Multimodal Recurrent Neural Networks with Information Transfer Layers for Indoor Scene Labeling

Abrar H. Abdulnabi, *Student Member, IEEE*, Bing Shuai, *Student Member, IEEE*,
Zhen Zuo, *Student Member, IEEE*, Lap-Pui Chau, *Fellow, IEEE*, and Gang Wang, *Senior Member, IEEE*

***Abstract*—This paper proposes a new method called Multimodal RNNs for RGB-D scene semantic segmentation. It is optimized to classify image pixels given two input sources: RGB color channels and Depth maps. It simultaneously performs training of two recurrent neural networks (RNNs) that are crossly connected through information transfer layers, which are learnt to adaptively extract relevant cross-modality features. Each RNN model learns its representations from its own previous hidden states and transferred patterns from the other RNNs previous hidden states; thus, both model-specific and cross-modality features are retained. We exploit the structure of quad-directional 2D-RNNs to model the short and long range contextual information in the 2D input image. We carefully designed various baselines to efficiently examine our proposed model structure. We test our Multimodal RNNs method on popular RGB-D benchmarks and show how it outperforms previous methods significantly and achieves competitive results with other state-of-the-art works.**

***Index Terms*—Multimodal learning, RNNs, CNNs, RGB-D Scene Labeling.**

I. Introduction

DATA comes from different sources and in different forms; images, videos, texts and audios. Each of which may complement the other in information content. Thus, multiple data modalities are usually more informative for a task than a single data modality. With the enormous availability of various electronic and digital multimedia devices, huge volumes of multimodal data contents are being generated on the Internet daily. However, for real-world applications, these modalities should be first well integrated and appropriately fused to have more comprehensive information contents. Many methods have been developed in multimodal learning to exploit both the different characteristics as well as the shared relationships between different data modalities in order to perform various tasks [2], [10], [29], [43], [63], [65]. One of the rigid milestones in developing many visual-based data applications is scene understanding. It is mainly applied to understand the contents of an image or video prior performing the target task (e.g. large-scale video retrieval [29]). Imperatively, scene labeling (i.e; semantic segmentation or image parsing) is a crucial part of understanding an outdoor or indoor captured scene image. The task here is to classify each pixel into its semantic category (belonging to some object or stuff) in an input scene image.

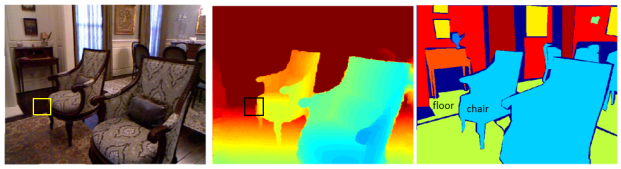


Fig. 1. An example scene image that have both the RGB image and the Depth plane. The corresponding Depth information can be utilized to better differentiate ambiguous appearance in the RGB image.

In our paper, we tackle the problem of RGB-D indoor scene labeling where we process two different data modalities; RGB color channels and Depth planes. Indoor RGB-D scene labeling is one of the most challenging visual classification problems [55], [56]. Many applications are built on understanding the surrounding scenes, e.g. social behavior understanding [49] and objects detection and recognition [70].

This problem is usually addressed as a multimodal learning problem where the task is to exploit and fuse both RGB and Depth modalities to better label each pixel. Depth planes provide informative representation where the RGB representations are ambiguous. For example, Figure 1 show how the depth information can help to distinguish some similar appearance locations in the RGB image.

Scene labeling in general is a challenging classification problem since the scene image tends to contain multiple messy objects. These objects may also have variations due to factors affecting their appearance and geometry in the image. One key useful strategy is to leverage the neighborhood/contextual information for each pixel within each modality [1], [3], [15], [41]. Typically, the feature representation of a pixel is extracted from a local patch (cropped from the scene image) containing that target pixel and used for classification. Long-range/global contextual information (distant image patches) are important as well for local pixel classification. However, both local and global contextual information should be utilized adequately to

A. H. Abdulnabi is working with both the Rapid-Rich Object Search (ROSE) Lab at Nanyang Technological University, Singapore, and the Advanced Digital Sciences Center (ADSC), Illinois at Singapore Pt Ltd, Singapore, Email: abrarham001@ntu.edu.sg. B. Shuai is with ROSE, Z. Zuo is with ROSE, L. Chau and G. Wang are with the Department of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, Emails: elpchau@ntu.edu.sg and wanggang@ntu.edu.sg respectively. Address of ADSC: Advanced Digital Sciences Center, 1 Fusionopolis Way, Illinois at Singapore, Singapore 138632. Address of ROSE: The Rapid-Rich Object Search Lab, School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, 637553.

maintain a good balance between the discriminative features and the abstract/top-level features of that pixel feature representation.

Recently, Recurrent Neural Networks (RNN) has been shown to be very successful on encoding contextual information into local feature representations [6], [54], [72]. Recurrent models have feedback connections so that the current state is engaged in the calculation of the future state. Thus, RNN is effectively used in tasks which require modeling the long and short-range dependency within the input sequence, e.g. speech recognition and natural language processing [18], [19], [21], [32]. We use RNNs to model the contextual information within each modality. However, traditionally, RNN is only used for a single modality signals. In this paper, we introduce a new multimodal RNNs method which models contextual information into local representations from multimodal RGB-D data simultaneously. In our work, we first train Convolution Neural Networks [37] (CNNs) to extract features from local RGB-D image patches (from both the RGB images and the Depth planes). These convolutional local features form the input to our multimodal RNNs to further contextualize them and select informative patterns across the modalities. Our model can be easily extended to perform prediction tasks considering more modalities (> 2).

Our new multimodal RNNs method is built based on the basic quad-directional 2D-RNNs structures [18], [54]. The quad-directional 2D-RNN contains four hidden states where each is dedicated to traverse the image in specific 2D direction out of four possible directions **(top-left, top-right, bottom-left and bottom-right)**. To process two modalities which are the RGB image and the Depth plane; our model has two RNNs, one of which is assigned to learn the representations of a single input modality. To connect both RNN modalities and allow information fusion, we develop information transfer layers that crossly connecting the RNNs. The transfer layers are responsible about learning to select and transfer the relevant patterns from one modality to the other modality, and vice versa. Concretely, for every patch in the input image, and during the process of learning the RNN hidden representations for one modality, our method does not only encode the contextual information within its own modality, but also learns to encode relevant contextual patterns from the other modality. As a result, our method can learn powerful context-aware and multimodal features for local pixel representations.

Our method is different from existing deep multimodal learning methods [38], [45], [58], [59]. They usually concatenate inputs at the beginning or concatenate learned middle level features to extract high-level common features as the representations of the multimodal data. These methods mainly focus on discovering common patterns between different modalities. Although common patterns are important to extract, however these methods are prone to miss important modality specific information that are highly discriminative within a single modality, e.g., some texture patterns inside the RGB channels. Concretely, our model retains modality-specific information by assigning an RNN model to learn features from each modality. Our method also allows sharing information between modalities by using the information transfer layers to adaptively transfer relevant cross-modality patterns.

Our model is trained end-to-end and the transfer layers are learned to extract relevant across-modality information for each patch of the image. We perform experiments on two popular RGB-D benchmarks, the NYU V1 and V2 and achieve comparable performance with other state-of-the-art methods on them. Additionally, the proposed method significantly outperforms its counterparts baselines (e.g. concatenation of the RGB-D data as the input of RNN models).

The remaining parts of our paper are summarized as follows: We first discuss the related work in Section II. Our proposed model alongside the framework details is presented in Section III. Experiments on popular RGB-D benchmarks and results are demonstrated in Section IV. Finally, we conclude the paper in Section V.

II. Related Work

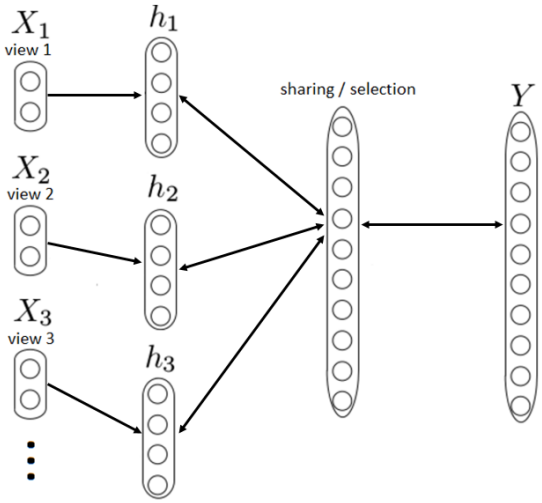
Because this work is mainly related to RGB-D scene labeling and RNN, we briefly review the most recent literature.

Indoor RGB-D Scene Labeling: Many papers propose different methods to solve RGB-D scene labeling [8], [22], [42], [51], [55], [56]. The most popular indoor scene datasets are the NYU depth datasets V1 [55] and V2 [56]. The initial results [55] are generated by extracting SIFT features on the RGB color images in addition to depth maps. Their results prove that depth information can refine the prediction performance.

Further improvements are made to the NYU V1 by the work [51], where they adapt a framework of kernel descriptors which converts the local similarities (kernels) to patch descriptors. They further use a superpixel Markov Random Field (MRF) and a segmentation tree for contextual modeling. Other works [8], [11] explore depth information through a feature learning approach; [8] learns their features using convolution neural network on four channels; three from the RGB image and the fourth is the Depth image. Wang et al. [64] adapt an unsupervised feature learning approach by performing the learning and encoding simultaneously to boost the performance. Another interesting work done by Tang et al. [61] proposes new feature vectors called Histogram of Oriented Normal Vectors (HONV), designed specifically to capture local geometric properties for object recognition with a depth sensor.

Meanwhile some works in RGB image parsing utilize the label dependency modeling approaches, such as graphical models (Conditional Random Fields CRFs [25]), and other works perform feature learning to generate hierarchical multiscale features that are capable of capturing the input context, which is successfully applied through the aid of deep convolutional neural networks [14], [48]. In our multimodal recurrent neural networks, we can capture short and long range context within and between input modalities.

Multi-modal Learning: In contrast to single-view approaches, most multimodal learning methods introduce a separate function to model one data modality and jointly optimize



Conceptual illustration of the typical Multi-modal/Multi-view learning method

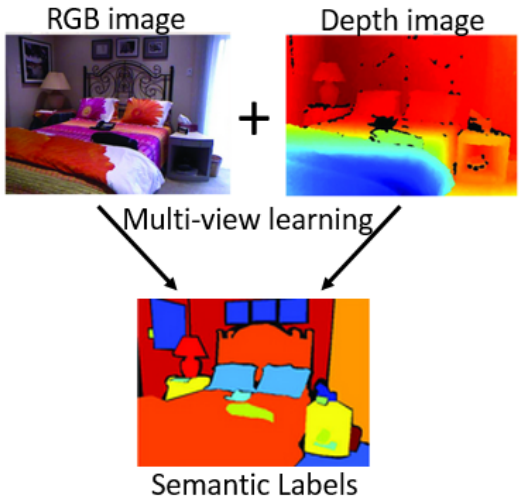


Fig. 2. The top part illustrates the concept behind the multi-modal/multi-view learning method where usually different data modalities are combine by various techniques to exploit all information sources. Multi-modal is also efficient to combine and fuse the RGB color channels with Depth planes (bottom part)

them together [4], [66], [67], [69]. Deep networks have been applied to learn features over multiple views [38], [45], [58], [59]. Multimodal learning improves the classification performance through exploiting the sharable knowledge of the data across modalities.

