# Cross Modal Distillation for Supervision Transfer

Saurabh Gupta Judy Hoffman Jitendra Malik
University of California, Berkeley
{sgupta, jhoffman, malik}@eecs.berkeley.edu

## Abstract

*In this work we propose a technique that transfers supervision between images from different modalities. We use learned representations from a large labeled modality as supervisory signal for training representations for a new unlabeled paired modality. Our method enables learning of rich representations for unlabeled modalities and can be used as a pre-training procedure for new modalities with limited labeled data. We transfer supervision from labeled RGB images to unlabeled depth and optical flow images and demonstrate large improvements for both these cross modal supervision transfers.*

## 1. Introduction

Current paradigms for recognition in computer vision involve learning a generic feature representation on a large dataset of labeled images, and then specializing or finetuning the learned generic feature representation for the specific task at hand. Successful examples of this paradigm include almost all state-of-the-art systems: object detection [11], semantic segmentation [31], object segmentation [17], and pose estimation [44], which start from generic features that are learned on the ImageNet dataset [5] using over a million labeled images and specialize them for each of the different tasks. Several different architectures for learning these generic feature representations have been proposed over the years [26, 39], but all of these rely on the availability of a large dataset of labeled images to learn feature representations.

The question we ask in this work is, what is the analogue of this paradigm for images from modalities which do not have such large amounts of labeled data? There are a large number of image modalities beyond RGB images which are dominant in computer vision, for example depth images coming from a Microsoft Kinect, infra-red images from thermal sensors, aerial images from satellites and drones, LIDAR point clouds from laser scanners, or even images of intermediate representations output from current vision systems *e.g.* optical flow and stereo images. The number of labeled images from such modalities are at least a few orders of magnitude smaller than the RGB image datasets used for learning features, which raises the question: do we need similar large scale annotation efforts to learn generic features for images from each such different modality?

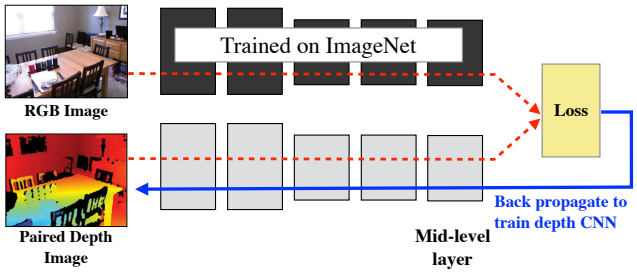


**Figure 1: Architecture for supervision transfer**: We train a CNN model for a new image modality (like depth images), by teaching the network to reproduce the mid-level semantic representations learned from a well labeled image modality (such as RGB images) for modalities for which there are paired images.

We answer this question in this paper and propose a technique to transfer learned representations from one modality to another. Our technique uses 'paired' images from the two modalities and utilizes the mid-level representations from the labeled modality to supervise learning representations on the paired un-labeled modality. We call our scheme ***supervision transfer*** and show that our learned representations perform well on standard tasks like object detection. We also show that our technique leads to learning useful feature hierarchies in the unlabeled modality, which can be improved further with finetuning, and are still complementary to representations in the source modality.

As a motivating example, consider the case of depth images. While the largest labeled RGB dataset, ImageNet [5] consists of over a million labeled images, the size of most existing labeled depth datasets is of the order of a few thousands [37, 41]. At the same time there are a large number of unlabeled RGB and depth image pairs. Our technique leverages this large set of unlabeled paired images to transfer the

Code, data and pretrained models are available at https://github.com/s-gupta/fast-rcnn/tree/distillation.

ImageNet supervision on RGB images to depth images. Our technique is illustrated in Figure 1. We use a convolutional neural network that has been trained on labeled images in the ImageNet dataset [5], and use the mid-level representation learned by these CNNs as a supervisory signal to train a CNN on depth images. This results in improvements in performance for the end task of object detection on the NYUD2 dataset, where we improve the state-of-the-art from 34.2% to 41.7% when using just the depth image and from 46.2% to 49.1% when using both RGB and depth images together. We report similar improvements for the task of simultaneous detection and segmentation [17] and also show how supervision transfer can be used for a zero-shot transfer of object detectors trained on RGB images to detectors that can run on depth images.

Though we show detailed experimental results for supervision transfer from RGB to depth images, our technique is equally applicable to images from other paired modalities. To demonstrate this, we show additional transfer results from RGB images to optical flow images where we improve mean average precision for action detection on the JHMDB dataset [23] from 31.7% to 35.7% when using just the optical flow image and no supervised pre-training.

Our technique is reminiscent of the distillation idea from Hinton *et al*. [20] (and its recent FitNets extension [34]). Hinton *et al*. [20] extended the model compression idea from Bucilua *et al*. [3] to what they call 'distillation' and showed how large models trained on large labeled datasets can be compressed by using the soft outputs from the large model as targets for a much smaller model operating on the same modality. Our work here is a generalization of this idea, and a) allows for transfer of supervision at arbitrary semantic levels, and b) additionally enables transfer of supervision between *different* modalities using paired images. More importantly, our work here allows us to extend the success of recent deep CNN architectures to new imaging modalities without having to collect large scale labeled datasets necessary for training deep CNNs.

## 2. Related Work

There has been a large body of work on transferring knowledge between different visual domains, belonging to the *same* modality. Initial work *e.g*. [1, 7, 13, 27] studied the problem in context of shallow image representations. More recently, with the introduction of supervised CNN models by Krizhevsky *et al*. [26], the community has been moving towards a generic set of features which are specialized to specific tasks and domains at hand [6, 11, 35] and traditional visual adaptation techniques can be used in conjunction with such features [9, 22, 32, 45].

