# Learning with Side Information through Modality Hallucination

Judy Hoffman Saurabh Gupta Trevor Darrell
EECS Department, UC Berkeley
{jhoffman, sgupta, trevor}@eecs.berkeley.edu

## Abstract

*We present a modality hallucination architecture for training an RGB object detection model which incorporates depth side information at training time. Our convolutional hallucination network learns a new and complementary RGB image representation which is taught to mimic convolutional mid-level features from a depth network. At test time images are processed jointly through the RGB and hallucination networks to produce improved detection performance. Thus, our method transfers information commonly extracted from depth training data to a network which can extract that information from the RGB counterpart. We present results on the standard NYUDv2 dataset and report improvement on the RGB detection task.*

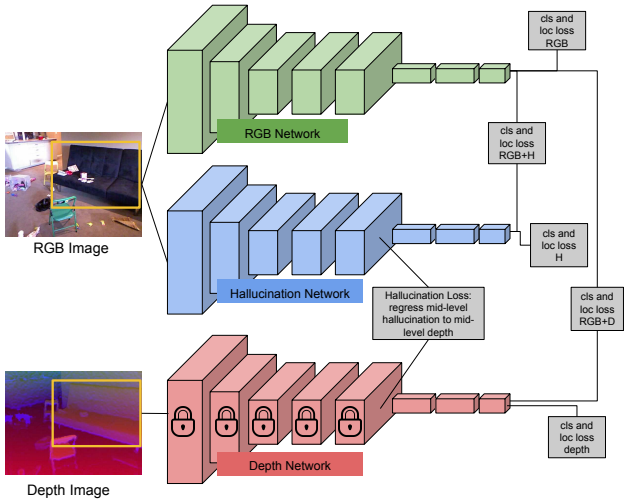


Figure 1: Training our modality hallucination architecture. We learn a multimodal Fast R-CNN [10] convolutional network for object detection. Our hallucination branch is trained to take an RGB input image and mimic the depth mid-level activations. The whole architecture is jointly trained with the bounding box labels and the standard softmax cross-entropy loss.

## 1. Introduction

RGB and depth images offer different and often complementary information. In fact, recent work has shown that the two image modalities can be used simultaneously to produce better recognition models than either modality alone [15, 36, 35]. While RGB image capturing devices are pervasive, depth capturing devices are much less prevalent. This means that many recognition models will need to perform well on RGB images alone as input. We present an algorithm which uses available paired RGB-d training data to learn to hallucinate mid-level convolutional features from an RGB image. We demonstrate that through our approach we produce a novel convolutional network model which operates over only the single RGB modality input, but outperforms the standard network which only trains on RGB images. Thus, our method transfers information commonly extracted from depth training data to a network which can extract that information from the RGB counterpart.

Convolutional networks (ConvNets) have produced tremendous success on visual recognition tasks, from classification [21, 28, 31], to detection [11, 25], to semantic segmentation [24, 39]. The standard approach for training these networks is to initialize the network parameters using a large labeled image corpra (ex: ImageNet [6]) and then fine-tune using the smaller target labeled data sources. While this strategy has been proven to be very effective, it offers only one technique for learning representations for recognition and due to the large parameter space of the network, runs the risk of overfitting to the nuances of the small RGB dataset.

We propose an additional representation learning algorithm which incorporates side information in the form of an additional image modality at training time to produce a more informed test time single modality model. We accomplish this by directly learning a modality hallucination network which optimizes over the standard class and bounding box localization losses while being guided by an additional hallucination loss which regresses the hallucination features

to the auxiliary modality features.

Due to its practicality, we consider the case of producing an RGB detector using some paired RGB-D data at training time. In doing so, we produce a final model which at test time only sees an RGB image, but is able to extract both the image features learned through finetuning with standard supervised losses as well as the hallucinated features which have been trained to mirror those features you would extract if a depth image were present. We demonstrate that our RGB with hallucination detector model outperforms the state-of-the-art RGB model on the NYUD2 dataset.

## 2. Related Work

We use depth side information at training time to transfer information through a new representation to our test time RGB model.

**RGB-D Detection.** Depth and RGB modalities often offer complementary information. Prior work has made use of this fact by producing detectors which take as input paired RGB and depth modalities to improve detection performance over the RGB only model. Many of these methods do so by introducing new depth representations [19, 29, 32, 38, 36], most recently by adding an additional depth network representation into a convolutional network architecture [15, 14, 35]. Our work is inspired by these approaches, which successfully learn complementary depth feature representations. We learn such representations at training time and learn to transfer information from the depth representation to an RGB only model through modality hallucination.

**Transfer Learning.** Our work is related to transfer learning and domain adaptation which learns to share information from one task to another. Classifical approaches consider learning to adapt across distributions, through some combination of parameter updates [2, 7, 18] and transformation learning [22, 12]. Christoudias *et al*. [5] learned a mapping to hallucinate a missing modality at training time, but was only shown with weak recognition models. Along these lines a transformation learning approach was recently introduced to use depth information at training time to inform RGB test time detection by learning transformations into a common feature representation across modalities [4]. In contrast to our approach, this paper learned a single representation for the joint modality space, while our work focuses on learning an additional RGB representation which is informed during training by the depth data. Such modality hallucination was explored in [30], which introduced a fusion approach which was able to fill in a missing modality.