Figure2 shows the general method of how multimodal learning is applied. Notice that the key idea here is to allow fusion/sharing between the modalities at some point so that the join space is able to capture the relationships between the input modalities. Typical methods in the literature can be categorized based on whether they combine on feature level [9], [50], [62], or classifier level [40], [52]. Some works perform preprocessing (module level) steps on various modalities to generate helpful cues before the learning process, as in [13], [16], [39]. Fusion can be done by combining all features into one high-dimensional vector, or by jointly training multiple classifiers to maximize the mutual agreement

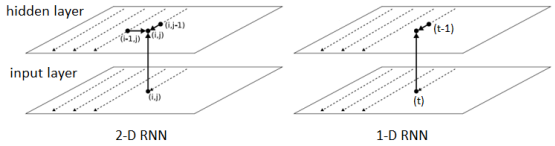


Fig. 3. Illustration of the one-dimensional (1D) RNN (right side) and multidimensional (e.g. two-directional) 2D RNN (left side) [18]. The key idea is to replace the single recurrent connection in the 1D-RNNs with as many recurrent connections as there are dimensions in the input data.

on distinct modalities of the input data [24], [30]. These methods employ typical regularizations which are applied to explore shared visual knowledge, such as group structured sparsity [36].

We can also group multimodal methods according to their training procedures into three main categories: Co-training [66], [69], Multiple kernel learning [67], [69], and Subspace learning [4], [31], [68], [69]. Co-training algorithms tend to train alternatingly to maximize the mutual agreement on two distinct views of data. Multiple kernel learning approaches improve the performance by exploiting different types of kernels that correspond naturally to different views, and combine these kernels either linearly or non-linearly. Meanwhile, Subspace learning methods are mainly similar in obtaining a latent subspace that is shared by multiple modalities. Another interesting line of work proposed by Gupta et al. [53], where they propose to transfer supervision between images from different modalities. Their model is able to learn representations for unlabeled modalities and can be used as a pre-training procedure for new modalities with limited labeled data. In our work and different from all previous methods, we introduce information transfer layers between two RNN modalities to perform the multimodal learning task simultaneously.

Recurrent Neural Networks (RNNs): A recurrent model refers to a model which has connections between its units to form a directed cycle, for example, when a feedback connection from the current state is engaged in the calculation of the future state. This structure creates a sort of complementary internal state for the network, which then allows it to exhibit dynamic temporal behavior. RNN is effectively used in tasks which require sequence modeling, like speech recognition, handwriting recognition, and other natural language processing tasks [18], [19], [21], [32].

One major drawback in the standard RNN is the vanishing gradient problem [27]. This drawback limits the context range of the input data, because the capacity of the model is limited to capture enough long dependencies. To address this problem, Hochreiter and Schmidhuber [28] propose the Long Short Term Memory (LSTM), where they treat the hidden layer as multiple recurrently connected subnets, known as memory blocks, thus allowing the network to store and access information over long periods of time. Graves et al. extend the idea of unidirectional LSTM network into bidirectional networks which have shown good improvements over the unidirectional networks [17], [20].

Graves et al. also extend the one-dimensional RNN into multidimensional one [18], [21] as shown in Figure3. The

key idea is to replace the single recurrent connection in the 1D-RNNs with as many recurrent connections as there are dimensions in the input data. Another interesting work from Graves et al. investigates deep structures of RNNs [19], which is also successfully applied in opinion mining by Irsoy et al. [32]. In our work we evaluate both deep and LSTM structures alongside the basic quad-directional 2D-RNNs. We found that there is no significant difference in the label prediction performance before and after stacking multiple layers of the network or engaging the LSTM units with the basic quad-directional 2D-RNNs. Thus, we only show the results of our multimodal-RNNs model using the basic quad-directional 2D-RNNs.

III. Model Framework

We first extract convolutional features from local RGB-D patches using our trained CNN models. Then, our multimodal-RNNs are developed to further learn context-aware and multimodal features based on the convolutional features. Afterwards, a softmax classifier is trained to classify each patch into its semantic category. Different from traditional single modality RNN, our multimodal-RNNs also have transfer layers to learn to extract relevant contextual information across both modalities at each time step. Below, we first introduce the traditional RNNs.

1D-RNN and 2D-RNN: The popular Elman-type 1D-RNN [12] and its 2D version are designed to capture the dynamic behavior of the signal over time, so that the hidden representations can capture the contextual information from the first time step until the current time step. Its forward pass is formulated as the following:

$$\begin{aligned} h^{(t)} &= f(Ux^{(t)} + Wh^{(t-1)} + b) \\ y^{(t)} &= g(Vh^{(t)} + c) \end{aligned} \tag{1}$$

where $x^{(t)}$, $y^{(t)}$ and $h^{(t)}$ are the input, output and hidden neurons at the time t respectively. The functions $f(.)$ and $g(.)$ are element-wise non-linear functions with bias terms b and c, and the matrices U, W and V are the input to hidden, hidden to hidden and hidden to output weights, respectively. The 2D-RNN [18], [54], [72] is generalized from the 1D-RNN so that the data propagation will come from two dimensional neurons instead of one, thus the formulation becomes:

$$\begin{aligned} h^{(i,j)} &= f(Ux^{(i,j)} + Wh^{(i-1,j)} + Wh^{(i,j-1)} + b) \\ y^{(i,j)} &= g(Vh^{(i,j)} + c) \end{aligned} \tag{2}$$

We can notice now the propagation is in a 2D-plane from top left regions and continues to flow until the end of the 2D sequence (in our case the image patch sequence), where (i, j) denotes the location of pixels or patches.

We show first a conceptual illustration of our proposed model using 1D-RNNs as shown in Figure 5. Concretely, in this paper, we adapt 2D-RNN to learn hidden representations for local RGB-D image patches. Figure 4 shows one scan direction using 2D-RNN, which scans only the top-left sequence. Scanning the image in only one direction leaves some patches in the top-left sequence without being informed of the contextual information from the bottom-right patches during the forward-pass of the testing images. We approximate images using four directional 2D sequences of patches following the work [54]. The other three directions are top-right, bottom-left and bottom-right sequences. We combine the features that are learned from these four directions to obtain the final features of the image patches.

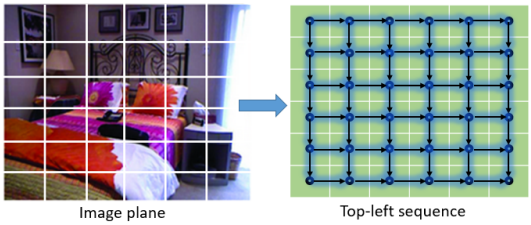


Fig. 4. 2D-RNN structure is a generalization of 1D-RNN. A visualization of processing patches in 2D image is shown. The scan is only presented in one direction (top-left). Here, the previous hidden states from the top and the left sided are encoded into the current hidden state, i.e. the previous top and left neighbored hidden state information are engaged in calculation of the current network state alongside the current patch at the same location. An image is approximated through four directions: top-left, bottom-left, top-right and bottom-right accordingly. Thus, we use the quad-directional 2D design to accumulate information from all sides.

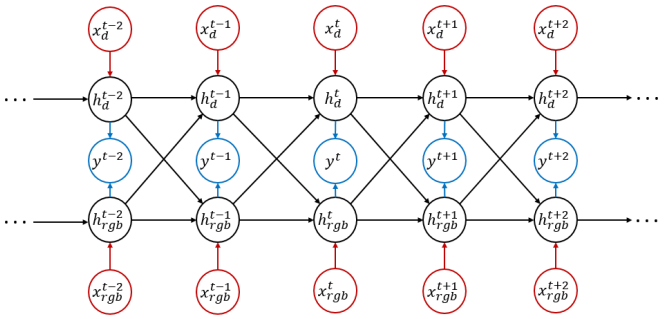


Fig. 5. A conceptual illustration to show our principal design in our proposed multimodal 2D-RNNs. This illustration is based on simple 1D-RNNs and intended to clarify the design structure of two crossly-connected 1D-RNNs through the introduced transfer layers. In our model, since we process 2D input signals (images) we use 2D-RNNs instead of 1D-RNN to retain the spacial dependencies within each scene image.

A. Multimodal RNNs via Information Transfer Layers

Traditional RNNs are developed to model single modality signals. In this work, we extend RNNs to represent RGB-D signal by our multimodal-RNNs, where we have a pair of single RNNs and each of them is assigned to process one modality (either RGB or Depth). Besides, we propose a transfer layer to connect the hidden planes in one RNN model and the other RNN model's hidden planes and vise versa. This transfer layers will learn to adaptively extract relevant patterns from one modality to enhance the feature representations of the other modality. If the Depth-RNN is processing the Depth patch in the sequence at location (i, j), the other RNN, which is RGB-RNN, will be processing the corresponding RGB patch at the same location. The Depth-RNN is also fed with the processed hidden state values obtained from the RGB-RNN and vice versa. The RGB-D data flows concurrently

in both models where their internal processing clocks are synchronized, so that at each time step we process a pair of RGB and Depth local patches simultaneously.

The architecture of our model is summarized in Figure 6. It is an end-to-end learning framework. Recurrent layers and transfer layers are automatically learned to maximize the labeling performance on the training RGB-D data. Compared to the baseline which concatenates RGB-D data and thus mixes the multimodality information (both relevant and irrelevant), our method retains modality-specific information and only shares relevant cross-modality information.

Given one 2D direction in the RNN (the first hidden plane from the quad-directional hidden planes), as shown in Figure 6, the current state of the network that is being processed at (i, j) depends basically on four main previous states. Two are obtained from the network itself and the others are obtained through the transfer layers from the previous states processed in the other modality network, in addition to the input patch features from either the RGB or the depth images at location (i, j). Both networks are synchronized and process the input modalities simultaneously.

