Learning a Discriminative Model for the Perception of Realism in Composite Images

Jun-Yan Zhu
UC Berkeley

Philipp Krähenbühl
UC Berkeley

Eli Shechtman
Adobe Research

Alexei A. Efros
UC Berkeley

Abstract

What makes an image appear realistic? In this work, we are looking at this question from a data-driven perspective, by learning the perception of visual realism directly from large amounts of unlabeled data. In particular, we train a Convolutional Neural Network (CNN) model that distinguishes natural photographs from automatically generated composite images. The model learns to predict visual realism of a scene in terms of color, lighting and texture compatibility, without any human annotations pertaining to it. Our model outperforms previous works that rely on hand-crafted heuristics for the task of classifying realistic vs. unrealistic photos. Furthermore, we apply our learned model to compute optimal parameters of a compositing method, to maximize the visual realism score predicted by our CNN model. We demonstrate its advantage against existing methods via a human perception study.

1. Introduction

The human ability to very quickly decide whether a given image is "realistic", *i.e.* a likely sample from our visual world, is very impressive. Indeed, this is what makes good computer graphics and photographic editing so difficult. So many things must be "just right" for a human to perceive an image as realistic, while a single thing going wrong will likely hurtle the image down into the Uncanny Valley [18].

Computers, on the other hand, find distinguishing between "realistic" and "artificial" images incredibly hard. Much heated online discussion was generated by recent results suggesting that image classifiers based on Convolutional Neural Network (CNN) are easily fooled by random noise images [19,29]. But in truth, no existing method (deep or not) has been shown to reliably tell whether a given image resides on the manifold of natural images. This is because the spectrum of unrealistic images is much larger than the spectrum of natural ones. Indeed, if this was not the case, photo-realistic computer graphics would have been solved long ago.

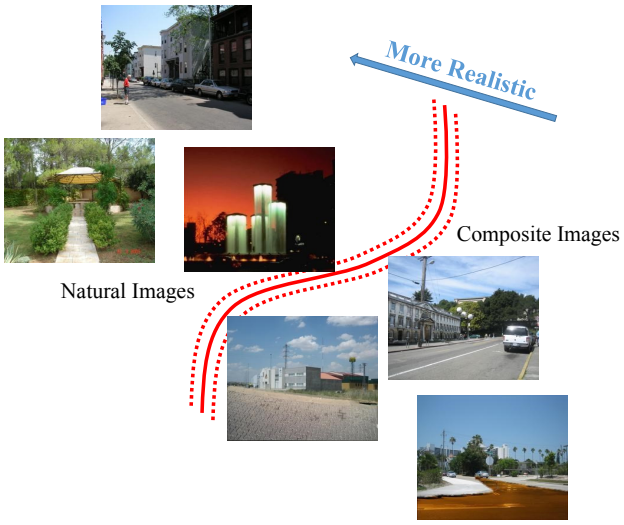

Figure 1: We train a discriminative model to distinguish natural images (top left) and automatically generated image composites (bottom right). The red boundary illustrates the decision boundary between two. Our model is able to predict the *degree* of perceived visual realism of a photo, whether it's an actual natural photo, or a synthesized composite. For example, the composites close to the boundary appear more realistic.

In this paper, we are taking a small step in the direction of characterizing the space of natural images. We restrict the problem setting by choosing to ignore the issues of image layout, scene geometry, and semantics and focus purely on appearance. For this, we use a large dataset of automatically generated image composites, which are created by swapping similarly-shaped object segments of the same object category between two natural images [15]. This way, the semantics and scene layout of the resulting composites are kept constant, only the object appearance changes. Our goal is to predict whether a given image composite will be perceived as realistic by a human observer. While this is admittedly a limited domain, we believe the problem still reveals the complexity and richness of our vast visual space, and therefore can give us insights about the structure of the

manifold of natural images.

Our insight is to train a high-capacity discriminative model (a Convolutional Neural Network) to distinguish natural images (assumed to be realistic) from automatically-generated image composites (assumed to be unrealistic). Clearly, the latter assumption is not quite valid, as a small number of "lucky" composites will, in fact, appear as realistic as natural images. But this setup allows us to train on a very large visual dataset without the need of costly human labels. One would reasonably worry that a classifier trained in this fashion might simply learn to distinguish natural images from composites, regardless of their perceived realism. But, interestingly, we have found that our model appears to be picking up on cues about visual realism, as demonstrated by its ability to rank image composites by their perceived realism, as measured by human subjects. For example, Figure 1 shows two composites which our model placed close to the decision boundary – these turn out to be composites which most of our human subjects thought were natural images. On the other hand, the composite far from the boundary is clearly seen by most as unrealistic. Given a large corpus of natural and composite training images, we show that our trained model is able to predict the degree of realism of a new image. We observe that our model mainly characterizes the visual realism in terms of color, lighting and texture compatibility.