All these lines of work study and solve the problem of domain adaptation within the same modality. In contrast, our work here tackles the problem of domain adaptation across *different* modalities. Most methods for intra-modality domain adaptation described above start from an initial set of features on the target domain, and *a priori* it is unclear how this can be done when moving across modalities, limiting the applicability of aforementioned approaches to our problem. This cross-model transfer problem has received much less attention. Notable among those include [4, 8, 33, 40, 43]. While [4, 43] hallucinate modalities during training time, [8, 33, 40] focus on the problem of jointly embedding or learning representations from multiple modalities into a shared feature space to improve learning [33] or enabling zero-shot learning[8, 40]. Our work here instead transfers high quality representations learned from a large set of labeled images of one modality to completely unlabeled images from a new modality, thus leading to a generic feature representations on the new modalities which we show are useful for a variety of tasks.

## 3. Supervision Transfer

Let us assume we have a modality $\mathcal{U}$ with unlabeled data, $D_u$ for which we would like to train a rich representation. We will do so by transferring information from a separate modality, $\mathcal{L}$, which has a large labeled set of images, $D_l$, and a corresponding $\#_l$ layered rich representation. We assume this rich representation is layered although our proposed method will work equally well for non-layered representations. We use convolutional neural networks as our layered rich representation.

We denote this image representation as $\Phi = \{\phi^i \ \forall i \in \{1, \ldots, \#_l\}\}$. $\phi^i$ is the $i^{th}$ layer representation for modality $\mathcal{L}$ which has been trained on labeled images from dataset $D_l$, and it maps an input image from modality $\mathcal{L}$ to a feature vector in $\mathbb{R}^{n_i}$

$$\phi^i : \mathcal{L} \mapsto \mathbb{R}^{n_i} \tag{1}$$

Feature vectors from consecutive layers in such layered representations are related to one another by simple operations like non-linearities, convolutions, pooling, normalizations and dot products (for example layer 2 features may be related to layer 1 features using a simple non-linearity like $\max$ with 0: $\phi^2(x) = \max(0, \phi^1(x))$). Some of these operations like convolutions and dot products have free parameters. We denote such parameters associated with operation at layer $i$ by $w_l^i$. The structure of such architectures (the sequence of operations, and the size of representations at each layer, *etc*.) is hand designed or validated using performance on an end task. Such validation can be done on a small set of annotated images. Estimating the model parameters $w_l^i$ is much more difficult. The number of these parameters for most reasonable image models can easily go up to a few millions, and state-of-the-art models employ discriminative learning and use large scale labeled training datasets.

Now suppose we want to learn a rich representation for images from modality $\mathcal{U}$, for which we do not have

access to a large dataset of labeled images. We assume we have already hand designed an appropriate architecture $\Psi = \{\psi^i \ \forall i \in \{1, \ldots, \#_u\}\}$. The task then is to effectively learn the parameters associated with various operations in the architecture, without having access to a large set of annotated images for modality $\mathcal{U}$. As before, we denote these parameters to be learned by $W_u^{\{1,\ldots,\#_u\}} = \{w_u^i \ \forall i \in \{1, \ldots, \#_u\}\}$.

In addition to $D_l$, let us assume that we have access to a large dataset of *un-annotated paired* images from modalities $\mathcal{L}$ and $\mathcal{U}$. We denote this dataset by $P_{l,u}$. By paired images we mean a set of images of the same scene in two different modalities. Our proposed scheme for training rich representations for images of modality $\mathcal{U}$ is to learn the representation $\Psi$ such that the image representation $\psi^{\#_u}(I_u)$ for image $I_u$ matches the image representation $\phi^{i^*}(I_l)$ for its image pair $I_l$ in modality $l$ for some chosen and fixed layer $i^* \in \{1, \ldots, \#_l\}$. We measure the similarity between the representations using an appropriate loss function $f$ (for example, euclidean loss). Note that the representations $\phi^{i^*}$ and $\psi^{\#_u}$ may not have the same dimensions. In such cases we embed features $\psi^{\#_u}$ into a space with the same dimension as $\phi^{i^*}$ using an appropriate simple transformation function $t$ (for example a linear or affine function)

$$\min_{W_u^{\{1,\ldots,\#_u\}}} \sum_{(I_l, I_u) \in P_{l,u}} f\left(t\left(\psi^{\#_u}(I_u)\right), \ \phi^{i^*}(I_l)\right) \qquad (2)$$

We call this process *supervision transfer* from layer $i^*$ in $\Phi$ of modality $\mathcal{L}$ to layer $\#_u$ in $\Psi$ of modality $\mathcal{U}$.

The recent distillation method from Hinton *et al*. [20] is a specific instantiation of this general method, where a) they focus on the specific case when the two modalities $\mathcal{L}$ and $\mathcal{U}$ are the same and b) the *supervision transfer* happens at the very last prediction layer, instead of an arbitrary internal layer in representation $\Phi$.

Our experiments in Section 4 demonstrate that this proposed method for transfer of supervision is a) effective at learning good feature hierarchies, b) these hierarchies can be improved further with finetuning, and c) the resulting representation can be complementary to the representation in the source modality $\mathcal{L}$ if the modalities permit.

## 4. Experiments

In this section we present experimental results on 1) the NYUD2 dataset where we use color and depth images as the modality pairs, and 2) the JHMDB video dataset where we use the RGB and optical flow frames as the modality pairs.