Given an RGB image I_c where c refers to 'color' and is processed by RGB-RNN and a depth image I_d where d refers to 'depth' and is processed by Depth-RNN, in this paper, we first extract multiple patches from the images and generate their corresponding convolutional feature vectors to form the input to our multimodal-RNNs model. We denote the corresponding convolutional feature maps as x_c or x_d for each patch in I_c or I_d. Concretely, the forward propagation formulation to process one hidden plane out of the four (quad) directional hidden planes is as the following:

$$
\begin{aligned}
h_c^{(i,j)} &= f(U_c x_c^{(i,j)} + W_c h_c^{(i-1,j)} + W_c h_c^{(i,j-1)} \\
&\quad + \boldsymbol{S_c} h_d^{(i-1,j)} + \boldsymbol{S_c} h_d^{(i,j-1)} + b_c) \\
h_d^{(i,j)} &= f(U_d x_d^{(i,j)} + W_d h_d^{(i-1,j)} + W_d h_d^{(i,j-1)} \\
&\quad + \boldsymbol{S_d} h_c^{(i-1,j)} + \boldsymbol{S_d} h_c^{(i,j-1)} + b_d) \\
z_c^{(i,j)} &= g(V_c h_c^{(i,j)} + c_c) \\
z_d^{(i,j)} &= g(V_d h_d^{(i,j)} + c_d)
\end{aligned}
\tag{3}
$$

where $x_c^{(i,j)}$ is a feature vector of a certain patch at location (i, j) in the RGB image and $x_d^{(i,j)}$ is a feature vector of a certain patch at location (i, j) in the Depth image. $h_c^{(i,j)}$ and $h_d^{(i,j)}$ are the hidden sates inside each RGB-RNN and Depth-RNN respectively. The weight matrices U_c and U_d are responsible to for input-hidden mapping in RGB and Depth modalities respectively. In the other hand, here we have two types of hidden-hidden transformation matrices; within-modality hidden-hidden transformation and across-modality hidden-hidden transformation. W_c and W_d are within-modality hidden-hidden mapping inside the RGB-RNN and the Depth-RNN respectively. Meanwhile, $\boldsymbol{S_c}$ is a transformation weight matrix to transform features from the Depth to RGB hidden states (from the Depth-RNN modality hidden state to the RGB-RNN modality hidden state). $\boldsymbol{S_d}$ is a transformation weight matrix to transform features from the RGB to Depth hidden states (from the RGB-RNN modality hidden state to the Depth-RNN modality hidden state), i.e. these weight matrices act as the **information transfer layers that crossly connect the RGB and Depth hidden states**. Ultimately, the V_c and V_d are the hidden-output transformation matrices in each corresponding modality.

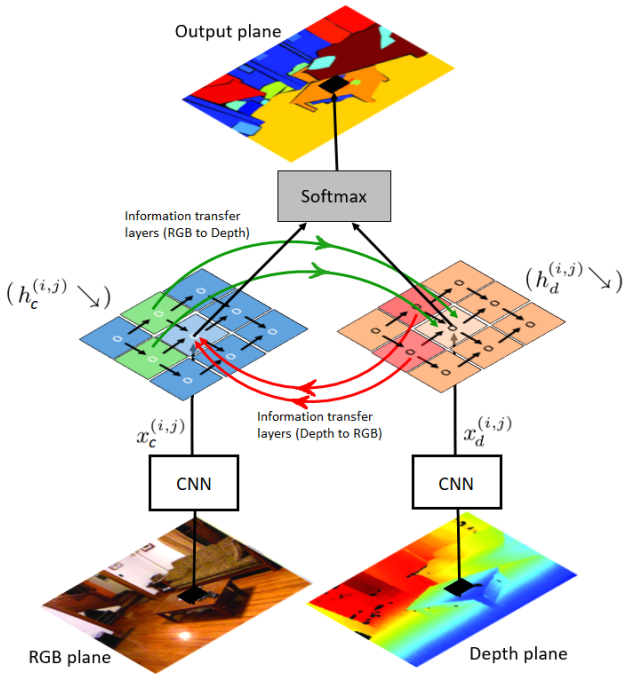


Fig. 6. An abstract overview of our multimodal-RNNs model which consists mainly of an RGB-RNN and a Depth-RNN. The RGB-RNN will process a patch from the RGB input plane in addition to its own previous hidden states and the selections made by the transfer layer from the previous hidden states of the Depth-RNN. The Depth-RNN will also process the corresponding patch from the Depth input plane in addition to its own previous hidden states and the selections made through the transfer layer from previous hidden states of RGB-RNN. For the simplicity in this figure, we only show one hidden plane from the quad-directional RNN hidden planes. Every hidden plane is crossly connected to its peer in the multimodal-RNNs through information transfer layers.

$\boldsymbol{S_c}$ and $\boldsymbol{S_d}$ are learnt to extract shared patterns between the modalities. Notice that the weight matrix that transforms from the top side hidden state in the RGB-RNN $h_c^{(i-1,j)}$ and the weight matrix that transforms from the left side hidden state $h_c^{(i,j-1)}$ are shared which is W_c. Similarly, W_d in the Depth-RNN is shared (transforms from $h_d^{(i-1,j)}$ and from $h_d^{(i,j-1)}$). Also the case in the transfer layers; $\boldsymbol{S_c}$ and $\boldsymbol{S_d}$ are shared. In detail, $\boldsymbol{S_c}$ transforms from $h_d^{(i-1,j)}$ and from $h_d^{(i,j-1)}$. While $\boldsymbol{S_d}$ transforms from $h_c^{(i-1,j)}$ and from $h_c^{(i,j-1)}$.The nonlinear function $f(.)$ is ReLU $max(x, 0)$ in our implementation. The function $g(.)$ is the typical softmax and c_c and c_d are biases.

Since we adopt the quad-directions, the forward pass is similar to Equation 3 and in addition to the remaining directions beside the top-left sequence, i.e. top-right, bottom-left and bottom-right. To facilitate readability of the equation and to easily distinguish between all quad/four directions; we will use arrow notations to represent different directions. $\searrow$ refers to the top-left processing sequence, $\swarrow$ indicates the top-right sequence, $\nearrow$ is the bottom-left and finally $\nwarrow$ is the bottom-

right sequence. Now, the full model forward propagation pass in the RGB-RNN becomes:

$$
\begin{aligned}
h_c^{(i,j)}\searrow &= f(U_c^{\searrow}x_c^{(i,j)} + W_c^{\searrow}h_c^{(i-1,j)\searrow} + W_c^{\searrow}h_c^{(i,j-1)}\searrow \\
&\quad + \boldsymbol{S_c^{\searrow}}h_d^{(i-1,j)}\searrow + \boldsymbol{S_c^{\searrow}}h_d^{(i,j-1)}\searrow + b_c^{\searrow}) \\
h_c^{(i,j)}\swarrow &= f(U_c^{\swarrow}x_c^{(i,j)} + W_c^{\swarrow}h_c^{(i-1,j)\swarrow} + W_c^{\swarrow}h_c^{(i,j+1)}\swarrow \\
&\quad + \boldsymbol{S_c^{\swarrow}}h_d^{(i-1,j)}\swarrow + \boldsymbol{S_c^{\swarrow}}h_d^{(i,j+1)}\swarrow + b_c^{\swarrow}) \\
h_c^{(i,j)}\nearrow &= f(U_c^{\nearrow}x_c^{(i,j)} + W_c^{\nearrow}h_c^{(i+1,j)\nearrow} + W_c^{\nearrow}h_c^{(i,j-1)}\nearrow \\
&\quad + \boldsymbol{S_c^{\nearrow}}h_d^{(i+1,j)}\nearrow + \boldsymbol{S_c^{\nearrow}}h_d^{(i,j-1)}\nearrow + b_c^{\nearrow}) \\
h_c^{(i,j)}\nwarrow &= f(U_c^{\nwarrow}x_c^{(i,j)} + W_c^{\nwarrow}h_c^{(i+1,j)\nwarrow} + W_c^{\nwarrow}h_c^{(i,j+1)}\nwarrow \\
&\quad + \boldsymbol{S_c^{\nwarrow}}h_d^{(i+1,j)}\nwarrow + \boldsymbol{S_c^{\nwarrow}}h_d^{(i,j+1)}\nwarrow + b_c^{\nwarrow}) \\
z_c^{(i,j)} &= g(V_c^{\searrow}h_c^{(i,j)}\searrow + V_c^{\swarrow}h_c^{(i,j)}\swarrow + V_c^{\nearrow}h_c^{(i,j)}\nearrow \\
&\quad + V_c^{\nwarrow}h_c^{(i,j)}\nwarrow + c_c)
\end{aligned}
\tag{4}
$$

As mentioned, we use four quad-directional 2D-RNN to approximate each image. The arrows $\searrow, \swarrow, \nearrow, \nwarrow$ indicate the quad-directions (top-left, top-right, bottom-left and bottom-right), and the $h_c^{(i,j)}\searrow$, $h_c^{(i,j)}\swarrow$, $h_c^{(i,j)}\nearrow$ and $h_c^{(i,j)}\nwarrow$ are the corresponding four hidden planes. Each hidden plane has its own weight matrices besides the introduced transfer layers $\boldsymbol{S_c}$, e.g. $U_c^{\searrow}$ (input-hidden mapping), $W_c^{\searrow}$ (hidden-hidden mapping), $S_c^{\searrow}$ (the transfer layer from Depth-RNN hidden plane to RGB-RNN hidden plane), $V_c^{\searrow}$ (hidden-output mapping) and the bias term which is denoted by $b_c^{\searrow}$. Note that the connection between the two hidden planes $h_d\searrow$ and the corresponding $h_c\searrow$ is weighted by the corresponding weight matrix $S_c^{\searrow}$, which is leaned to extract shared patterns between the modalities.

Simultaneously, the forward pass in the Depth-RNN model is as follows:

$$
\begin{aligned}
h_d^{(i,j)}\searrow &= f(U_d^{\searrow}x_d^{(i,j)} + W_d^{\searrow}h_d^{(i-1,j)}\searrow + W_d^{\searrow}h_d^{(i,j-1)}\searrow \\
&\quad + \boldsymbol{S_d^{\searrow}}h_c^{(i-1,j)}\searrow + \boldsymbol{S_d^{\searrow}}h_c^{(i,j-1)}\searrow + b_d^{\searrow}) \\
h_d^{(i,j)}\swarrow &= f(U_d^{\swarrow}x_d^{(i,j)} + W_d^{\swarrow}h_d^{(i-1,j)}\swarrow + W_d^{\swarrow}h_d^{(i,j+1)}\swarrow \\
&\quad + \boldsymbol{S_d^{\swarrow}}h_c^{(i-1,j)}\swarrow + \boldsymbol{S_d^{\swarrow}}h_c^{(i,j+1)}\swarrow + b_d^{\swarrow}) \\
h_d^{(i,j)}\nearrow &= f(U_d^{\nearrow}x_d^{(i,j)} + W_d^{\nearrow}h_d^{(i+1,j)}\nearrow + W_d^{\nearrow}h_d^{(i,j-1)}\nearrow \\
&\quad + \boldsymbol{S_d^{\nearrow}}h_c^{(i+1,j)}\nearrow + \boldsymbol{S_d^{\nearrow}}h_c^{(i,j-1)}\nearrow + b_d^{\nearrow}) \\
h_d^{(i,j)}\nwarrow &= f(U_d^{\nwarrow}x_d^{(i,j)} + W_d^{\nwarrow}h_d^{(i+1,j)}\nwarrow + W_d^{\nwarrow}h_d^{(i,j+1)}\nwarrow \\
&\quad + \boldsymbol{S_d^{\nwarrow}}h_c^{(i+1,j)}\nwarrow + \boldsymbol{S_d^{\nwarrow}}h_c^{(i,j+1)}\nwarrow + b_d^{\nwarrow}) \\
z_d^{(i,j)} &= g(V_d^{\searrow}h_d^{(i,j)}\searrow + V_d^{\swarrow}h_d^{(i,j)}\swarrow + V_d^{\nearrow}h_d^{(i,j)}\nearrow \\
&\quad + V_d^{\nwarrow}h_d^{(i,j)}\nwarrow + c_d)
\end{aligned}
\tag{5}
$$

where $x_d^{(i,j)}$ is the feature vector of a certain patch at location (i,j) in the depth image. The $h_d^{(i,j)}\searrow$, $h_d^{(i,j)}\nearrow$, $h_d^{(i,j)}\swarrow$ and $h_d^{(i,j)}\nwarrow$ are the quad hidden planes in the Depth-RNN. Each hidden plane accompanies its own weight matrices $U_d^{\searrow}$ (input-hidden mapping), $W_d^{\searrow}$ (hidden-hidden mapping), $S_d^{\searrow}$ (transfer layer from RNN-RGB hidden plane to Depth-RNN hidden plane), $V_d^{\searrow}$ (hidden-output mapping) and its own bias term $b_d^{\searrow}$. The remaining terms in the quad planes are similar to the case of the first hidden plane. The function $f(.)$ is a nonlinear ReLU unit and the function $g(.)$ is the typical softmax and c_d is a bias term.