We also demonstrate that our learned model can be used as a tool for creating better image composites automatically via simple color adjustment. Given a low-dimensional color mapping function, we directly optimize the visual realism score predicted by our CNN model. We show that this outperforms previous color adjustment methods on a large-scale human subjects study. We also demonstrate how our model can be used to choose an object from a category that best fits a given background at a specific location.

2. Related Work

Our work attempts to characterize properties of images that look realistic. This is closely related to the extensive literature on natural image statistics. Much of that work is based on generative models [6, 22, 35]. Learning a generative model for full images is challenging due to their high dimensionality, so these works focus on modeling local properties via filter responses and small patch-based representations. These models work well for low-level imaging tasks such as denoising and deblurring, but they are inadequate for capturing higher level visual information required for assessing photo realism.

Other methods take a discriminative approach [9, 17, 25, 27, 33]. These methods can generally attain better results than generative ones by carefully simulating examples labeled with the parameters of the data generation process (e.g. joint velocity, blur kernel, noise level, color transformation). Our approach is also discriminative, however, we generate the negative examples in a non-task-specific way and without recording the parameters of the process. Our intuition is that using large amounts of data leads to an emergent ability of the method to evaluate photo realism *from the data itself*.

In this work we demonstrate our method on the task of assessing realism of image composites. Traditional image compositing methods try to improve realism by suppressing artifacts that are specific to the compositing process. These include transition of colors from the foreground to the background [1,20], color inconsistencies [15,23,24,33], texture inconsistencies [4, 11], and suppressing "bleeding" artifacts [31]. Some work best when the foreground mask aligns tightly with the contours of the foreground object [15, 23, 24, 33], while others need the foreground mask to be rather loose and the two backgrounds not too cluttered or too dissimilar [4, 8, 16, 20, 31]. These methods show impressive visual results and some are used in popular image editing software like Adobe Photoshop, however they are based on hand-crafted heuristics and, more importantly, do not directly try to improve (or measure) the realism of their results. A recent work [30] explored the perceptual realism of outdoor composites but focused only on lighting direction inconsistencies.

The work most related to ours, and a departure point for our approach, is Lalonde and Efros [15] who study color compatibility in image composites. They too generate a dataset of image composites and attempt to rank them on the basis of visual realism. However, they use simple, hand-crafted color-histogram based features and do not do any learning.

Our method is also superficially related to work on digital image forensics [12, 21] that try to detect digital image manipulation operations such as image warping, cloning, and compositing, which are not perceptible to the human observer. But, in fact, the goals of our work are entirely different: rather than detecting which of the realistic-looking images are fake, we want to predict which of the fake images will look realistic.

3. Learning the Perception of Realism

Our goal is developing a model that could predict whether or not a given image will be judged to be realistic by a human observer. However, training such a model directly would require a prohibitive amount of human-labeled data, since the negative (unrealistic) class is so vast. Instead, our idea is to train a model for a different "pretext" task, which is: 1) similar to the original task, but 2) can be trained with large amounts of unsupervised (free) data. The "pretext" task we propose is to discriminate between natural images and computer-generated image composites. A high-capacity convolutional neural network (CNN) clas-

(a) Fully Supervised (b) Partially Supervised (c) Unsupervised

Figure 2: Example composite images for CNN training: (a) image composites generated by fully supervised foreground and background masks, (b) image composites generated by a hybrid ground truth mask and object proposal, (c) image composites generated by a fully unsupervised proposal system. See text for details. Best viewed in color.

sifier is trained using only automatically-generated "free" labels (i.e. natural vs. generated). While this "pretext" task is different from the original task we wanted to solve (realistic vs. unrealistic), our experiments demonstrate that it performs surprisingly well on our manually-annotated test set (c.f. Section 6).

We use the network architecture of the recent VGG model [28], a 16-layer model with small 3×3 convolution filters. We initialize the weights on the ImageNet classification challenge [5] and then fine-tune on our binary classification task. We optimize the model using back-propagation with Stochastic Gradient Descent (SGD) using Caffe [10].