Our general experimental framework consists of two steps. The first step is *supervision transfer* as proposed in Section 3, and the second step is to assess the quality of the transferred representation by using it for a downstream task. For both of the datasets we study, we consider the domain of RGB images as $\mathcal{L}$ for which there is a large dataset of labeled images in the form of ImageNet [5], and treat depth and optical flow respectively as $\mathcal{U}$. These choices for $\mathcal{L}$ and $\mathcal{U}$ are of particular practical significance, given the lack of large labeled datasets for depth images, at the same time, the abundant availability of paired images coming from RGB-D sensors (for example Microsoft Kinect) and videos on the Internet respectively.

For our layered image representation models, we use convolutional neural networks (CNNs) [26, 28]. These networks have been shown to be very effective for a variety of image understanding tasks [6]. We experiment with the network architectures from Krizhevsky *et al*. [26] (denoted AlexNet), Simonyan and Zisserman [39] (denoted VGG), and use the models pre-trained on ImageNet [5] from the Caffe [24] Model Zoo.

We use an architecture similar to [26] for the layered representations for depth and flow images. We do this in order to be able to compare to past works which learn features on depth and flow images [12, 15]. Validating different CNN architectures for depth and flow images is a worthwhile scientific endeavor, which has not been undertaken so far, primarily because of lack of large scale labeled datasets for these modalities. Our work here provides a method to circumvent the need for a large labeled dataset for these and other image modalities, and will naturally enable exploring this question in the future, however we do not delve in this question in the current work.

We next describe our design choices for which layers to transfer supervision between, and the specification of the loss function $f$ and the transformation function $t$. We experimented with what layer to use for transferring supervision, and found transfer at mid-level layers works best, and use the last convolutional layer `pool5` for all experiments in the paper. Such a choice also resonates well with observations from [2, 29, 46] that lower layers in CNNs are modality specific (and thus harder to transfer across modalities) and visualizations from [11] that neurons in mid-level layers are semantic and respond to parts of objects. Transferring at `pool5` also has the computational benefit that training can be efficiently done in a fully convolutional manner over the whole image.

For the function $f$, we use $L2$ distance between the feature vectors, $f(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2^2$. We also experimented with $f(\mathbf{x}, \mathbf{y}) = \mathbf{1}(\mathbf{y} > \tau) \cdot \log p(\mathbf{x}) + \mathbf{1}(\mathbf{y} \leq \tau) \cdot \log(1 - p(\mathbf{x}))$ (where $p(x) = \frac{e^{\alpha x}}{1+e^{\alpha x}}$, $\mathbf{1}(x)$ is the indicator function), for some reasonable choices of $\alpha$ and $\tau$ but this resulted in worse performance in initial experiments.

Finally, the choice of the function $t$ varies with different pairs of networks. As noted above, we train using a fully convolutional architecture. This requires the spatial resolution of the two layers $i^*$ in $\Phi$ and $\#_u$ in $\Psi$ to be similar, which is trivially true if the architectures $\Phi$ and $\Psi$ are the same. When they are not (for example when we transfer from VGG net to AlexNet), we adjust the padding in the

| Does supervision transfer work? | | | How good is the transferred representation by itself? | | | Are the representations complementary? | | |
|---|---|---|---|---|---|---|---|---|
| Exp. 1A | no init | 22.7 | Exp. 2A | copy from RGB (ft `fc` only) | 19.8 | Exp. 3A | [RGB]: RGB network on RGB images AlexNet | 22.3 |
| Exp. 1B | copy from RGB | 25.1 | Exp. 2B | supervision transfer (ft `fc` only) AlexNet $^*$ $\rightarrow$ AlexNet | 30.0 | Exp. 3B | [RGB] + copy from RGB | 33.8 |
| Exp. 1C | supervision transfer AlexNet $\rightarrow$ AlexNet | 29.7 | Exp. 2C | supervision transfer (ft `fc` only) VGG $^*$ $\rightarrow$ AlexNet | 32.2 | Exp. 3C | [RGB] + supervision transfer AlexNet $^*$ $\rightarrow$ AlexNet | 35.6 |
| Exp. 1D | supervision transfer AlexNet $^*$ $\rightarrow$ AlexNet | 30.5 | Exp. 2D | supervision transfer VGG $^*$ $\rightarrow$ AlexNet | 33.6 | Exp. 3D | [RGB]+ supervision transfer VGG $^*$ $\rightarrow$ AlexNet | 37.0 |

**Table 1:** We evaluate different aspects of our *supervision transfer* scheme on the object detection task on the NYUD2 *val* set using the mAP metric. Left column demonstrates that our scheme for pre-training is better than alternatives like no pre-training, and copying over weights from RGB networks. The middle column demonstrates that our technique leads to transfer of mid-level semantic features which by themselves are highly discriminative, and that improving the quality of the supervisory network translated to improvements in the learned features. Finally, the right column demonstrates that the learned features on the depth images are still complementary to the features on the RGB image they were supervised with.

AlexNet to obtain the same spatial resolution at `pool5` layer.

This apart, we introduce an adaptation layer comprising of $1 \times 1$ convolutions followed by $ReLU$ to map from the representation at layer $\#_u$ in $\Psi$ to layer $i^*$ in $\Phi$. This accounts for difference in the number of neurons (for example when adapting from VGG to AlexNet), or even when the number of neurons are the same, allows for domain specific fitting. For VGG to AlexNet transfer we also needed to introduce a scaling layer to make the average norm of features comparable between the two networks.

### 4.1. Transfer to Depth Images

We first demonstrate how we transfer supervision from color images to depth images as obtained from a range sensor like the Microsoft Kinect. As described above, we do this set of experiments on the NYUD2 dataset [37] and show results on the task of object detection and instance segmentation [15]. The NYUD2 dataset consists of 1449 paired RGB and D images. These images come from 464 different scenes and were hand selected from the full video sequence while ensuring diverse scene content [37]. The full video sequence that comes with the dataset has over 400K RGB-D frames, we use 10K of these frame pairs for supervision transfer.