We can notice that the cross-connections through the transfer layers are applied in each processing direction (four sequences). By this method, the processing of a specific patch will rely on both previous hidden neighbors from its own modality in addition to the other modalities, thus learns more contextually-aware hidden representations of the RGB-D image patches from both modalities.

Our labeling task is a typical supervised classification problem. We aggregate the cross entropy losses from both modalities and calculate the loss for every patch. The error signal of an image is averaged across all the valid patches (those that are semantically labeled), which is mathematically formulated as the following:

$$
L = -\frac{1}{N}\sum_{n=1}^{N}\sum_{b=1}^{B}(log(z_c^n(b)\ )\delta(y^n = b) + log(z_d^n(b)\ )\delta(y^n = b))
\tag{6}
$$

where $\delta(\cdot)$ is the indicator function, N is the total number of patches and B is the number of semantic classes, y^n is the ground truth label for the RGB-D patch representation, which is the same as that of the center pixel, z_c^n and z_d^n are the class likelihood generated from both the RGB-RNN and the Depth-RNN for the patch representations $\mathbf{x}_c^n$ and $\mathbf{x}_d^n$ respectively, where they are B-dimensional vectors. Note that we ignore the contribution of unlabeled (invalid) patches in the loss calculation.

Optimization of the multimodal-RNNs model: To learn the multimodal RNNs parameters in Equations 4 and 5, we optimize the objective function in Equation 6 with a stochastic gradient-based method. Both RGB-RNN and Depth-RNN are optimized simultaneously using Back Propagation Through Time (BPTT) [12]. We unfold both networks in time and calculate their gradients which are back-propagated at each time step throughout both networks. This is similar to the typical multilayer feed-forward neural network propagation, but with the difference that the weights are shared across time steps defined by the architecture of the recurrent network. The whole model is differentiable and trained end-to-end. The propagation of the gradients through the RGB-RNN and Depth-RNN is simultaneous.

In more detail, we provide the first plane (top-left sequence) backward pass in the RGB-RNN. The backward pass derivations of the remaining sequences (the remaining three planes) can be easily inferred by following similar derivation strategy as we will explain next in the backward pass formulations for the first plane, but with considering the change of the sequence directions (top-right, bottom-left and bottom-right sequences). Similarity, the total derivations for the Depth-RNN are straightforward and exactly the same as for the RGB-RNN but with swapping the notations of c and d and vise versa.

In the RGB-RNN top-left sequence, we generate the gradients of the loss function in Equation 6 by deriving it with respect to the model internal parameters, i.e. top-left sequence weight matrices $U_c^{\searrow}$, $W_c^{\searrow}$, $\boldsymbol{S_c^{\searrow}}$, $V_c^{\searrow}$, $b_c^{\searrow}$ and c_c.

Notice that since the weight matrix $W_c^{\searrow}$ is shared between $h_c^{(i-1,j)}\searrow$ and $h_c^{(i,j-1)}\searrow$ and the transfer layer weight matrix $\boldsymbol{S_c^{\searrow}}$ is shared between $h_d^{(i-1,j)}\searrow$ and $h_d^{(i,j-1)}\searrow$, hence we will rewrite the forward pass as the following to further facilitate the understanding of the weight sharing concept and the backward pass:

$$
\begin{aligned}
\tilde{h_c} \searrow &= h_c^{(i-1,j)\searrow} + h_c^{(i,j-1)} \searrow \\
\tilde{h_d} \searrow &= h_d^{(i-1,j)\searrow} + h_d^{(i,j-1)} \searrow \\
h_c^{(i,j)} \searrow &= f(U_c^{\searrow} x_c^{(i,j)} + W_c^{\searrow} \tilde{h_c} \searrow + \boldsymbol{S_c}^{\searrow} \tilde{h_d} \searrow + b_c^{\searrow}) \\
z_c^{(i,j)} &= g(V_c^{\searrow} h_c^{(i,j)} \searrow + c_c)
\end{aligned}
\tag{7}
$$

Notice that the other remaining terms that present originally in Equation 4 (i.e. $V_c^{\swarrow} h_c^{(i,j)} \swarrow$, $V_c^{\nearrow} h_c^{(i,j)} \nearrow$, and $V_c^{\nwarrow} h_c^{(i,j)} \nwarrow$) do not engage any of the internal parameters of the top-left sequence, hence we omit them.

Before we formulate the backward pass, Figure 7 show an illustration of the forward and backward passes in the top-left plane sequence. Notice that the derivatives which are computed in the backward pass at each hidden state at some specific location (i, j) are processed in the **reverse order** of forward propagation sequence.

For better readability, we ignore the south-east arrow sign $\searrow$ in the following derivations for the top-left sequence.

Notice that now there are **two** types of error signals: **direct** one which is reachable directly through the loss ($\frac{\partial z_c^{(i,j)}}{\partial h_c^{(i,j)}}$) and **indirect** ones, i.e. the error signals that are coming from neighboring future states at $(i+1, j)$ and $(i, j+1)$ locations; ($\frac{\partial z_c^{(i+1,j)}}{\partial h_c^{(i+1,j)}} \frac{\partial h_c^{(i+1,j)}}{\partial h_c^{(i,j)}} + \frac{\partial z_c^{(i,j+1)}}{\partial h_c^{(i,j+1)}} \frac{\partial h_c^{(i,j+1)}}{\partial h_c^{(i,j)}}$).

Concretely, and given that the derivative of loss function L with respect to the Softmax output function g is $g'(.) = \frac{\partial L}{\partial z_c^{(i,j)}(.)} \frac{\partial z_c^{(i,j)}(.)}{\partial g}$, similarity $f'(.) = \frac{\partial h_c^{(i,j)}}{\partial f}$, hence at location (i, j) the backward pass of the RGB-RNN (derivations w.r.t. all internal parameters) is formulated as the following:

$$
\begin{aligned}
\Delta V_c &= g'(z_c^{(i,j)})(h_c^{(i,j)})^T \\
dh_c^{(i,j)} &= V_c^T g'(z_c^{(i,j)}) + W_c^T dh_c^{(i+1,j)} \circ f'(h_c^{(i+1,j)}) \\
&\quad + W_c^T dh_c^{(i,j+1)} \circ f'(h_c^{(i,j+1)}) \\
\Delta U_c &= dh_c^{(i,j)} \circ f'(h_c^{(i,j)})(x_c^{(i,j)})^T \\
\Delta W_c &= dh_c^{(i+1,j)} \circ f'(h_c^{(i+1,j)})(h_c^{(i,j)})^T \\
&\quad + dh_c^{(i,j+1)} \circ f'(h_c^{(i,j+1)})(h_c^{(i,j)})^T \\
\Delta \boldsymbol{S_c} &= dh_c^{(i+1,j)} \circ f'(h_c^{(i+1,j)})(h_d^{(i,j)})^T \\
&\quad + dh_c^{(i,j+1)} \circ f'(h_c^{(i,j+1)})(h_d^{(i,j)})^T \\
\Delta b_c &= dh_c^{(i,j)} \circ f'(h_c^{(i,j)}) \\
\Delta c_c &= g'(z_c^{(i,j)})
\end{aligned}
\tag{8}
$$

The term $dh_c^{(i,j)}$ is defined to allow the model to propagate local information internally. The $\circ$ sign is the Hadamard product (element-wise). We can notice through the term $dh_c^{(i,j)}$ that there are two sources of gradients, one is generated directly from the current hidden state and the other generated indirectly from the bottom and right locations (corresponding to the three terms: direct error: $V_c^T g'(z_c^{(i,j)})$, indirect error from neighbor bottom location: $W_c^T dh_c^{(i+1,j)} \circ f'(h_c^{(i+1,j)})$ and indirect from right neighbor location: $W_c^T dh_c^{(i,j+1)} \circ f'(h_c^{(i,j+1)})$). Notice that these two types of error signals (direct and indirect) are due to the weight sharing effect. It is also important to mention that in our derivations we show the error signals coming from the direct bottom and right neighbors, however, in real implementations the equations are recurrently applied to allow propagation of the errors recursively (recurrently) from all potential future neighbors until we reach out the current referenced patch at specific location.

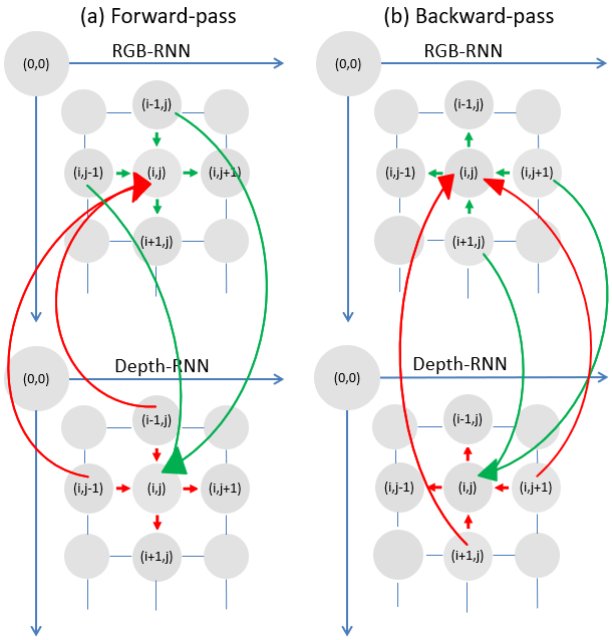


Fig. 7. An illustration of the forward and backward passes in the top-left plane sequence in both RGB-RNN and Depth-RNN. (Best viewed in color)

Similarly, the backward passes for the remaining three planes in the RGB-RNN are straightforward to be derived following Equation 8 but with paying attention to the directions of the processing sequence. Additionally, the backward pass in the Depth-RNN follows exactly similar derivations but with swapping the notations of c with d and vise versa. Notice that the derivation w.r.t c_c is the same in the other remaining three planes/sequences.