**Learning using side information.** Our problem can also be viewed from the learning with side or priviledged information perspective. This is when a learning algorithm has additional knowledge at training time, whether meta data or in our case an additional modality. One then uses this extra information to inform training of a stronger model than could be produced otherwise. The theoretical framework was explored in [34] and a max-margin framework for learning with side-information in the form of bounding boxes, image tags, and attributes was examined in [26], while Shrivastava and Gupta [27] showed how surface normals at training time could produce detection improvement within the DPM framework.

**Network transfer through distillation.** Most related to our work is the concept of network distillation and its extensions. Hinton *et al*. [17] and concurrently Ba *et al*. [3] introduced the idea of model compression and fast transfer of information from one convolutional network to another. Essentially, the output from one network is used as the target probability distribution for a new network. This was shown to reduce training time of a new network and in some cases reduce the number of parameters needed in order to achieve equivalent performance. This approach was further applied for transfering task correlation across domains [33]. Wang *et al*. [37] transfered information across networks without labels by used a ranking loss across video frames to learn a deep representation which mapped patches from the same track closer together than patches from distinct tracks.

Our approach can also be seen as using distillation to learn representations on RGB images by transferring supervision from paired depth images, but we employ joint training instead of staged training as was used in [16] for supervision transfer. In contrast to [16], our focus is different, we are studying the problem of enriching RGB representations using depth as side information. We show the result that learning representations using depth as side information in this manner can lead to representation which when used in conjunction with representations learned on ImageNet lead to boosts in performance for recognition tasks like object detection.

## 3. Modality Hallucination Model

We present a modality hallucination architecture for training an RGB object detection model which incorporates depth side information at training time. Our hallucination network learns a new and complementary RGB image representation which is trained to mimic depth mid-level features. This new representation is combined with the RGB image representation learned through standard fine-tuning.

### 3.1. Architecture Definition

Figure 1 illustrates the training architecture for our hallucination model. We use multi-layer convolutional networks (ConvNets) as our base recognition architecture which have been shown to be very effective for many different recognition tasks. Prior work on RGB-D detection [15] has found success using a two channel model where RGB and depth

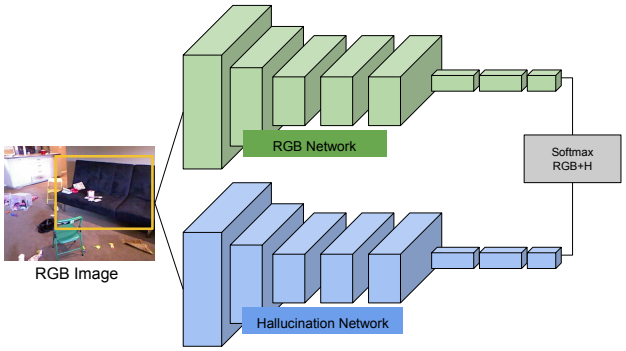


Figure 2: Test time modality hallucination architecture.

images are processed independently with a final detection score being the softmax of the average of both predictions.

For our architecture we build off of this same general model. However, we seek to share information between the two modalities and in particular to use the training time privileged depth modality to inform our final RGB only detector. To accomplish this, we introduce a third channel which we call the *hallucination network* (blue network in Figure 1). The hallucination network takes as input an RGB image and a set of regions of interest and produces detection scores for each category and for each region.

To cause the depth modality to share information with the RGB modality through this hallucination network, we add a regression loss between paired hallucination and depth layers. This choice is inspired by prior work which uses similar techniques for model distillation [17], task correlation transfer across domains [33], and supervision transfer from a well labeled modality to one with limited labels [16]. Essentially, this loss guides the hallucination network to extract features from an RGB image which mimic the responses extracted from the corresponding depth image. We will discuss the details of this loss and its optimization in the next section. It is important that the hallucination network has parameters independent of both the RGB and depth networks as we want the hallucination network activations to match the corresponding depth mid-level activations, however we do not want the feature extraction to be identical to the depth network as the inputs are RGB images for the hallucination network and depth images for the depth network.

At test time, given only an RGB image and regions of interest, we pass our image through both the RGB network and the hallucination network to produce two scores per category, per region, which we average and take the softmax to produce our final predictions (see Figure 2).

### 3.2. Architecture Optimization

In this section we describe the implementation and optimization details for our architecture. At training time we assume access to paired RGB and depth images and regions of interest within the image. We train our model one set of paired images at a time using the Fast R-CNN [10] framework. The RGB and depth network are independently trained using the Fast R-CNN algorithm with the corresponding image input. Next, the hallucination network parameters are intialized with the learned depth network weights before joint training of the three channel network. The choice of initialization for the hallucination parameters is explored in Section 4.1.1. Note, that finetuning of the hallucination network with only a softmax loss on the label space would be equivalent to the training procedure of the RGB network. To faciliate transfer we must use an additional objective by introducing a hallucination loss.

**Hallucination Loss.** We add the objective that activations after some layer, $\ell$, should be similar between the hallucination and depth networks. In particular, we add a euclidean loss between the depth activations $A_\ell^{\text{dNet}}$ and the hallucination activations $A_\ell^{\text{hNet}}$ so that the hallucination loss for the given layer is defined as:

$$\mathcal{L}_{\text{hallucinate}}(\ell) \quad = \quad \|\sigma(A_\ell^{\text{dNet}}) - \sigma(A_\ell^{\text{hNet}})\|_2^2 \qquad (1)$$

where $\sigma(x) = 1/(1+e^{-x})$ is the sigmoid function.

This loss can be applied after any layer in the network and can be optimized directly. However, we are trying to learn an asymmetric transfer of information, namely we seek to inform our RGB hallucination model using the prelearned depth feature extraction network. Therefore, we set the learning rates of all layers lower than the hallucination loss in the depth network to zero. This effectively freezes the depth extractor up to and including layer $\ell$ so that the target depth activations are not modified through backpropagation of the hallucination loss.