In all our experiments we report numbers on the standard *val* and *test* splits that come with the dataset [15, 37]. Images in these splits have been selected while ensuring that all frames belonging to the same scene are contained entirely in exactly one split. We additionally made sure only frames from the corresponding training split were used for supervision transfer.

The downstream task that we study here is that of object detection. We follow the experimental setup from Gupta *et al*. [15] for object detection and study the 19 category object detection problem, and use mean average precision (mAP) to measure performance.

**Baseline Detection Model** We use the model from Gupta *et al*. [15] for object detection. Their method builds off R-CNN [11]. In our initial experiments we adapted their model to the more recent Fast R-CNN framework [10]. We summarize our key findings here. First, [15] trained the final detector on both RGB and D features jointly. We found training independent models all the way and then simply averaging the class scores before the SoftMax performed better. While this is counter-intuitive, we feel it is plausible given limited amount of training data. Second, [15] use features from the `fc6` layer and observed worse performance when using `fc7` representation; in our framework where we are training completely independent detectors for the two modalities, using `fc7` representation is better than using `fc6` representation. Finally, using bounding box regression boosts performance. Here we simply average the predicted regression target from the detectors on the two modalities. All this analysis helped us boost the mean AP on the *test* set from 38.80% as reported by [14, 15] to 44.39%, using the same CNN network and supervision. This already is the state-of-the-art result on this dataset and we use this as a baseline for the rest of our experiments. We denote this model as '[15] + Fast R-CNN'. We followed the default setup for training Fast R-CNN, $40K$ iterations, base learning rate of $0.001$ and stepping it down by a factor of 10 after $30K$ iterations, except that we finetune all the layers, and use $688$px length for the shorter image side. We used RGB-D box proposals from [15] for all experiments.

Note that Gupta *et al*. [15] embed depth images into a geocentric embedding which they call HHA (HHA encodes horizontal disparity, height above ground and angle with gravity) and use the AlexNet architecture to learn HHA features and *copy over the weights from the RGB CNN that was trained for 1000 way classification [26] on ImageNet [5]* to initialize this network. All through this paper, we stick with using HHA embedding[1] to represent the input depth images, and their network architecture, and show how our proposed

[1] We use the term depth and HHA interchangeably.

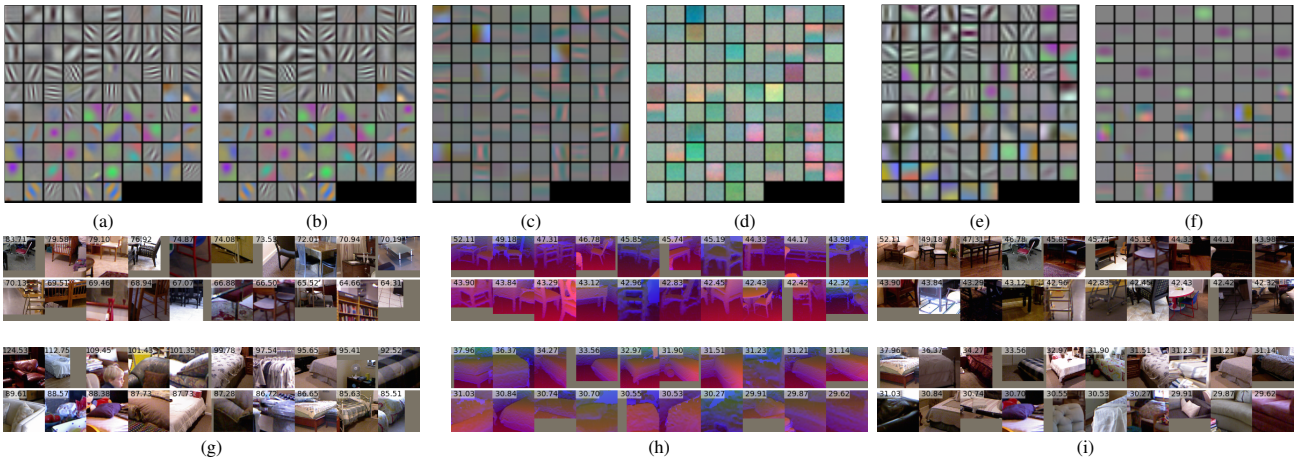


**Figure 2: Visualization of learned filters** (best viewed in color): (a) visualizes filters learned on RGB images from ImageNet data by AlexNet. (b) shows these filters after the finetuning on HHA images, and hardly anything changes visually. (c) shows HHA image filters from our pre-training scheme, which are much different from ones that are learned on RGB images. (d) shows HHA image filters learned without any pre-training. (e) shows optical flow filters learned by [12]. Note that they initialize these filters from RGB filters and these also do not change much over their initial RGB filters. (f) shows filters we learn on optical flow images, which are again very different from filters learned on RGB or HHA images. (g) shows image patches corresponding to highest scoring activations for two neurons in the RGB CNN. (h) shows HHA image patches corresponding to highest scoring activations of the same neuron in the supervision transfer depth CNN. (i) shows the corresponding RGB image patch for these depth image patches for ease of visualization.

*supervision transfer* scheme improves performances over their technique for initialization. We summarize our various transfer experiments below:

**Does *supervision transfer* work?** The first question we investigate is if we are able to transfer supervision to a new modality. To understand this we conducted the following three experiments:

**1. no init (1A)**: randomly initialize the depth network using weight distributions typically used for training on ImageNet and simply train this network for the final task. While training this network we train for 100K iterations, start with a learning rate on 0.01 and step it down by a factor of 10 every 30K iterations.