To recap, we adopt BPTT to train both basic quad-directional 2D-RNN (RGB-RNN and Depth-RNN). We use the cross entropy loss in our implementation and the errors are calculated by chain rule as we described previously in detail.

IV. Experiments and Results

A. Datasets

We evaluate our model on the benchmark dataset NYU versions 1 and 2 [55], [56]. The NYU V1 dataset contains 2284 RGB-D indoor scene images labeled with around 13 categories. The NYU V2 is also comprised of video sequences from indoor scenes as recorded by both the RGB and Depth cameras from the Microsoft Kinect. It contains 1449 densely labeled pairs of aligned RGB and depth images. We follow the settings in [56], where the first task is to predict the pixel

label out of four semantic classes: Ground, Furniture, Props and Structure. The second task is to predict the label out of 14 categories. The accuracy is calculated in terms of total pixel-wise, average class-wise and Intersection Over Union (average IOU) among the semantic classes for comparison.

B. CNNs and RNNs Training

We train our convolutional neural networks (CNNs) on both NYU version 1 and 2 images. We follow the network structure proposed by [54] without considering any spatial information channels (RGB locations). We also didn't perform any hybrid sampling as they mention in their implementation details. The CNN model consists mainly of three convolutional layers (where in between there are max-pooling and ReLU layers) and two last fully-connected layers followed by an B-way softmax loss layer. In more details, conv1($8\times8\times3$ or 4×32), max pool(2×2), conv2($6\times6\times32\times64$), max pool(2×2), conv3($5\times5\times64\times64$), max pool(2×2), FC1(1024×64) and FC2($64\times B$). The CNN models are trained on 65×65 patches associated with their centering pixel labels. Each image contains 4661 patches. Each CNN model produces a 64-dimensional vector per patch for each modality. We use these CNN features as the input of our RNN models.

In CNN training, we start by a learning rate of 0.001 and decrease it every 5 epochs by 10 (divided by 10). The momentum is initialized as 0.9 and remains the same throughout the training. We perform the typical normalization as a preprocessing step of images, we subtract the mean image and divide by the standard deviation. The CNN models nearly converge after 50 epochs (around 6 hours on NVIDIA TK40 GPU), however, we iterate the models in most cases until 100 epochs.

We train our multimodal RNNs using BPTT (back propagation through time). We adapt stochastic gradient descent (SGD) throughout our training. We initialize the learning rate as 0.00001 and decrease it by 0.01 after each epoch. The momentum is set to 0.9. The internal parameter dimension of the networks is set to 64. We apply gradient clipping [5], [46] and set the threshold value to $2*10^3$, however, even if the ReLU can potentially cause gradient explosion, it plays a critical role in RNN to mitigate the gradient vanishing problem. Thus, we mainly use ReLU in all of our RNN models. The other RNN model parameters are initialized either by randomly or zeros. Notice that we didn't consider any pixels that have zero labels in our training, we held them out. Our model converges much faster compared to the other competitive baselines. It converged at almost the similar speed of a single RNN model, with the benefit of processing multiple input modalities simultaneously.

C. Baselines

To show the effectiveness of our proposed model, we develop the following baselines for comparison alongside our proposed model:

- **CNN-RGB:** in this baseline, we train CNN based on RGB images (input is three RGB channels) for label prediction.
- **CNN-Depth:** we train CNN as in 'CNN-RGB' but using Depth images only (input is one Depth channel).
- **CNN-RGBD:** we train CNN as in 'CNN-RGB' with extra Depth images (input is four RGB-D channels) similar to the work presented in [8].
- **RNN-RGB:** in this baseline, we follow the structure of the quad-directional 2D-RNN proposed by [54]. We use the model in 'CNN-RGB' to extract RGB features. We only input the RGB features to train the RNN for label prediction.
- **RNN-Depth:** we extract Depth features using the trained model in 'CNN-Depth' and use these features to train quad 2D-RNN as in 'RNN-RGB'. We only input the Depth features to train the RNN for label prediction.
- **RNN-RGBD:** we use the trained model in 'CNN-RGBD' for feature extraction. We use these RGB-D features to train quad 2D-RNN similar to 'RNN-RGB' baseline to perform label prediction.
- **RNN-Classifiers-Combined:** or as we call it 'post-fusion', here we train RGB-RNN and Depth-RNN for label prediction and combine their classification scores on the classifier level. We use the trained models in 'CNN-RGB' and 'CNN-Depth' for feature extraction.
- **RNN-Features-Combined:** or 'pre-fusion', here we use both trained models in 'CNN-RGB' and 'CNN-Depth' for feature extraction. We concatenate both RGB and Depth features (to form higher dimensional feature vectors) and train one quad 2D-RNN similar to 'RNN-RGB' baseline to perform label prediction.
- **RNN-Hiddens-Combined:** or 'middle-fusion', we fuse the hidden representations of both RNNs just before classification and train them jointly.
- **Multimodal-RNNs-Ours** in this sitting we implement our proposed multimodal RNNs structure. Here, we have two internal RNN models one is responsible for processing the RGB features and the other is responsible for processing the Depth features. Both models are optimized simultaneously using BPTT where we combine their classification scores to finally obtain the label map per RGB-D image.
- **Multimodal-RNNs-Ours-Multiscale:** similar to our 'Multimodal-RNNs-Ours', but in this setting we applied our proposed structure on multiscale convolutional features as proposed by [14]. We follow similar training sittings to train multiscale CNN models on different image sizes, then we use them to extract multiscale features. We concatenate these features together to form our final input patch representation to our multimodal-RNNs.

  We also compare our results with other state-of-the-art methods.

D. Results

Given the input RGB images and their corresponding Depth images, we divide each image into non-overlapping patches of size 65×65. The extracted local CNN features are used as the input of our multimodal RNNs. The RGB-RNN and

Algorithm	Pixel Acc	Class Acc	IOU
CNN-RGB	69.21%	56.23%	45.17%
CNN-Depth	58.56%	32.82%	22.21%
CNN-RGBD	69.17%	57.83%	45.24%
RNN-RGB	71.38%	64.43%	52.80%
RNN-Depth	60.67%	46.88%	28.24%
RNN-RGBD	71.09%	67.01%	57.87%
RNN-Features-Combined	72.94%	69.06%	60.54%
RNN-Hiddens-Combined	71.50%	67.00%	55.59%
RNN-Classifiers-Combined	71.28%	66.37%	54.79%
Multimodal-RNNs-Ours	74.68%	72.54%	62.53%
Multimodal-RNNs-Ours-Multiscale	**78.89%**	**75.73%**	**65.70%**
Wang et al. [64]	-	61.71%	-
gradient KDES [51]	-	51.84%	-
color KDES [51]	-	53.27%	-
spin/surface normal KDES [51]	-	40.28%	-
depth gradient KDES [51]	-	53.56%	-
Silberman et al [55]	-	53.00%	-
Pei et al. [47]	-	50.50%	-
Ren et al. [51]	-	71.40%	-
Eigen et al. [11]	75.40%	66.90%	-

TABLE I
RESULTS ON **NYU V1** DATASET [55]. THE PERFORMANCE OF ALL BASELINES AND OTHER STATE-OF-THE-ART METHODS IS MEASURED ON TOTAL PIXEL-WISE, AVERAGE CLASS-WISE ACCURACY AND AVERAGE IOU (INTERSECTION OVER UNION) AMONG 13 SEMANTIC CLASSES. THE HIGHER THE BETTER.

Algorithm	Pixel Acc	Class Acc	IOU
CNN-RGB	65.55%	62.07%	45.42%
CNN-Depth	69.61%	65.96%	49.13%
CNN-RGBD	71.60%	69.89%	54.19%
RNN-RGB	68.14%	65.96%	51.05%
RNN-Depth	70.95%	67.76%	52.21%
RNN-RGBD	74.18%	69.99%	56.63%
RNN-Features-Combined	74.36%	72.80%	60.08%
RNN-Hiddens-Combined	73.70%	71.10%	56.00%
RNN-Classifiers-Combined	73.39%	70.64%	55.34%
Multimodal-RNNs-Ours	75.74%	75.01%	62.10%
Multimodal-RNNs-Ours-Multiscale	78.60%	76.69%	**65.09%**
Wang et al. [64]	-	65.30%	-
Couprie et al. [8]	64.50%	63.50%	-
Stuckler et al. [60]	70.90%	67.00%	-
Khan et al. [35]	69.20%	65.60%	-
Mueller et al. [44]	72.30%	71.90%	-
Gupta et al. [23]	78.00%	-	64.00%
Cadena and Kosecka [7]	-	64.10%	-
Eigen et al. [11]	**83.20%**	**82.00%**	-

TABLE II
RESULTS ON **NYU V2** DATASET [56]. THE PERFORMANCE OF ALL BASELINE AND OTHER STATE-OF-THE-ART METHODS IS MEASURED ON TOTAL PIXEL-WISE, AVERAGE CLASS-WISE ACCURACY AND AVERAGE IOU (INTERSECTION OVER UNION) AMONG 4 SEMANTIC CLASSES: GROUND, FURNITURE, PROPS AND STRUCTURE. THE HIGHER THE BETTER.

the Depth-RNN models process RGB and Depth local patch features respectively.

Results on NYU V1 - 13 categories: In this dataset the task is to predict pixel labels out of 12 categories plus an unknown category. Table I shows the results of our baselines alongside the multimodal RNNs model and other state-of-the-art methods. RNN models outperform CNN baselines by a large margin. And our multimodal RNNs further improve the accuracy over the RNN baselines. Our model achieves comparable results with the other state-of-the-art methods. The accuracy comparison with all baselines shows the effectiveness of our method with the proposed transfer layers. Figure 8 show some qualitative results generated by our method and the most competitive baseline method 'RNN-Features-Combined'. Our multi-modal-RNNs model can correctly classify many mis-classifications results compared to the baseline in most cases.

Results on NYU V2 - 4 categories: For example, Eigen et al. [11] achieve higher results than our model on this task, while we outperform their model on NYU V1 task. Their multi-scale CNN structure is also well designed to address the problem. However, their network appears to be much more complex than ours as they have higher number of computational operations (way more convolutions). But we still believe our work can be potentially orthogonal and complementary to their method if it is trained end-to-end with their method.