**Multi-task Optmization** The full training of our model requires balancing multiple losses. More precisely we have 11 total losses, 5 softmax cross-entropy losses using bounding box labels as targets, 5 Smooth L1 losses [10] using the bounding box coordinates as the targets, and one additional hallucination loss which matches midlevel activations from the hallucination branch to those from the depth branch. The 5 standard supervision and 5 bounding box regression losses operate over each of the three subnetworks, RGB, depth, hallucination, indepdently so that each learns weights that are useful for then final task. We then have 2 joint losses over the average of the final layer activations from both the RGB-depth branches and from the RGB-

hallucination branches. These losses encourage the paired networks to learn complementary scoring functions.

For a given network, N, let us denote the softmax cross-entropy loss over category labels as $\mathcal{L}_{\text{cls}}^{\text{N}}$ and the Smooth L1 loss over bounding box coordinate regression as $\mathcal{L}_{\text{loc}}^{\text{N}}$. Then, the total joint loss of our optimization can be described as follows:

$$\begin{aligned}\mathcal{L} \quad = \quad & \gamma \mathcal{L}_{\text{hallucinate}} \\ & + \alpha \left[ \mathcal{L}_{\text{loc}}^{\text{dNet}} + \mathcal{L}_{\text{loc}}^{\text{rNet}} + \mathcal{L}_{\text{loc}}^{\text{hNet}} + \mathcal{L}_{\text{loc}}^{\text{rdNet}} + \mathcal{L}_{\text{loc}}^{\text{rhNet}} \right] \\ & + \beta \left[ \mathcal{L}_{\text{cls}}^{\text{dNet}} + \mathcal{L}_{\text{cls}}^{\text{rNet}} + \mathcal{L}_{\text{cls}}^{\text{hNet}} + \mathcal{L}_{\text{cls}}^{\text{rdNet}} + \mathcal{L}_{\text{cls}}^{\text{rhNet}} \right]\end{aligned} \tag{2}$$

Balancing these objective is an important part of our joint optimization. For simplicity, we choose to weight all localization losses equivalently and all category losses equivalently. This leaves us with three parameters to set, denoted above as $\alpha$, $\beta$, and $\gamma$.

We set the category loss weights, $\beta = 1.0$, and then let the localization weights be a factor of 2 smaller, $\alpha = 0.5$. Finally, to set the hallucination loss weight will depend on the approximate scale of the loss function. This will vary based on the layer at which the hallucination loss is added. For lower layers in the network, the loss tends to be larger. Thus, a smaller value for $\gamma$ would make sense to avoid the hallucination loss dominating the other objectives. We therefore use a heuristic that the contribution of the hallucination loss should be around 10 times the size of the contribution from any of the other losses. For example, if the contribution from a category loss is about 0.5, then the contribution from the hallucination loss should be around 5. In practice, one can determine this by running a few iterations of training and examining the losses.

**Gradient Clipping** In developing our model, we found that the optimization could be suceptible to outliers causing large varitions in gradient magnitudes for the hallucination loss. One potential way to address this issue would be to set the loss weight very low on the hallucintation loss so that even when a large gradient appears the network optimization does not diverge. However, this will limit the effectiveness of the hallucination loss.

Instead, we have found that a more robust way to train with this euclidean loss is to use gradient clipping. This simply means that when the total gradient (in terms of $\ell 2$ norm) in the network exceeds some threshold, $T$, all gradients are scaled by T / (total norm). Thus, the effective contribution of an outlier example is reduced since the large gradients will be scaled down to the standard range. This approach is simple and already implemented in many standard deep learning packages (ex: it involves a single line change in the Caffe [20] solver file).

# 4. Experiments

We evaluate our model using a standard RGB-D detection dataset, NYUD2 [38]. The NYUD2 dataset consists of 1449 labeled RGB-D images. The dataset is split into train (381 images), val (414 images), and test (654 images) sets. For our ablation experiments we train our model using the train set only and evaluate our model on the validation set. For our overall detection experiment which compares to prior work, we present results on the test set for our algorithm trained using the combined trainval set.

**Base Network.** For the following experiments our base network architecture (used for each of the RGB, depth and hallucination networks), is the single scale Fast R-CNN modification to the AlexNet [21] architecture or the VGG-1024 architecture introduced in [10] as a lower memory modification of VGG [28]. The RGB AlexNet network is initialized with the CaffeNet [20] released weights, which were learned using ILSVRC12 [6] and the RGB VGG-1024 network was intialized with the weights released with Fast R-CNN [10]. We then finetune our RGB network on the NYUD2 dataset. We represent the depth images using the HHA encoding introduced by Gupta *et al*. [15] and independently finetune the depth network after initializing with the RGB weights.

**Region Proposals.** A Fast R-CNN architecture takes as input an image and its corresponding regions of interest. To compute these regions of interest we use two different region proposal algorithms. For the NYUD2 dataset we use multiscale combinatorial grouping (MCG) [1], which has been used in the past for this dataset as it is capable of incorporating depth information into the proposal mechanism. We use the RGB-D version of MCG for training all networks and then use the RGB version at test time. We found this to work better than using RGB MCG for both training and testing by about 1-2%.