**2. copy from RGB (1B)**: copy weights from a RGB network that was trained on ImageNet. This is same as the scheme proposed in [15]. This network is then trained using the standard Fast R-CNN settings.

**3. *supervision transfer* (1C)**: train layers `conv1` through `pool5` from random initialization using the *supervision transfer* scheme as proposed in Section 3, on the 5K paired RGB and D images from the video sequence from NYUD2 for scenes contained in the training set. We then plug in these trained layers along with randomly initialized `fc6`, `fc7` and classifier layers for training with Fast R-CNN. We report the results in Table 1. We see that 'copy from RGB' surprisingly does better than 'no init', which is consistent with what Gupta *et al*. report in [15], but our scheme for *supervision transfer* outperforms both these baselines by a large margin pushing up mean AP from 25.1% to 29.7%. We also experimented with using a RGB network $\Psi$ that has been adapted for object detection on this dataset for supervising the transfer (1D) and found that this boosted performance further from 29.7% to 30.5% (1D in Table 1, AlexNet* indicates RGB AlexNet that has been adapted for detection on the dataset). We use this scheme for all subsequent experiments.

**Visualizations.** We visualize the filters from the first layer for these different schemes of transfer in Figure 2(a-f), and observe that our training scheme learns reasonable filters and find that these filters are of different nature than filters learned on RGB images. In contrast, note that schemes which initialize depth CNNs with RGB CNNs weights, filters in the first layer change very little. We also visualize patches giving high activations for neurons paired across RGB and D images Figure 2(g-i). High scoring patches from RGB CNN (AlexNet in this case), correspond to parts of object (g), high scoring patches from the depth CNN also corresponds to parts of the same object class (h and i).

**How good is the transferred representation by itself?** The next question we ask is if our *supervision transfer* scheme transfers good representations or does it only provide a good initialization for feature learning. To answer this question, we conducted the following experiments:

**1. Quality of transferred `pool5` representation (2A, 2B)**: The first experiment was to evaluate the quality of the transferred `pool5` representation. To do this, we froze the network parameters for layers `conv1` through `pool5` to be those learned during the transfer process, and only learn parameters in `fc6`, `fc7` and classifier layers during Fast R-CNN training (2B 'supervision transfer adapted (ft `fc` only)').

| *val* | $AP^r$ at 0.5 | | $AP^r$ at 0.7 | |
|---|---|---|---|---|
| | fc7 | +pool2+conv4 | fc7 | +pool2+conv4 |
| RGB | 26.3 | 29.8 | 14.8 | 18.3 |
| D | 28.4 | 31.5 | 17.4 | 19.6 |

**Table 2: Region detection average precision $AP^r$ on NYUD2 *val* set**: Performance on NYUD2 *val* set where we observe similar boosts in performance when using hyper-column transform with our learned feature hierarchies (learned using supervision transfer on depth images) as obtained with more standard feature hierarchies learned on ImageNet on RGB images.

We see that there is only a moderate degradation in performance for our learned features from 30.5% (1D) to 30.0% (2B) indicating that the features learned on depth images at pool5 are discriminative by themselves. In contrast, when freezing weights when copying from ImageNet (2A), performance degrades significantly to 19.8%.

**2. Improved transfer using better supervising network $\Phi$ (2C, 2D)**: The second experiment investigated if performance improves as we improve the quality of the supervising network. To do this, we transferred supervision from VGG net instead of AlexNet (2C)[2]. VGG net has been shown to be better than AlexNet for a variety of vision tasks. As before we report performance when freezing parameters till pool5 (2C), and learning all the way (2D). We see that using a better supervising net results in learning better features for depth images: when the representation is frozen till pool5 we see performance improves from 30.0% to 32.2%, and when we finetune all the layers performance goes up to 33.6% as compared to 30.5% for AlexNet.

**Is the learned representation complementary to the representation on the source modality?** The next question we ask is if the representation learned on the depth images complementary to the representation on the RGB images from which it was learned. To answer this question we look at the performance when using both the modalities together. We do this the same way that we describe for the baseline model and simply average the category scores and regression targets from the RGB and D detectors. Table 1(right) reports our findings. Just using RGB images (3A) gives us a performance of 22.3%. Combining this with the HHA network as initialized using the scheme from Gupta *et al*. [15] (3B) boosts performance to 33.8%. Initializing the HHA network using our proposed supervision transfer scheme when transferring from AlexNet* to AlexNet (3C) gives us 35.6% and when transferring from VGG* to AlexNet (3D) gives us 37.0%. These results show that the representations are still complementary and using the two together can help the final performance.

**Transfer to other architectures.** We also conducted preliminary experiments of transferring supervision from RGB VGG to a depth VGG network, and found a performance of 33.5% (RGB only VGG performance on the *val* set is 28.0%). Thus, *supervision transfer* can be used to transfer supervision to different target architectures.