Likewise, Couprie et al. [56], use many types of features, including SIFT features, histograms of surface normals, 2D and 3D bounding box dimensions, color histograms, relative depth and their support features. In our model, we only use the transfer layers alongside the basic quad 2D-RNN structures, and can achieve much better performance. Figure 9 also show some qualitative results generated by our method and the most competitive baseline method 'RNN-Features-Combined'. Our multi-modal-RNNs model can correctly classify many mis-classifications results compared to the baseline in most cases.

Results on NYU V2 - 14 categories: On the NYU V2, we evaluate our model to label image pixels with one of 13 categories plus an unknown category. We show the comparison in this task with various baselines and state-of-the-art methods as shown in Table III. This is the most competitive benchmark presented on this dataset. Notice that the work of Cadena and Kosecka [7] used many RGB image and 3D features and formulate the problem in the CRF framework. Meanwhile the work of Wang et al. [64] adapted an existing unsupervised feature learning technique to directly learn features. They stack their basic learning structure to learn hierarchical features. They combined their higher-level features with low-level features and train linear SVM classifiers to perform labeling. Compared to these methods, our model is much simpler and achieve better performance.

Figure 10 also shows some qualitative results generated by our method and the most competitive baseline method 'RNN-Features-Combined'. Our multimodal-RNNs model can correctly classify many mis-classifications results compared to the baseline in most cases. We also show our per class accuracy on this sitting, as shown in Figure 11. We can notice that the improvement gain of our multimodal RNNs over CNN-RGBD and RNN-RGBD models is significant. This is an evidence that our model can effectively learn powerful context-aware and multimodel features.

We also study the effect of increasing the dimensionality of the internal hidden layer on single RNN performance. We notice that RNN performs almost the same but slightly better when the hidden layer dimension increases, while it becomes extremely slow and takes a lot of time to converge. Thus, we choose the dimension of the hidden layer to be 64. All

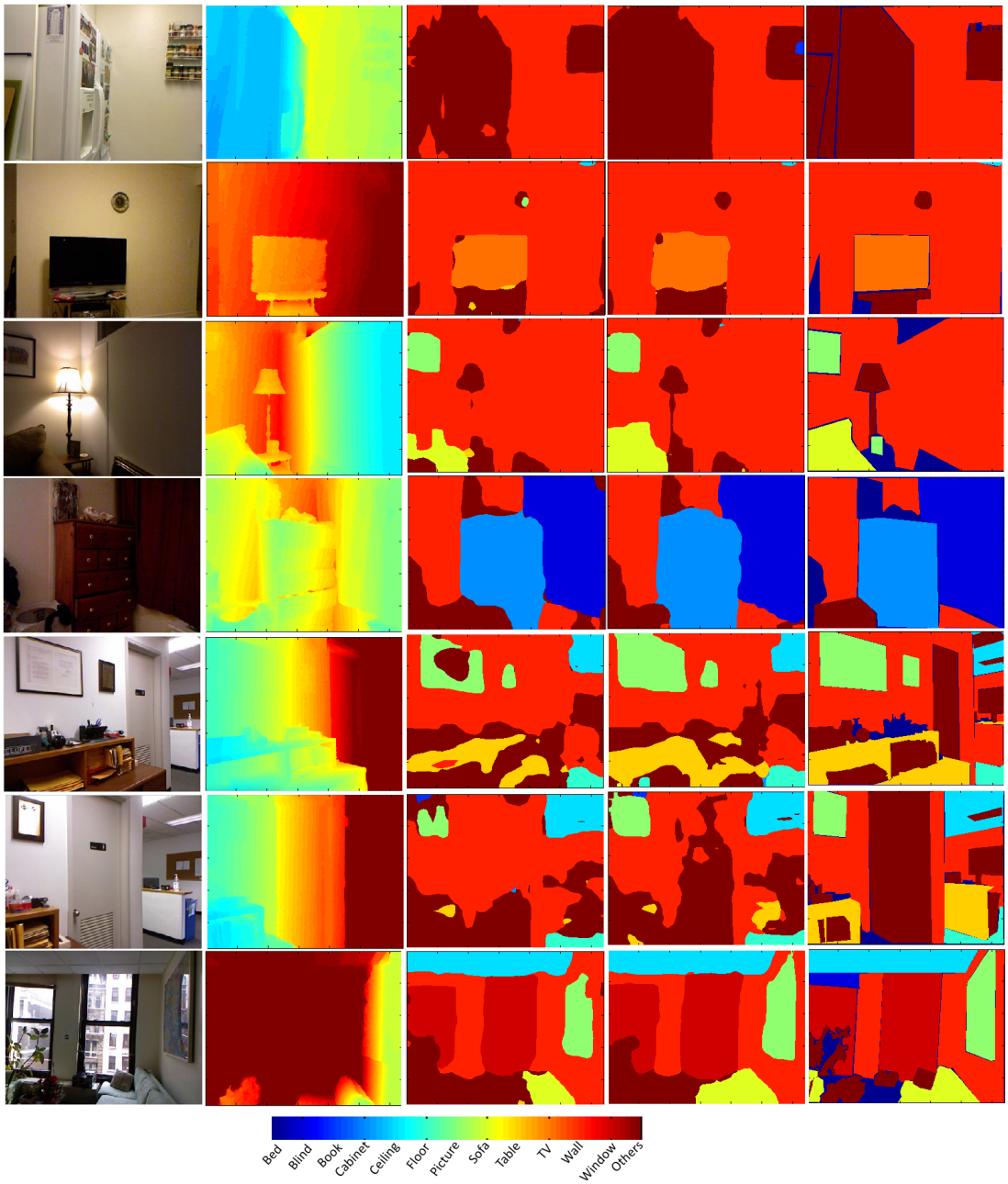


Fig. 8. 7 example results from NYU-V1 (13 semantic categories). Columns 1st and 2nd: RGB color images and their corresponding Depth images. Column 3rd: the prediction results of our most competitive baseline 'RNN-Features-Combined' on the NYU V2 four semantic categories. Column 4th: our multimodal-RNNs prediction results. Column 5th:ground truth. Our model is able to correctly classify many miss-classified pixels compared with our baseline. (Best viewed in color)

RNN models in our baselines can achieve good performance and can converge in reasonable amount of time. Figure 12 shows the relationship between a single RNN hidden layer dimensionality versus the accuracy (in terms of global pixel-wise) trained on the NYU V2 to predict 4 semantic categories.

Examining our multimodal RNNs with other CNN features - VGG Features on NYU V2 - 14 We also examine our proposed multimodal RNNs while replacing the input CNN features by the extracted features from the VGG-16 pre-trained model (extracted from Conv5-3 layer) [57]. We focused mainly on the most competitive task among our addressed tasks, i.e. classifying the 14 classes in NYU V2. The purpose of these experiments is to validate whether our fusion structure is a network-independent model, and concretely orthogonal to other CNN networks. In other words, replacing the CNN models with more powerful network like VGG [57], ResNet [34], FCNs [33], Dilated Networks [71] and others can boost the overall performance while the relative improvement of our proposed cross-connectivity fusion is maintained. Throughout all of our experiments, we observe that **replacing our CNN features with VGG features result in a constant overall increase in the accuracies in all of our RNN models**

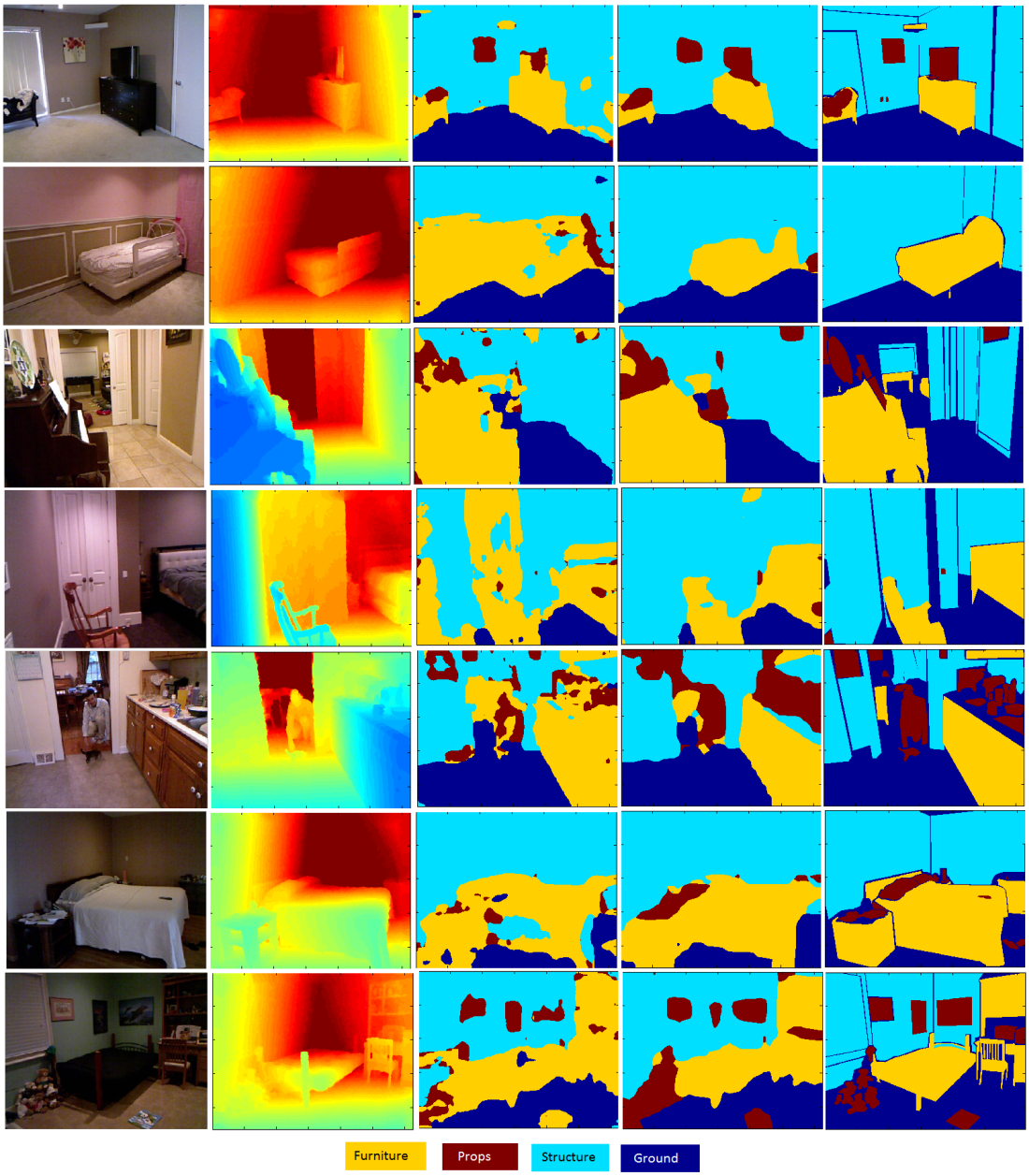


Fig. 9. 7 example results. Columns 1st and 2nd: RGB color images and their corresponding Depth images. Column 3rd: the prediction results of our most competitive baseline 'RNN-Features-Combined' on the NYU V2 four semantic categories. Column 4th: our multimodal-RNNs prediction results. Column 5th:ground truth. Our model is able to correctly classify many miss-classified pixels compared with our baseline. (Best viewed in color)

(including the baselines) of around 5% higher in terms of IOU (most competitive metric).