**SGD Hyper-parameters.** We optimize our network using the Caffe [20] learning framework. We use a base learning rate of 0.001 and allow all layers of the three channel network to update with the same learning rate, with the exception of the depth network layers below the hallucination loss, which are frozen. We use a momentum of 0.9 and a weight decay of 0.0005. We optimize our ablation experiments for 40K iterations and our full NYUD2 experiment for 60K iterations[1] using a step learning rate policy where the base learning rate is lowered by a factor of 10 ($\gamma = 0.1$) every 30K iterations. Finally, we clip gradients when the L2 norm of the network gradients exceeds 10.

[1]Note that for one of the initial RGB AlexNet models we use the weights released with [16] which was only trained for 40K iterations. We also note that in our experience training the RGB only AlexNet baseline model for more than 40K iterations did not provide any benefit as it does for the joint hallucination model and for the VGG-1024 architecture.

| method | btub | bed | bshelf | box | chair | counter | desk | door | dresser | gbin | lamp | monitor | nstand | pillow | sink | sofa | table | tv | toilet | mAP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RGB only [10] (A) | 7.5 | 50.6 | **36.8** | 1.4 | 30.2 | 34.9 | 10.8 | 21.5 | 27.8 | 16.9 | 26.0 | 32.6 | 20.6 | **25.1** | 31.6 | **36.7** | 14.8 | 25.1 | 54.6 | 26.6 |
| RGB ensemble (A-A) | 10.5 | 53.7 | 33.6 | 1.6 | 32.0 | 34.8 | 12.2 | 20.8 | 34.5 | 19.6 | 28.6 | 45.7 | 28.5 | 24.4 | 31.4 | 34.7 | 14.5 | **34.0** | **56.1** | 29.0 |
| Our Net (A-RGB, A-H) | **13.9** | **56.1** | 34.4 | **1.9** | **32.9** | **40.5** | **12.9** | **22.6** | **37.4** | **22.0** | **28.9** | **46.2** | **31.9** | 22.9 | **34.2** | 34.2 | **19.4** | 33.2 | 53.6 | **30.5** |
| RGB only [10] (V) | 15.6 | 59.4 | 38.2 | 1.9 | 33.8 | 36.3 | 12.1 | 24.5 | 31.6 | 18.6 | 25.5 | 46.5 | 30.1 | 20.6 | 30.3 | 40.5 | 19.5 | **37.8** | 45.7 | 29.9 |
| RGB ensemble (A-V) | 14.8 | 60.4 | **43.1** | **2.1** | 36.4 | 40.7 | 13.3 | **27.1** | 35.5 | 20.8 | 29.9 | **52.9** | **33.5** | 26.2 | 33.0 | 44.4 | 19.9 | 36.7 | 50.2 | 32.7 |
| Our Net (A-RGB, V-H) | **16.8** | **62.3** | 41.8 | **2.1** | **37.3** | **43.4** | **15.4** | 24.4 | **39.1** | **22.4** | **30.3** | 46.6 | 30.9 | **27.0** | **42.9** | **46.2** | **22.2** | 34.1 | **60.4** | **34.0** |

Table 1: **Detection (AP%) on NYUD2 test set:** We compare our performance (pool5 hallucinate) against a Fast R-CNN [10] RGB detector trained on NYUD2 and against an ensemble of Fast R-CNN RGB detectors. AlexNet [21] architecture is denoted as 'A' and VGG-1024 [10, 28] architecture is denoted as 'V'. Our method outperforms both the RGB-only baselines and the RGB ensemble baselines.

### 4.1. NYUD2 Detection Evalutation

Table 1 reports performance of our full system with two different architecture on the NYUD2 dataset. The two base architectures are either AlexNet (indicated as 'A') [21] or VGG-1024 (indicated as 'V') [10, 28]. We train our initial RGB and depth networks using the strategy proposed in [15], but use Fast R-CNN instead of RCNN as used in [15]. We then initialize our hallucination network using the depth parameter values. Finally, we jointly optimize the three channel network structure with a hallucination loss on the pool5 activations. When our hallucination network is labeled with a particular architecture this refers to the choice of the depth network and the hallucination network architecture and the RGB architecture is chosen and indicated separately. In the next two sections we explore our choice of initialization and at which layer to add a hallucination loss.

For each architecture choice we first compare against the corresponding RGB only Fast R-CNN model and find that our hallucination network outperforms this baseline, with 30.5 mAP vs 26.6 mAP for the AlexNet architecture and 34.0 mAP vs 29.9 mAP for the VGG-1024 architecture. Note that for our joint AlexNet method, A-RGB + A-H, we average the APs of the joint model using each of the AlexNet RGB baseline models. As an additional reference, the state-of-the-art performance of RGB-D detection algorithms on NYUD2 is 41.2 mAP [14], 44.4 mAP [15] when run with Fast R-CNN [10] and 47.1 mAP [16]. However, these algorithms operate in the privileged regime with access to depth at test time, thus they are able to achieve the highest overall performance.

It is well known that ensemble methods tend to outperform the single model approach. For example, an ensemble of two ConvNets each initialized randomly and then trained using the same data source, outperforms either model independently [13]. Since our method is the combination of an RGB model trained using a standard supervised approach and an RGB model trained using our depth halluciation technique, we additionally compare our approach to an ensemble of standard trained RGB models. Table 1 reports the performance both for an ensemble of two different AlexNet RGB models, the weights for which were randomly initialized with different seeds before being pre-trained with ImageNet [6], and for an ensemble of an AlexNet RGB model with a VGG-1024 RGB model. We find in both cases that the RGB ensemble improves performance over the single RGB model, while our hallucination model offers the highest performance overall, with 14/19 categories improving for the AlexNet comparisons to ensemble and 13/19 categories improving for the VGG-1024 hallucination net comparisons to ensemble. This suggests that our hallucination model offers more benefit than a simple RGB ensemble.