[2] To transfer from VGG to AlexNet, we use $150K$ transfer iterations instead of $100K$. Running longer helps for VGG to AlexNet transfer by 1.5% and much less (about 0.5%) for AlexNet to AlexNet transfer.

| *test* | modality | RGB Arch. | D Arch. | $AP^r$ at 0.5 | $AP^r$ at 0.7 |
|---|---|---|---|---|---|
| [18] | RGB | AlexNet | - | 23.4 | 13.4 |
| Gupta *et al*. [14] | RGB + D | AlexNet | AlexNet | 37.5 | 21.8 |
| Our (*supervision transfer*) | RGB + D | AlexNet | AlexNet | **40.5** | **25.4** |
| [18] | RGB | VGG | - | 31.0 | 17.7 |
| Our (*supervision transfer*) | RGB + D | VGG | AlexNet | **42.1** | **26.9** |

**Table 3: Region detection average precision on NYUD2 *test* set.**

**Does supervision transfer lead to meaningful intermediate layer representations?** The next questions we investigate is if the intermediate layers learned in the target modality $\mathcal{U}$ through *supervision transfer* carry useful information. [25] hypothesize that information from intermediate layers in such hierarchies carry information which may be useful for fine grained tasks. Recent work as presented in [18, 31, 36] operationalize this and demonstrate improvements for fine grained tasks like object and part segmentation. Here we investigate if the representations learned using *supervision transfer* also share this property. To test this, we follow the hyper-column architecture from Hariharan *et al*. [18] and study the task of simultaneous detection and segmentation (SDS) [17] and investigate if the use of hyper-columns with our trained networks results in similar improvements as obtained when using more traditionally trained CNNs. We report the results in Table 2. On the NYUD2 dataset, the hyper-column transform improves $AP^r$ from 26.3% to 29.8% when using AlexNet for RGB images. We follow the same experimental setup as proposed in [16], and fix the CNN parameters (to a network that was finetuned for detection on NYUD2 dataset) and only learn the classifier parameters and use features from pool2 and conv4 layers in addition to fc7 for figure ground prediction. When doing the same for our *supervision transfer* network we observe a similar boost in performance from 28.4% to 31.5% when using the hyper-column transform. This indicates that models trained using *supervision transfer* not only learn good representations at the point of supervision transfer (pool5 in this case), but also in the intermediate layers of the network.

**How does performance vary as the transfer point is changed?** We now study how performance varies as we vary the layer used for supervision transfer. We stick to the same experimental setup as used for Exp. 1D in Table 1, and conduct supervision transfer at different layers of the network. Layers above the transfer point are initialized randomly and learned during detector training. For transferring features from layers 1 to 5, we use fully convolutional training as before. But when transferring fc6 and fc7 fea-

| pool1 | pool2 | conv3 | conv4 | pool5 | fc6 | fc7 | conv3 + fc7 |
|---|---|---|---|---|---|---|---|
| 24.4 | 28.4 | 30.6 | 29.9 | 30.5 | 29.7 | 27.7 | 31.3 |

**Table 4: Mean AP on NYUD2 *val* set as a function of layer used for supervision transfer.**

tures we compute them over bounding box proposals (we use RGB-D MCG bounding box proposals [15]) using Spatial Pyramid Pooling on conv5 [10, 19].

We report the obtained AP on the NYUD2 *val* set in Table 4. We see performance is poor when transferring at lower layers (pool1 and pool2). Transfer at layers conv3, conv4, pool5, fc6 works comparably, but performance deteriorates when moving to further higher layers (fc7). This validates our choice for using an intermediate layer as a transfer point. We believe the drop in performance at higher layers is an artifact of the amount of data used for supervision transfer. With a richer and more diverse dataset of paired images we expect transfer at higher layers to work similar or better than transfer at mid-layers. Explained variance during supervision transfer is also higher for transfers at layers 3, 4, and 5 than other layers.

We also conducted some initial experiments with using multiple transfer points. When transferring at conv3 and fc7 we observe performance improves over transferring at either layer alone, indicating learning is facilitated when supervision is closer to parameters being learned. We defer exploration of other choices in this space for future work.

**Is input representation in the form of HHA images still important?** Given our tool for training CNNs on depth images, we can now investigate the question whether hand engineering the input representation is still important. We conduct an experiment in exactly the same settings as Exp. 1D except that we work with disparity images (replicated to have 3 channels) instead of HHA images. This gives a mAP of 29.2% as compared to 30.5% for the HHA images. The difference in performance is smaller than what [15] reports but still significant (14 of 19 categories improve), which suggests that encoding the depth image into a geocentric coordinate frame using the HHA embedding is still useful.

**Applications to zero-shot detection on depth images.** *Supervision transfer* can be used to transfer detectors trained on RGB images to depth images. We do this by the following steps. We first train detectors on RGB images. We then split the network into two parts at an appropriate mid-level point to obtain two networks $\Gamma_{rgb}^{lower}$ and $\Gamma_{rgb}^{upper}$. We then use the lower domain specific part of the network $\Gamma_{rgb}^{lower}$ to train a network $\Gamma_{d}^{lower}$ on depth images to generate the same representation as the RGB network $\Gamma_{rgb}^{lower}$. This is done using the same supervision transfer procedure as before on a set of unlabeled paired RGB-D images. We then construct a 'franken' network with the lower domain specific part coming from $\Gamma_{d}^{lower}$ and the upper more semantic network coming from $\Gamma_{rgb}^{upper}$. We then simply use the output of this franken network on depth images to obtain zero-shot object detection output.

| | Train on MS COCO and adapt to NYUD2 using supervision transfer | | | | | | | | Train on NYUD2 |
|---|---|---|---|---|---|---|---|---|---|
| | bed | chair | sink | sofa | table | tv | toilet | mean | mean |
| RGB | 51.6 | 26.6 | 25.1 | 43.1 | 14.4 | 12.9 | 57.5 | 33.0 | 35.7 |
| D | 59.4 | 27.1 | 23.8 | 32.2 | 13.0 | 13.6 | 43.8 | 30.4 | 45.0 |
| RGB + D | **60.2** | **35.3** | **27.5** | **48.2** | **16.5** | **17.1** | **58.1** | **37.6** | **54.4** |

**Table 5: Adapting RGB object detectors to RGB-D images**: We transfer object detectors trained on RGB images (on MS COCO dataset) to RGB-D images in the NYUD2 dataset, without using any annotations on depth images. We do this by learning a model on depth images using supervision transfer and then use the RGB object detector trained on the representation learned on depth images. We report detection AP(%) on NYUD2 *test* set. These transferred detectors work well on depth images even without using any annotations on depth images. Combining predictions from the RGB and depth image improves performance further.