Notice that in this paper, we didn't perform joint training between the CNN layers and our multimodal RNN layers. In contrast, we perform stage training; we first train the CNN for feature extraction and then train our RNN model for final local classification. This makes the performance comparisons on the NYU-V2 tasks between our model and other state-of-the-art models not fair (many works perform end-to-end joint training of various CNN or multi-scale CNN and CRF-based methods and even with RNN/LSTM). Notice also that end-to-end training with CNN can allow training of efficient deconvolution layers to upsample the output feature maps in order to restore their original resolution, while we use simple bilinear interpolation to upsample our final output map produced by the RNNs. Thus, in order to fairly examine the full performance of our multimodal RNN model combined with other types of recent CNN models like ResNets [34], FCNs [33] and Dilated Networks [71]; a joint end-to-end training is required. In this paper, we didn't perform this joint training between the CNN and the RNN models as it is not our main contribution, but we consider it as a very good future work.

E. Observations

From Table I, III and II we have the following observations:

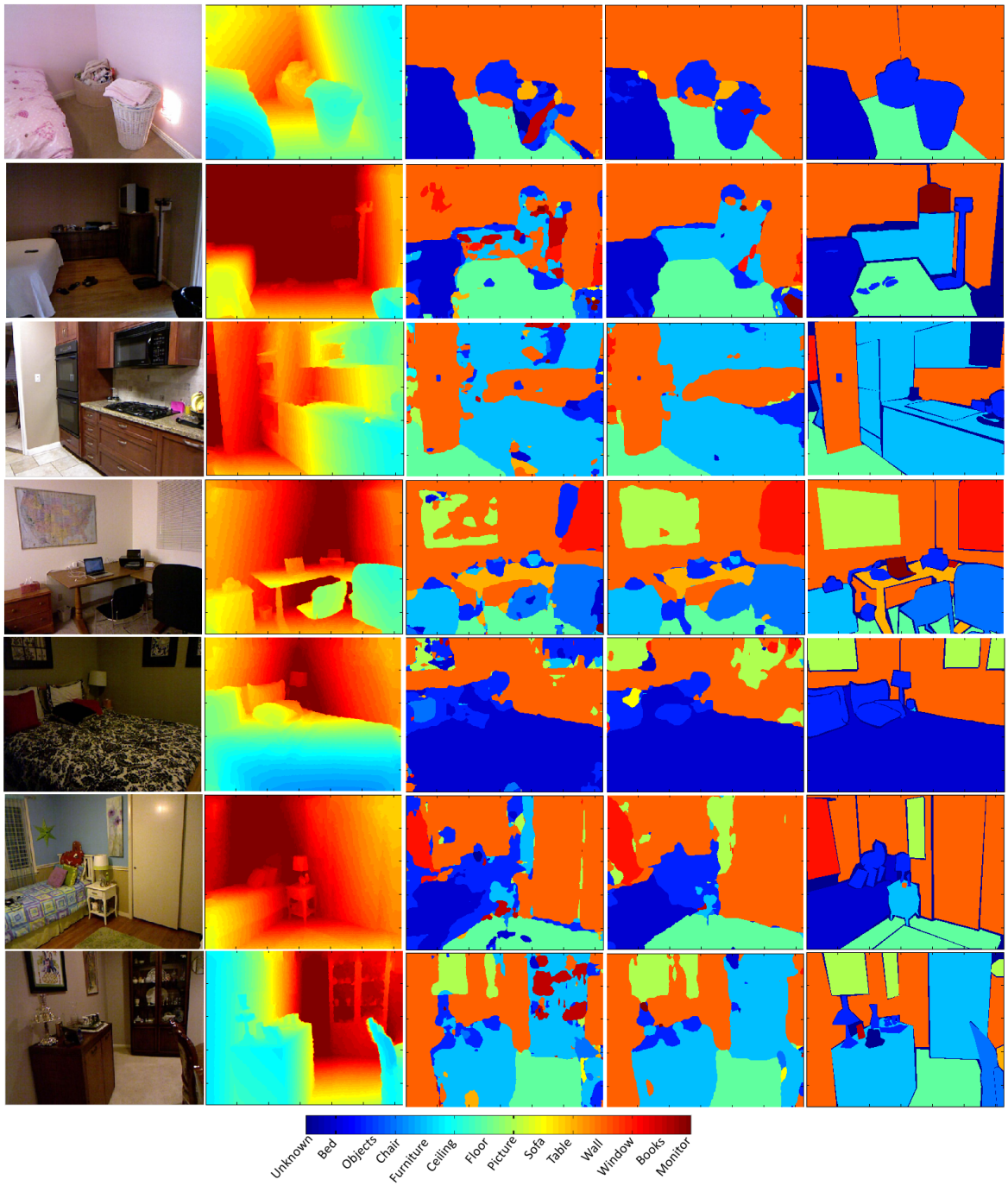


Fig. 10. 7 example results from NYU-V2 (14 semantic categories). Columns 1st and 2nd: RGB color images and their corresponding Depth images. Column 3rd: the prediction results of our most competitive baseline 'RNN-Features-Combined' on the NYU V2 four semantic categories. Column 4th: our multimodal-RNNs prediction results. Column 5th:ground truth. Our model is able to correctly classify many miss-classified pixels compared with our baseline. (Best viewed in color)

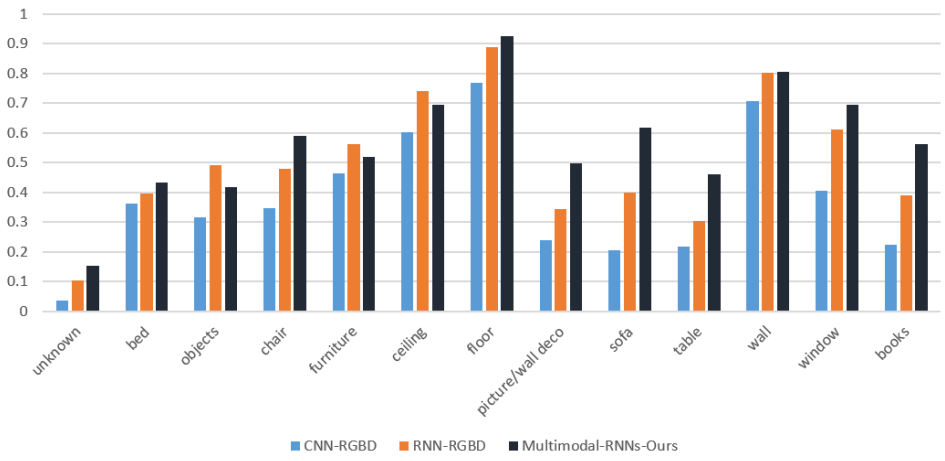


Fig. 11. Per class accuracy comparison between our multimodal RNNs, baselines CNN-RGBD and RNN-RGBD on NYU V2 dataset [56](14 semantic categories).

Algorithm	Pixel Acc	Class Acc	IOU
CNN-RGB	50.84%	35.37%	21.72%
CNN-Depth	56.08%	36.82%	23.41%
CNN-RGBD	54.40%	35.48%	22.07%
RNN-RGB	56.22%	42.41%	29.19%
RNN-Depth	62.27%	45.99%	33.62%
RNN-RGBD	61.95%	48.15%	35.74%
RNN-Features-Combined	64.30%	51.54%	39.31%
RNN-Hiddens-Combined	64.50%	51.70%	37.30%
RNN-Classifiers-Combined	64.44%	50.27%	37.42%
Multimodal-RNNs-Ours	66.23%	53.06%	40.59%
Multimodal-RNNs-Ours-Multiscale	**67.90%**	**54.67%**	**43.27%**
Wang et al. [64]	-	42.20%	-
Couprie et al. [8]	52.40%	36.20%	-
Hermans et al. [26]	54.20%	48.00%	-
Khan et al. [35]	58.30%	45.10%	-

TABLE III
RESULTS ON **NYU V2** DATASET [56]. THE PERFORMANCE OF ALL BASELINE AND OTHER STATE-OF-THE-ART METHODS IS MEASURED ON TOTAL PIXEL-WISE, AVERAGE CLASS-WISE AND AVERAGE IOU (INTERSECTION OVER UNION) AMONG 14 SEMANTIC CLASSES. THE HIGHER THE BETTER.

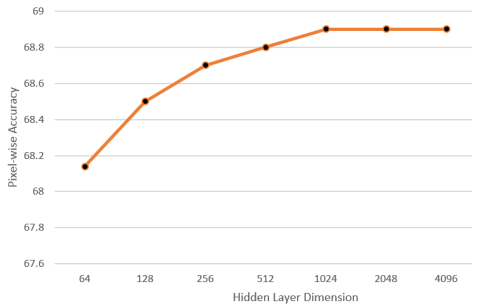


Fig. 12. The relationship between the hidden layer dimensions and the pixel-wise accuracy in a single RNN model trained on the NYU V2 dataset [56]

Modeling contextual dependencies between patches using RNNs helps: The improvement gain achieved by our RNN models over the CNN models is significant. CNN features are locally learned when performing the convolutions and thus fail to encode long-range contextual information. RNNs are powerful on modeling short and long range dependencies between patches within the image and can learn context-aware features effectively.

Sharing information between RNNs helps: We design the baseline 'RNN-Classifiers-Combined' that combines two RNN models on the classifier level. Our multimodal RNNs model outperforms this baseline as it benefits from the shared contextual information extracted through the transfer layers.

Learning transfer layers to connect RNNs helps: The baselines 'RNN-RGBD' and 'RNN-Features-Combined' are designed to mix RGB and Depth data modalities together before learning the features. Our model outperforms these baselines because it has an assigned single RNN for each modality to retain the modality-specific information. Plus, the transfer layers are learned to adaptively extracts only the relevant multimodal shared information.

V. CONCLUSION

This paper presents a new method for RGB-D scene semantic segmentation. We introduce information transfer layers between two quad-directional 2D-RNNs. Transfer layers extract relevant contextual information across the modalities and help each modality to learn context-aware features that can capture shared information. In our future work, we will evaluate the effectiveness and the scalability of the transfer layers on more modalities (modalities > 2).

ACKNOWLEDGMENT

The authors would like to thank NVIDIA Corporation for their donation of Tesla K40 GPUs used in this research at the

Rapid-Rich Object Search Lab. This research was carried out at both the Advanced Digital Sciences Center (ADSC), Illinois at Singapore Pt Ltd, Singapore, and at the Rapid-Rich Object Search (ROSE) Lab at the Nanyang Technological University, Singapore. This work is supported by the research grant for ADSC from A*STAR. The ROSE Lab is supported by the National Research Foundation, Singapore, under its Interactive & Digital Media (IDM) Strategic Research Programme.