While our method hallucinates mid-level depth features, other work has proposed hallucinating the pixel level depth from an RGB image. As an additional baseline, we have taken a state-of-the-art depth estimation approach [23] and used the model to produce hallucinated depth images at test time which can be used as input to the depth channel of our pre-trained RGB-D detector. However, doing this performed worse than using our RGB model alone (22% mAP vs 27% mAP) so we have ommitted the results from Table 1. Note that we do not fine-tune our detection model using the depth pixel hallucinations and thus a drop in performance is likely due, at least in part, to the mismatch between the true depth used at training time and the hallucinated depth images used at test time. We refer the interested reader to a related and more comprehensive investigation of pixel depth hallucination by Eigen and Fergus [8] who replaced the true depth input into their network with their hallucinated depths and normals and did fine-tune, yet still did not observe performance improvements for the final semantic segmentation task.

In the next subsections we explore ablation studies and analysis on our hallucination model. For all the following experiments we use the AlexNet RGB and hallucination architecture.

#### 4.1.1 How to initialize the hallucination net?

One important parameter of training our model is how to initialize the hallucination network. We explore three natural choices in Table 2, random initialization, initialization with the RGB network parameter values, and initialization

| Initial Weights | bathtub | bed | bshelf | box | chair | counter | desk | door | dresser | gbin | lamp | monitor | nstand | pillow | sink | sofa | table | tv | toilet | mAP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RGB | 7.5 | 50.4 | 9.9 | **0.9** | **26.2** | **24.9** | 5.8 | **15.8** | 13.0 | **29.8** | 12.0 | 43.1 | 20.9 | 14.7 | 17.9 | 25.3 | **15.1** | 32.5 | 59.1 | 22.4 |
| depth | 9.9 | **52.4** | **14.9** | **0.9** | 24.9 | 24.4 | 4.3 | 15.3 | 18.1 | 24.1 | **14.8** | **45.8** | **27.2** | **18.5** | **21.3** | **29.0** | 13.7 | **33.6** | **66.4** | **24.2** |
| random | **10.5** | 47.6 | 12.3 | 0.6 | 23.5 | 20.2 | **6.0** | 13.0 | **19.3** | 12.0 | 13.3 | 42.8 | 12.8 | 12.1 | 13.6 | 23.0 | 13.9 | 28.6 | 61.5 | 20.3 |

Table 2: **RGB Detection (AP%) on NYUD2 val set:** We compare intializing the hallucination network by randomly initializing or by using the pre-trained RGB or depth parameter values.

with the depth network parameter values. Here we use RGB and depth networks trained using NYUD2 *train set* only and then we use the NYUD2 *validation set* for evaluation of the different choices. We find that both the RGB and depth initialization schemes outperform the baseline RGB only model (20.6% mAP for this setting) and the random initialization model. The depth initialization model has the highest mAP performance and higher AP than the RGB initialization model on 12/19 categories (plus 1 tied category). We thus choose to intialize our hallucination network in all future experiments with the depth parameter values.

#### 4.1.2 Which layer to hallucinate?

Another important parameter of our method is to choose which mid-level activations the hallucination loss should regress to. In Table 3 we systematically explore placing the hallucination loss after each layer from pool1 to fc8. We found that overall adding the hallucination loss at a mid to lower layer improved performance the most over the RGB only baseline network.

The highest overall performance was achieved with the hallucination loss on the pool5 activations. However, the result was not uniformly distributed across all categories. For example, *bathtub* received a noticeably greater performance increase with a hallucination loss at pool1.

We also experimented with adding the hallucination loss at multiple layers in the network, but did not find this to be more effective than pool5 alone.

#### 4.1.3 Does hallucination help on other datasets?

We next study the application of our hallucination network on the Pascal [9] dataset (VOC 2007) which lacks depth data. First, we directly evaluate both the NYUD2 RGB-only network and our NYUD2 RGB plus hallucination network on the four overlapping categories in Pascal. Results for this experiment are reported in the first two rows of Table 4.

We find that our hallucination network provides 3.9% mAP improvements across these four Pascal categories when compared to the RGB-only baseline (from 16.9 to 20.8 mAP). Additionally, we note that there is a dataset shift between Pascal and NYUD2 which causes the overall performance of both methods to be lower than that of a network which was explicitly trained on Pascal. Therefore, we also explore further fine-tuning on the available Pascal

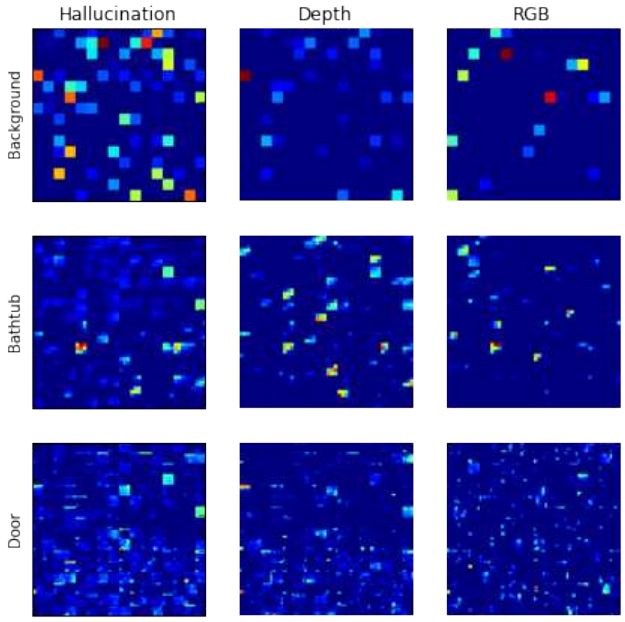


Figure 3: **Roi-pool5 activations** on three top scoring regions from an NYUD2 *test set* image. This figure illustrates the difference between the activations from the three networks.

