achieved by the depth-only recognition compared to the RGB-only recognition. A sample off-diagonal entry of confusion matrix (Fig. 6) is depicted in Fig. 5b for qualitative analysis of a mis-classification case for this dataset.

TABLE III
PERFORMANCE COMPARISON IN TERMS OF RECOGNITION ACCURACY (%) OF THE PROPOSED HYPERCUBE PYRAMIDS WITH STATE-OF-THE-ART METHODS ON 2D3D OBJECT DATASET [21].

Methods		RGB	D	RGB-D
2D+3D [21]	ICCVW '11	66.6	74.6	82.8
HMP [4]	ISER '13	86.3	87.6	91.0
R^{2}ICA [10]	ACCV '14	87.9	89.2	92.7
Subset-RNN [24]	Neurocomp.'15	88.0	90.2	92.8
Hypercube Pyramid	this work	**90.6**	**91.6**	**94.3**

V. CONCLUSIONS

This paper approached RGB-D object recognition problem via a novel CNN-based Hypercube Pyramid representation which utilizes all convolutional layers to construct a powerful representation for recognition. We presented a novel fusion scheme to combine Hypercube Pyramid features with the activations of the fully connected layer, leading to a compact representation and fast inference scheme. We also devise an effective depth and point cloud embedding technique to allow seamless knowledge transfer between the color and depth domains. The performance of the presented method is examined on two benchmark RGB-D datasets, where it consistently outperformed state-of-the-art methods by a significant margin.

ACKNOWLEDGMENT

This research was supported by the Ministry of Higher Education Malaysia and Australian Research Council (ARC) grant DP110102399.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. NIPS*, 2012, pp. 1097–1105.
[2] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view RGB-D object dataset," in *Proc. ICRA*, 2011, pp. 1817–1824.
[3] L. Bo, X. Ren, and D. Fox, "Depth kernel descriptors for object recognition," in *Proc. IROS*, 2011, pp. 821–826.
[4] L. Bo, X. Ren, and D. Fox, "Unsupervised feature learning for RGB-D based object recognition," in *Proc. ISER*, 2013, pp. 387–402.
[5] R. Socher, B. Huval, B. Bath, C. D. Manning, and A. Ng, "Convolutional-recursive deep learning for 3D object classification," in *Proc. NIPS*, 2012, pp. 665–673.
[6] Y. Cheng, X. Zhao, K. Huang, and T. Tan, "Semi-supervised learning for RGB-D object recognition," in *Proc. ICPR*, 2014, pp. 2377–2382.
[7] M. Schwarz, H. Schulz, and S. Behnke, "RGB-D object recognition and pose estimation based on pre-trained convolutional neural network features," in *Proc. ICRA*, 2015.
[8] U. Asif, M. Bennamoun, and F. Sohel, "Efficient RGB-D object categorization using cascaded ensembles of randomized decision trees," in *Proc. ICRA*, 2015, pp. 1295–1302.
[9] W. Liu, R. Ji, and S. Li, "Towards 3D object detection with bimodal deep boltzmann machines over RGBD imagery," in *Proc. CVPR*, June 2015.
[10] I.-H. Jhuo, S. Gao, L. Zhuang, D. Lee, and Y. Ma, "Unsupervised feature learning for RGB-D image classification," in *Proc. ACCV*, 2014, pp. 276–289.
[11] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *Proc. BMVC*, 2014.
[12] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: an astounding baseline for recognition," in *Proc. Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2014, pp. 512–519.
[13] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from RGB-D images for object detection and segmentation," in *Proc. ECCV*, 2014, pp. 345–360.
[14] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik, "Hypercolumns for object segmentation and fine-grained localization," in *Proc. CVPR*, 2015.
[15] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.
[16] M. Blum, J. T. Springenberg, J. Wulfing, and M. Riedmiller, "A learned feature descriptor for object recognition in RGB-D data," in *Proc. ICRA*, 2012, pp. 1298–1303.
[17] L. Liu, C. Shen, and A. van den Hengel, "The treasure beneath convolutional layers: cross convolutional layer pooling for image classification," in *Proc. CVPR*, 2015.
[18] T. Welsh, M. Ashikhmin, and K. Mueller, "Transferring color to greyscale images," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 277–280, July 2002. [Online]. Available: http://doi.acm.org/10.1145/566654.566576
[19] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1, pp. 489–501, 2006.
[20] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Trans. on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 42, no. 2, pp. 513–529, 2012.
[21] B. Browatzki, J. Fischer, B. Graf, H. Bulthoff, and C. Wallraven, "Going into depth: Evaluating 2D and 3D cues for object classification on a new, large-scale object dataset," in *IEEE International Conference on Computer Vision Workshops (ICCVW)*, 2011, pp. 1189–1195.
[22] A. Vedaldi and K. Lenc, "Matconvnet-convolutional neural networks for matlab," *arXiv preprint arXiv:1412.4564*, 2014.
[23] A. Vedaldi and B. Fulkerson, "Vlfeat: An open and portable library of computer vision algorithms," in *Proc. of the international conference on Multimedia*. ACM, 2010, pp. 1469–1472.
[24] J. Bai, Y. Wu, J. Zhang, and F. Chen, "Subset based deep learning for RGB-D object recognition," *Neurocomputing*, 2015.
[25] H. F. Zaki, F. Shafait, and A. Mian, "Localized deep extreme learning machines for efficient RGB-D object recognition," in *Proc. Digital Image Computing: Techniques and Applications (DICTA)*, 2015, pp. 1–8.