More specifically, we use Fast R-CNN with AlexNet CNN to train object detectors on the MS COCO dataset [30]. We then split the network right after the convolutional layers pool5, and train a network on depth images to predict the same pool5 features as this network on unlabeled RGB-D images from the NYUD2 dataset (using frames from the *trainval* video sequences). We study all 7 object categories that are shared between MS COCO and NYUD2 datasets, and report the performance in Table 5. We observe our zero-shot scheme for transferring detectors across modalities works rather well. While the RGB detector trained on MS COCO obtains a mean AP of 33.0% on these categories, our zero-shot detector on D images performs comparably and has a mean AP of 30.4%. Note that in doing so we have not used any annotations from the NYUD2 dataset (RGB or D images). Furthermore, combining predictions from RGB and D object detectors results in boost over just using the detector on the RGB image giving a performance of 37.6%. Performance when training detectors using annotations from the NYUD2 dataset (Table 5 last column) is much higher as expected. This can naturally be extended to incorporate annotations from auxiliary categories as explored in [21], but we defer this to future work.

**Performance on test set.** Finally, we report the performance of our best performing supervision transfer scheme (VGG $^*$ $\rightarrow$ AlexNet) on the *test* set in Table 6. When used with AlexNet for obtaining color features, we obtain a final performance of 47.1% which is about 2.7% higher than the current state-of-the-art on this task (Gupta *et al*. [15] Fast R-CNN). We see similar improvements when using VGG for obtaining color features (46.2% to 49.1%). The improvement when using just the depth image is much larger, 41.7% for our final model as compared to 34.2% for the baseline model which amounts to a 22% relative improvement. Note that in obtaining these performance improvements we are using exactly the same CNN architecture and amount of la-

| method | modality | RGB Arch. | D Arch. | mAP |
|---|---|---|---|---|
| Fast R-CNN [10] | RGB | AlexNet | - | 27.8 |
| Fast R-CNN [10] | RGB | VGG | - | **38.8** |
| Gupta *et al*. [15] | RGB + D | AlexNet | AlexNet | 38.8 |
| Gupta *et al*. [14] | RGB + D | AlexNet | AlexNet | 41.2 |
| Gupta *et al*. [15] + Fast R-CNN | RGB + D | AlexNet | AlexNet | 44.4 |
| Our (*supervision transfer*) | RGB + D | AlexNet | AlexNet | **47.1** |
| Gupta *et al*. [15] + Fast R-CNN | RGB + D | VGG | AlexNet | 46.2 |
| Our (*supervision transfer*) | RGB + D | VGG | AlexNet | **49.1** |
| Gupta *et al*. [15] + Fast R-CNN | D | - | AlexNet | 34.2 |
| Our (*supervision transfer*) | D | - | AlexNet | **41.7** |

**Table 6: Object detection mean AP(%) on NYUD2 *test* set:** We compare our performance against several state-of-the-art methods. RGB Arch. and D Arch. refers to the CNN architecture used by the detector. We see when using just the depth image, our method is able to improve performance from 34.2% to 41.7%. When used in addition to features from the RGB image, our learned features improve performance from 44.4% to 47.1% (when using AlexNet RGB features) and from 46.2% to 49.1% (when using VGG RGB features) over past methods for learning features from depth images. We see improvements across almost all categories, performance on individual categories is tabulated in supplementary material.

beled data. We also report performance on the SDS task in Table 3 and obtain state-of-the-art performance of 40.5% as compared to previous best 37.5% [14] when using AlexNet, using VGG CNN for the RGB image improves performance further to 42.1%.

**Training Time.** Finally, we report the amount of time it takes to learn a model using supervision transfer. For AlexNet to AlexNet supervision transfer we trained for 100K iterations which took a total of 2.5 hours on a NVIDIA k40 GPU. This is a many orders of magnitude faster than training models from random initialization on ImageNet scale data using class labels.

## 4.2. Transfer to Flow Images

We now report our experiments for transferring supervision to optical flow images. We consider the end task of action detection on the JHMDB dataset. The task is to detect people doing actions like `catch`, `clap`, `pick`, `run`, `sit` in frames of a video. Performance is measured in terms of mean average precision as in the standard PASCAL VOC object detection task and what we used for the NYUD2 experiments in Section 4.1.

A popular technique for getting better performance at such tasks on video data is to additionally use features computed on the optical flow between the current frame and the next frame [12, 38]. We use *supervision transfer* to learn features for optical flow images in this context.

**Detection model** For JHMDB we use the experimental setup from Gkioxari and Malik [12] and study the 21 class task. Here again, Gkioxari and Malik build off of R-CNN and we first adapt their system to use Fast R-CNN, and observe similar boosts in performance as for NYUD2 when going from R-CNN to Fast R-CNN framework (Table 7, full table with per class performance is in the supplementary material). We denote this model as [12]+[10]. We attribute this large difference in performance to a) bounding box regression and b) number of iterations used for training.

| | RGB | | optical flow | | | |
|---|---|---|---|---|---|---|
| | [12] | [12] + [10] | [12] Sup PreTr | [12] + [10] Sup PreTr | Random Init No PreTr | Our Sup Transfer |
| mAP | 27.0 | **32.0** | 24.3 | **38.4** | 31.7 | 35.7 |

**Table 7: Action Detection AP(%) on the JHMDB *test* set:** We report action detection performance on the *test* set of JHMDB using RGB or flow images. Right part of the table compares our method *supervision transfer* against the baseline of random initialization, and the ceiling using fully supervised pre-training method from [12]. Our method reaches more than half the way towards fully supervised pre-training.