VOC 2007 trainval set. This set only contains RGB images so we may only further fine-tune the RGB network.

This means that the dataset shift is mitigated in the RGB network but not in the hallucination network. Nevertheless, we find that the combination of our Pascal fine-tuned RGB network with our NYUD2 trained hallucination network continues to outperform the RGB-only baseline, achieving 53.2 mAP instead of 52.1 mAP and higher performance on 3/4 categories.

This indicates that the hallucination technique provides benefit beyond the NYUD2 dataset and we expect that the gains from the hallucination network would only become larger if we were able to adapt the parameters to the new dataset directly.

### 4.2. What did the hallucination net learn?

Regression losses can often be difficult to train together with the supervised cross-entropy loss. We first verify that

| hallucination layer | bathtub | bed | bshelf | box | chair | counter | desk | door | dresser | gbin | lamp | monitor | nstand | pillow | sink | sofa | table | tv | toilet | mAP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| RGB only | 4.9 | 45.5 | 10.9 | 1.3 | 21.5 | 23.6 | 5.4 | 14.5 | 12.7 | 17.4 | 9.4 | 40.9 | 17.2 | 14.9 | 19.9 | 19.2 | 14.0 | 32.5 | 66.3 | 20.6 |
| pool1 | **12.0** | **54.0** | **17.9** | **1.1** | 24.5 | 23.6 | 5.0 | 15.2 | 16.3 | 12.7 | 13.3 | 40.0 | 24.7 | 16.6 | 20.5 | **29.6** | 14.9 | 27.4 | 55.3 | 22.3 |
| pool2 | 8.4 | 50.7 | 13.5 | 1.0 | 24.2 | 26.0 | 6.6 | 13.1 | 13.8 | 17.8 | 11.7 | 40.7 | 21.8 | 15.0 | 20.5 | 22.4 | 15.2 | 27.2 | 59.7 | 21.5 |
| conv3 | 8.8 | 52.5 | 13.2 | 1.0 | 25.6 | 26.2 | 3.3 | 13.2 | 14.9 | 17.0 | **16.2** | 41.6 | 22.2 | **20.2** | 22.9 | 24.6 | **17.2** | **37.4** | 65.6 | 23.3 |
| conv4 | 9.7 | 51.2 | 12.9 | 1.0 | 26.3 | **26.8** | **6.9** | **17.4** | 16.7 | 22.0 | 12.4 | 43.2 | 15.5 | 16.4 | **24.0** | 23.5 | 16.2 | 34.2 | 64.2 | 23.2 |
| pool5 | 9.9 | 52.4 | 14.9 | 0.9 | 24.9 | 24.4 | 4.3 | 15.3 | **18.1** | **24.1** | 14.8 | **45.8** | **27.2** | 18.5 | 21.3 | 29.0 | 13.7 | 33.6 | **66.4** | **24.2** |
| fc6 | 10.3 | 47.2 | 12.0 | 0.6 | 21.7 | 20.0 | 5.9 | 12.8 | 13.8 | 20.5 | 11.8 | 34.4 | 16.3 | 13.1 | 14.8 | 27.3 | 16.1 | 28.8 | 60.5 | 20.4 |
| fc7 | 3.3 | 49.4 | 12.7 | 0.8 | 24.1 | 21.8 | 4.8 | 15.2 | 16.8 | 11.7 | 10.0 | 43.4 | 18.7 | 14.2 | 20.6 | 25.2 | 14.4 | 29.5 | 63.1 | 21.0 |
| fc8 | 4.2 | 50.7 | 13.9 | 0.9 | 23.8 | 23.6 | 5.4 | 15.5 | 18.0 | 13.2 | 13.3 | 42.0 | 20.9 | 15.8 | 22.3 | 23.8 | 14.5 | 29.6 | 63.6 | 21.8 |

Table 3: **RGB Detection (AP%) on NYUD2 val set:** We compare hallucinating different mid-level features with our method.