REFERENCES

[1] A. H. Abdulnabi, B. Shuai, S. Winkler, and G. Wang. Episodic camn: Contextual attention-based memory networks with iterative feedback for scene labeling. In *CVPR*, 2017. 1

[2] A. H. Abdulnabi, G. Wang, J. Lu, and K. Jia. Multi-task cnn model for attribute prediction. *TMM*, 2015. 1

[3] A. H. Abdulnabi, S. Winkler, and G. Wang. Beyond forward shortcuts: Fully convolutional master-slave networks (msnets) with backward skip connections for semantic segmentation. *arXiv preprint arXiv:1707.05537*, 2017. 1

[4] G. Andrew, R. Arora, J. Bilmes, and K. Livescu. Deep canonical correlation analysis. In *ICML*, 2013. 2, 3

[5] Y. Bengio, N. Boulanger-Lewandowski, and R. Pascanu. Advances in optimizing recurrent networks. In *ICASSP*, 2013. 8

[6] W. Byeon, T. M. Breuel, F. Raue, and M. Liwicki. Scene labeling with lstm recurrent neural networks. In *CVPR*, 2015. 2

[7] C. Cadena and J. Kosecka. Semantic parsing for priming object detection in rgb-d scenes. In *Workshop on Semantic Perception, Mapping and Exploration*, pages 582–597, 2013. 9

[8] C. Couprie, C. Farabet, L. Najman, and Y. LeCun. Indoor semantic segmentation using depth information. *arXiv preprint arXiv:1301.3572*, 2013. 2, 8, 9, 13

[9] N. Dalal, B. Triggs, and C. Schmid. Human detection using oriented histograms of flow and appearance. In *ECCV*, 2006. 3

[10] C. Ding and D. Tao. Robust face recognition via multimodal deep face representation. In *IEEE Transactions on Multimedia*, volume 17, pages 2049–2058, Nov 2015. 1

[11] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015. 2, 9

[12] J. L. Elman. Finding structure in time. In *Cognitive Science*, 1990. 4, 6

[13] A. Ess, B. Leibe, and L. V. Gool. Depth and appearance for mobile scene analysis. In *ICCV*, pages 1–8, 2007. 3

[14] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Scene parsing with multiscale feature learning, purity trees, and optimal covers. In *ICML*, 2012. 2, 8

[15] J. Fu, H. Zheng, and T. Mei. Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In *CVPR*, 2017. 1

[16] D. Gavrila and S. Munder. Multi-cue pedestrian detection and tracking from a moving vehicle. In *IJCV*, 2007. 3

[17] A. Graves, S. Fernandez, and J. Schmidhuber. Bidirectional LSTM networks for improved phoneme classification and recognition. In *ICANN*, pages 799–804, 2005. 3

[18] A. Graves, S. Fernandez, and J. Schmidhuber. Multi-dimensional recurrent neural networks. In *ICANN*, 2007. 2, 3, 4

[19] A. Graves, A. Mohamed, and G. E. Hinton. Speech recognition with deep recurrent neural networks. In *ICASSP*, 2013. 2, 3, 4

[20] A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. In *Neural Networks*, 2005. 3

[21] A. Graves and J. Schmidhuber. Offline handwriting recognition with multidimensional recurrent neural networks. In *NIPS*, 2008. 2, 3

[22] S. Gupta, P. Arbelaez, R. Girshick, and J. Malik. Indoor scene understanding with rgb-d images: Bottom-up segmentation, object detection and semantic segmentation. In *IJCV*, 2015. 2

[23] S. Gupta, P. Arbelaez, and J. Malik. Perceptual organization and recognition of indoor scenes from rgb-d images. In *CVPR*, 2013. 9

[24] J. He and R. Lawrence. A graph-based framework for multi-task multi-view learning. In *ICML*, pages 25–32, 2011. 3

[25] X. He, R. Zemel, and M. Carreira-Perpindn. Multiscale conditional random fields for image labeling. In *CVPR*, 2004. 2

[26] A. Hermans, G. Floros, and B. Leibe. Dense 3d semantic mapping of indoor scenes from rgb-d images. In *ICRA*, 2014. 13

[27] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In *A Field Guide to Dynamical Recurrent Neural Networks, IEEE Press*, 2001. 3

[28] S. Hochreiter and J. Schmidhuber. Long short-term memory. In *MIT Press*, 1997. 3

[29] S. C. H. Hoi and M. R. Lyu. A multimodal and multilevel ranking scheme for large-scale video retrieval. 10(4):607–619, June 2008. 1

[30] Z. Hong, X. Mei, D. Prokhorov, and D. Tao. Tracking via robust multi-task multi-view joint sparse representation. In *ICCV*, 2013. 3

[31] H. Hotelling. Relations between two sets of variates. In *Biometrika*, 1936. 3

[32] O. Irsoy and C. Cardie. Opinion mining with deep recurrent neural networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2014. 2, 3, 4

[33] E. S. Jonathan Long and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 640–651, 2015. 10, 11

[34] S. R. Kaiming He, Xiangyu Zhang and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 10, 11

[35] S. H. Khan, M. Bennamoun, F. Sohel, and R. Togneri. Geometry driven semantic labeling of indoor scenes. In *ECCV*, 2014. 9, 13

[36] S. Kim and E. P. Xing. Tree-guided group lasso for multi-task regression with structured sparsity. In *ICML*, 2010. 3

[37] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012. 2

[38] H. Lee, C. Ekanadham, and A. Y. NG. Sparse deep belief net model for visual area v2. In *NIPS*, 2007. 2, 3

[39] B. Leibe, N. Cornelis, K. Cornelis, and L. V. Gool. Dynamic 3d scene analysis from a moving vehicle. In *CVPR*, pages 1–8, 2007. 3

[40] I. Lenz, H. Lee, and A. Saxena. Deep learning for detecting robotic grasps. In *IJRR*, 2014. 3

[41] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan. Perceptual generative adversarial networks for small object detection. 2017. 1

[42] W. Liu, Y. Zhang, S. Tang, J. Tang, R. Hong, and J. Li. Accurate estimation of human body orientation from rgb-d sensors. In *IEEE Trans. Cybernetics 43(5) 1442-1452*, 2013. 2

[43] S. S. Mukherjee and N. M. Robertson. Deep head pose: Gaze-direction estimation in multimodal video. In *IEEE Transactions on Multimedia*, volume 17, pages 2094–2107, Nov 2015. 1

[44] A. C. Muller and S. Behnke. Learning depth-sensitive conditional random fields for semantic segmentation of rgb-d images. In *ICRA*, 2014. 9

[45] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng. Multimodal deep learning. In *ICML*, 2011. 2, 3

[46] R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *arXiv preprint arXiv:1211.5063*, 2012. 8

[47] D. Pei, H. Liu, Y. Liu, and F. Sun. Unsupervised multimodal feature learning for semantic image segmentation. *IJCNN*, pages 1–6, 2013. 9

[48] P. H. O. Pinheiro and R. Collobert. Recurrent convolutional neural networks for scene labeling. In *ICML*, 2014. 2

[49] H. Pirsiavash and D. Ramanan. Detecting activities of daily living in first-person camera views. In *CVPR*, 2012. 1

[50] M. Rapus, S. Munder, G. Baratoff, and J. Denzler. Pedestrian recognition using combined low-resolution depth and intensity images. In *Intelligent Vehicles Symposium*, 2008. 3

[51] X. Ren, L. Bo, and D. Fox. Rgb-(d) scene labeling: Features and algorithms. In *CVPR*, 2012. 2, 9

[52] M. Rohrbach, M. Enzweiler, and D. M. Gavrila. High-Level fusion of depth and intensity for pedestrian classification. In *DAGM 2009: Pattern Recognition*, pages 101–110, 2009. 3

[53] J. H. Saurabh Gupta and J. Malik. Cross modal distillation for supervision transfer. In *CVPR*, 2016. 3

[54] B. Shuai, Z. Zuo, and G. Wang. Quaddirectional 2d-recurrent neural networks for image labeling. In *Signal Processing Letters*, 2015. 2, 4, 8

[55] N. Silberman and R. Fergus. Indoor scene segmentation using a structured light sensor. In *ICCV Workshops*, 2011. 1, 2, 7, 9

[56] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 1, 2, 7, 9, 13

[57] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556v6*, 2015. 10

[58] K. Sohn, W. Shang, and H. Lee. Improved multimodal deep learning with variation of information. In *NIPS*, 2014. 2, 3

[59] N. Srivastava and R. Salakhutdinov. Multimodal learning with deep boltzmann machines. In *NIPS*, pages 2949–2980, 2012. 2, 3

[60] J. Stuckler, B. Waldvogel, H. Schulz, and S. Behnke. Dense real-time mapping of object-class semantics from rgb-d video. In *J. Real-Time Image Processing*, 2014. 9

[61] S. Tang, X. Wang, X. Lv, T. Han, J. Keller, Z. He, M. Skubic, and S. Lao. Histogram of oriented normal vectors for object recognition with a depth sensor. In *ACCV*, 2013. 2

[62] W. van der Mark and D. M. Gavrila. Real-time dense stereo for intelligent vehicles. In *IEEE Transactions on Intelligent Transportation Systemsn*, 2006. 3

[63] A. Wang, J. Lu, J. Cai, T. J. Cham, and G. Wang. Large-margin multi-modal deep learning for rgb-d object recognition. In *IEEE Transactions on Multimedia*, volume 17, pages 1887–1898, Nov 2015. 1

[64] A. Wang, J. Lu, G. Wang, J. Cai, and T. Cham. Multi-modal unsupervised feature learning for rgb-d scene labeling. In *ECCV*, 2014. 2, 9, 13

[65] D. Wang, P. Cui, M. Ou, and W. Zhu. Learning compact hash codes for multimodal representations using orthogonal deep structure. In *IEEE Transactions on Multimedia*, volume 17, pages 1404–1416, Sept 2015. 1

[66] W. Wang and Z. hua Zhou. A new analysis of co-training. In *ICML*, 2010. 2, 3

[67] Z. Wang, S. Chen, and T. Sun. Multik-mhks: A novel multiple kernel learning algorithm. In *TPAMI*, 2008. 2, 3

[68] H. Wolfgang and S. Leopold. Canonical correlation analysis. In *Applied Multivariate Statistical Analysis*, 2007. 3

[69] C. Xu, D. Tao, and C. Xu. A survey on multi-view learning. *arXiv preprint arxiv:1304.5634*, 2013. 2, 3

[70] J. G. Yong Jae Lee and K. Grauman. Discovering important people and objects for egocentric video summarization. In *CVPR*, 2012. 1

[71] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, pages 1–9, 2016. 10, 11

[72] Z. Zuo, B. Shuai, G. Wang, X. Liu, X. Wang, B. Wang, and Y. Chen. Convolutional recurrent neural networks:learning spatial dependencies for image representation. In *CVPR Workshops*, 2015. 2, 4