***Supervision transfer* performance** We use the videos from UCF 101 dataset [42] for our pre-training. Note that we do not use any labels provided with the UCF 101 dataset, and simply use the videos as a source of paired RGB and flow images. We take 5 frames from each of the $9K$ videos in the *train1* set. We report performance on JHMDB *test* set in Table 7. Note that JHMDB has 3 splits and as in past work, we report the AP averaged across these 3 splits.

We report performance for three different schemes for initializing the flow model: a) **Random Init** (No PreTr) when the flow network is initialized randomly using the weight initialization scheme used for training a RGB model on ImageNet, b) **Supervised Pre-training** ([12]+[10] Sup PreTr) on flow images from UCF 101 for the task of video classification starting from RGB weights as done by Gkioxari and Malik [12] and c) **supervision transfer** (Our Sup Transfer) from an RGB model to train optical flow model as per our proposed method. We see that our scheme for *supervision transfer* improves performance from 31.7% achieved when using random initialization to 35.7%, which is more than half way towards what fully supervised pre-training can achieve (38.4%), thereby illustrating the efficacy of our proposed technique.

**Acknowledgments:** We thank Georgia Gkioxari for sharing her wisdom and experimental setup for the UCF 101 and JHMDB datasets. This work was supported by ONR SMARTS MURI N00014-09-1-1051, a grant from Intel Corporation, a Berkeley Graduate Fellowship, a Google Fellowship in Computer Vision and a NSF Graduate Research Fellowship. We gratefully acknowledge NVIDIA corporation for the Tesla and Titan GPUs used for this research.

## References


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

[2] H. Azizpour, A. Razavian, J. Sullivan, A. Maki, and S. Carlsson. From generic to specific deep representations for visual recognition. In *CVPR Workshops*, 2015.

[3] C. Bucilua, R. Caruana, and A. Niculescu-Mizil. Model compression. In *ACM SIGKDD*, 2006.

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

[5] 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.

[6] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *ICML*, 2014.

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

[8] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, T. Mikolov, et al. Devise: A deep visual-semantic embedding model. In *NIPS*, 2013.

[9] Y. Ganin and V. Lempitsky. Unsupervised Domain Adaptation by Backpropagation. *ArXiv e-prints*, Sept. 2014.

[10] R. Girshick. Fast R-CNN. In *ICCV*, 2015.

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

[12] G. Gkioxari and J. Malik. Finding action tubes. In *CVPR*, 2015.

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

[14] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik. Aligning 3D models to RGB-D images of cluttered scenes. In *CVPR*, 2015.

[15] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik. Learning rich features from RGB-D images for object detection and segmentation. In *ECCV*, 2014.

[16] B. Hariharan. *Beyond Bounding Boxes: Precise Localization of Objects in Images*. PhD thesis, EECS Department, University of California, Berkeley, Aug 2015.

[17] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *ECCV*, 2014.

[18] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *CVPR*, 2015.

[19] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, 2014.

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

[21] J. Hoffman, S. Gupta, J. Leong, S. Guadarrama, and T. Darrell. Cross-modal adaptation for RGB-D detection. In *ICRA*, 2016.

[22] J. Hoffman, E. Tzeng, J. Donahue, , Y. Jia, K. Saenko, and T. Darrell. One-shot learning of supervised deep convolutional models. In *arXiv 1312.6204; presented at ICLR Workshop*, 2014.

[23] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards understanding action recognition. In *ICCV*, 2013.

[24] Y. Jia. Caffe: An open source convolutional architecture for fast feature embedding. `http://caffe.berkeleyvision.org/`, 2013.

[25] J. J. Koenderink and A. J. van Doorn. Representation of local geometry in the visual system. *Biological cybernetics*, 55(6):367–375, 1987.

[26] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012.

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

[28] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1989.

[29] K. Lenc and A. Vedaldi. Understanding image representations by measuring their equivariance and equivalence. In *CVPR*, 2015.

[30] T. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014.

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

[32] M. Long and J. Wang. Learning transferable features with deep adaptation networks. *CoRR*, abs/1502.02791, 2015.

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

[34] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

[35] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *ICLR*, 2014.

[36] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. LeCun. Pedestrian detection with unsupervised multi-stage feature learning. In *CVPR*, 2013.

[37] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from RGBD images. In *ECCV*, 2012.

[38] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014.

[39] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[40] R. Socher, M. Ganjoo, C. D. Manning, and A. Ng. Zero-shot learning through cross-modal transfer. In *NIPS*, 2013.

[41] S. Song, S. P. Lichtenberg, and J. Xiao. SUN RGB-D: A RGB-D scene understanding benchmark suite. In *CVPR*, 2015.

[42] K. Soomro, A. R. Zamir, and M. Shah. UCF101: A dataset of 101 human action classes from videos in the wild. In *CRCV-TR-12-01*, 2012.

[43] N. Srivastava and R. Salakhutdinov. Multimodal learning with deep boltzmann machines. *JMRL*, 2014.

[44] S. Tulsiani and J. Malik. Viewpoints and keypoints. In *CVPR*, 2015.

[45] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep domain confusion: Maximizing for domain invariance. *CoRR*, abs/1412.3474, 2014.

[46] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *NIPS*, 2014.