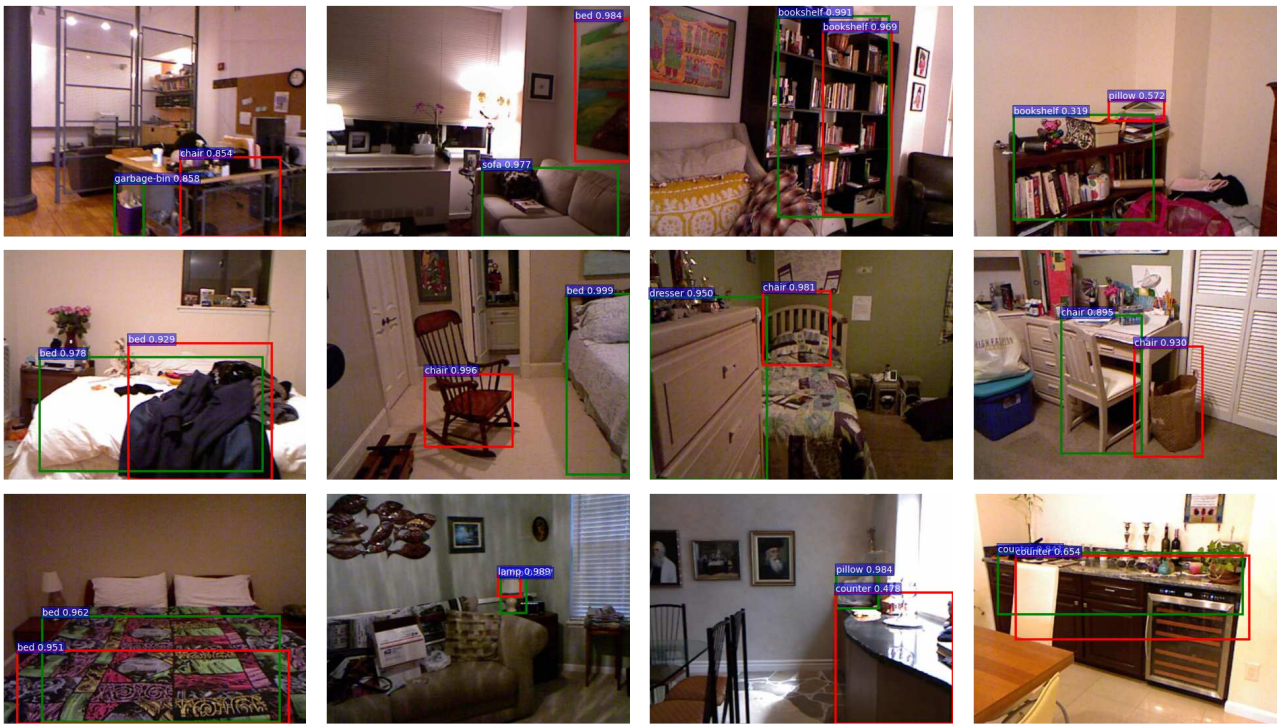


Figure 4: Example Detections on the NYUD2 *test set* where our RGB hallucination network's (green box) top scoring detection for the image is correct while the baseline RGB detector's (red box) top scoring detection is incorrect.

our hallucination loss is effectively learning by examining the training loss vs iteration and confirming that the hallucination loss does indeed decrease.

We next verify that this training loss decrease translates to a decreased loss on the test data and hence a better depth activation alignment. To this end, we examine the network outputs on the NYUD2 test set. We first compute the hallucination loss value across the entire test set before and after learning and find that the value decreases from 216.8 to 94.6.

We additionally compare the euclidean distance between the hallucination activations and the RGB activations and find that after learning, the hallucination and depth activations are closer than the hallucination and RGB activations. Specifically, for the case where the hallucination network was intialized with RGB weights, the hallucination network activations start out being same as the RGB network activations but over time become closer to the depth network as can be seen from the post-training euclidean losses of H-RGB =113.0 while H-HHA=97.5.m

As an example, Figure 3 shows roi-pool5 activations from corresponding regions in the test image which have highest final detection scores. The visualization shows all $256 \times 6 \times 6$ roi-pool5 activations and corresponding region label. This figure illustrates the difference between the RGB activations learned through our approach and through the

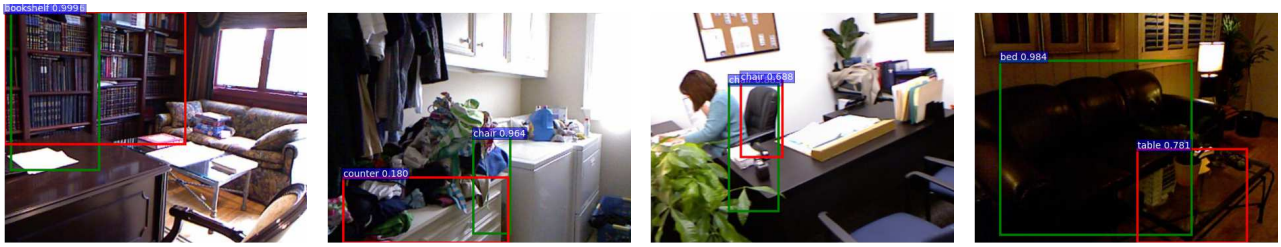


Figure 5: Example Detections on the NYUD2 *test set* where our RGB hallucination network's (green box) top scoring detection for the image is a false positive while the baseline RGB detector's (red box) top scoring detection is a true positive.

| method | chair | dining table | sofa | tv | mAP |
|---|---|---|---|---|---|
| RGB | 17.5 | 13.0 | 10.4 | 26.7 | 16.9 |
| RGB+H | **19.5** | **17.4** | **19.3** | **27.1** | **20.8** |
| RGB (pascal ft) | 33.1 | **63.5** | 49.1 | 62.7 | 52.1 |
| RGB (pascal ft) + H (no ft) | **34.3** | 61.9 | **53.3** | **63.9** | **53.4** |

Table 4: **RGB Detection (AP%) on PASCAL voc 2007 test set:** We compare running our hallucination network on a new dataset. We compare the RGB only vs hallucination network of NYUD2 by first directly applying the networks on pascal. Then we finetune the RGB model on pascal data (leaving the hallucination portion fixed) and continue to find that the nyud trained hallucination model provides performance improvements.

standard learning procedure.

Finally, we know from the detection experiments in the previous section that training with the hallucination loss offers performance improvements over a single RGB model or an ensemble of RGB models trained without the depth hallucination loss. However, it's important to know how the network is improving.

Therefore, in Figure 4, we show randomly sampled images from the NYUD2 test set where the top scored region from our hallucination model corresponds to a true positive and the top scoring region from the single RGB baseline corresponds to a false positive. Our method output is illustrated with a green box and the baseline is illustrated with a red box.

## 5. Conclusion

We have introduced a novel technique for incorporating additional information, in the form of depth images, at training time to improve our test time RGB only detection models. We accomplish this through our modality hallucination architecture which combines a traditional RGB ConvNet representation with an additional and complementary RGB representation which has been trained to hallucinate depth mid-level features. Our approach outperforms the corresponding Fast R-CNN RGB detection models on the NYUD2 dataset.

**Acknowledgements** This work was supported in part by DARPA; AFRL; DoD MURI award N000141110688; NSF awards IIS-1212798, IIS-1427425, and IIS-1536003, and the Berkeley Vision and Learning Center.

## References

[1] P. Arbeláez, J. Pont-Tuset, J. T. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *CVPR*, 2014. 4

[2] Y. Aytar and A. Zisserman. Tabula rasa: Model transfer for object category detection. In *ICCV*, 2011. 2

[3] J. Ba and R. Caruana. Do deep nets really need to be deep? In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2654–2662. Curran Associates, Inc., 2014. 2

[4] L. Chen, W. Li, and D. Xu. Recognizing rgb images by learning from rgb-d data. In *CVPR*, 2014. 2

[5] C. M. Chrisoudias, R. Urtasun, M. Salzmann, and T. Darrell. Learning to recognize objects from unseen modalities. In *ECCV*, 2010. 2

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009. 1, 4, 5

[7] L. Duan, D. Xu, and I. W. Tsang. Learning with augmented features for heterogeneous domain adaptation. In *ICML*, 2012. 2

[8] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015. 5

[9] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010. 6

[10] R. Girshick. Fast R-CNN. *International Conference on Computer Vision (ICCV)*, 2015. 1, 3, 4, 5

[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 1

[12] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, 2012. 2

[13] J. Guo and S. Gould. Deep CNN ensemble with data augmentation for object detection. *CoRR*, abs/1506.07224, 2015. 5

[14] S. Gupta, P. A. Arbeláez, R. B. Girshick, and J. Malik. Aligning 3D models to RGB-D images of cluttered scenes. In *Computer Vision and Pattern Recognition (CVPR)*, 2015. 2, 5

[15] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik. Learning rich features from rgb-d images for object detection and segmentation. In *Computer Vision–ECCV 2014*, pages 345–360. Springer, 2014. 1, 2, 4, 5

[16] S. Gupta, J. Hoffman, and J. Malik. Cross modal distillation for supervision transfer. In *CVPR*, 2016. 2, 3, 4, 5

[17] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2014. 2, 3

[18] J. Hoffman, E. Rodner, J. Donahue, K. Saenko, and T. Darrell. Efficient learning of domain-invariant image representations. In *ICLR*, 2013. 2

[19] A. Janoch, S. Karayev, Y. Jia, J. T. Barron, M. Fritz, K. Saenko, and T. Darrell. A category-level 3D object dataset: Putting the kinect to work. In *Consumer Depth Cameras for Computer Vision*. 2011. 2

[20] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 4

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Proc. NIPS*, 2012. 1, 4, 5

[22] B. Kulis, K. Saenko, and T. Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *CVPR*, 2011. 2

[23] F. Liu, C. Shen, and G. Lin. Deep convolutional neural fields for depth estimation from a single image. In *CVPR*, 2015. 5

[24] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1

[25] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *CoRR*, abs/1312.6229, 2013. 1

[26] V. Sharmanska, N. Quadrianto, and C. Lampert. Learning to rank using privileged information. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 825–832, Dec 2013. 2

[27] A. Shrivastava and A. Gupta. Building part-based object detectors via 3d geometry. In *ICCV*, 2013. 2

[28] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 1, 4, 5

[29] B. soo Kim, S. Xu, and S. Savarese. Accurate localization of 3D objects from RGB-D data using segmentation hypotheses. In *CVPR*, 2013. 2

[30] L. Spinello and K. O. Arras. Leveraging rgb-d data: Adaptive fusion and domain adaptation for object detection. In *ICRA*, 2012. 2

[31] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv:1409.4842*, 2014. 1

[32] S. Tang, X. Wang, X. Lv, T. X. Han, J. Keller, Z. He, M. Skubic, and S. Lao. Histogram of oriented normal vectors for object recognition with a depth sensor. In *ACCV*, 2012. 2

[33] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko. Simultaneous deep transfer across domains and tasks. In *International Conference in Computer Vision (ICCV)*, 2015. 2, 3

[34] V. Vapnik and A. Vashist. A new learning paradigm: Learning using privileged information. *Neural Networks*, 22(56):544 – 557, 2009. Advances in Neural Networks Research: {IJCNN20092009} International Joint Conference on Neural Networks. 2

[35] A. Wang, J. Lu, J. Cai, T. Cham, and G. Wang. Large-margin multi-modal deep learning for rgb-d object recognition. In *IEEE Transactions on Multimedia*, 2015. 1, 2

[36] C. Wang, W. Ren, K. Huang, and T. Tan. Weakly supervised object localization with latent category learning. In *European Conference on Computer Vision (ECCV)*, 2014. 1, 2

[37] X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. In *ICCV*, 2015. 2

[38] E. S. Ye. Object detection in rgb-d indoor scenes. Master's thesis, EECS Department, University of California, Berkeley, Jan 2013. 2, 4

[39] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. Torr. Conditional random fields as recurrent neural networks. In *ICCV*, 2015. 1