3.1. Automatically Generating Composites

To generate training data for the CNN model, we use the LabelMe image dataset [26] because it contains many categories along with detailed annotation for object segmentation. For each natural image in the LabelMe dataset, we generate a few composite images as follows.

Generate a Single Composite Figure 3 illustrates the process of generating a single composite image, which follows [15]. Starting with a background image B (Figure 3c) that contains an object of interest (target object), we locate a source object F (Figure 3a) with a similar shape elsewhere in the dataset, and then rescale and translate the source object F so that the source object matches the target location. (Figure 3b). We assume the object is well segmented and the alpha map α of the source object is known (Figure 3d). We apply a simple feathering based on a distance transform map to the object mask α of the source object. We generate the final composite by combining the source object and background $I = \alpha \cdot F + (1 - \alpha) \cdot B$.

Generate Composite Dataset For each target object in each image, we search for source objects with similar shapes by computing the SSD of blurred and subsampled

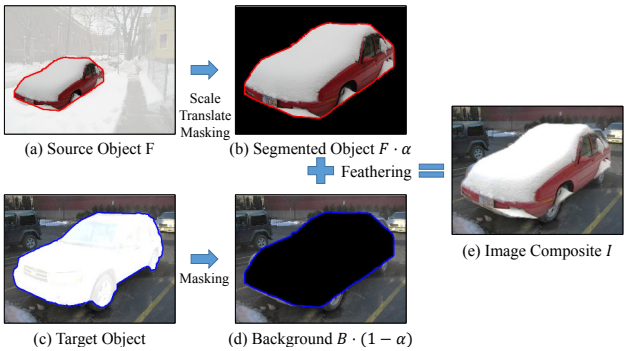

Figure 3: We generate a composite image by replacing the target object (c) by the source object F (a). We rescale and translate the source object to match the location and scale of the target object (c). We generate the final composite (e) by combining the segmented object (b) and the masked background (d).

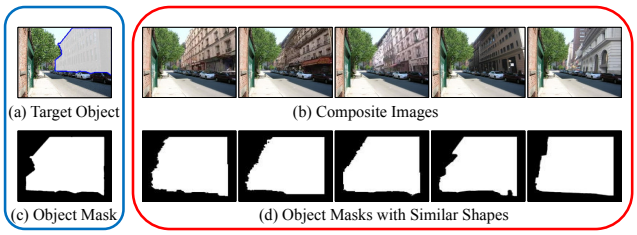

Figure 4: Given an original photo with target object (a) and its object mask (c), we search for source objects whose object mask matches well the shape of target object, and replace the target object with them. We show the nearest neighbor object masks in (d) and their corresponding generated composites (b).

(64×64) object masks. Take Figure 4, for example. We replace the original building with other buildings with similar outlines. The purpose of the rough matching of object shape is to make sure that the generated composites are already close to the manifold of natural images. However, this procedure requires detailed segmentation annotations for both source and target objects. We call this procedure *FullySupervised* as it requires full annotation of object masks.

An alternative way is to use automatic image segmentation produced by an "object proposal" method (in our implementation we used Geodesic Object Proposals [13]). In this case, training images are still generated using human labeled segmentation for the target objects, but source objects are obtained by searching for object proposal segments with similar shapes to the target objects in all images. This requires much fewer segmented training images. We name this procedure *PartiallySupervised*. The third way is fully automatic: we use object proposals for both source and target objects. In particular, we randomly sample an object proposal for a given image, and replace it by other object

(a) Most realistic composites ranked by our model

(b) Least realistic composites ranked by our model

Figure 5: Ranking of generated composites in terms of realism scores. Best viewed in color.

proposals with the most similar shapes from the dataset. This procedure is fully unsupervised and we call it *Unsupervised*. Later, we show that this fully automatic procedure only performs slightly worse than *FullySupervised* w.r.t human annotations, in terms of predicting visual realism (Section 6). We also experimented with randomly cutting and pasting objects from one image to the other without matching object masks. In this case, the CNN model we trained mainly picked up artifacts of high-frequency edges that appear in image composites and performed significantly worse. In our experiments, we used $\sim 11,000$ natural images containing $\sim 25,000$ object instances from the largest 15 categories of objects in the LabelMe dataset. For *FullySupervised* and *PartiallySupervised*, we generated a composite image for each annotated object in the image. For *Unsupervised*, we randomly sample a few object proposals as target objects, and generate a composite image for each of them.

Figure 2 shows some examples of image composites generated by all three methods. Notice that some composite images are artifact-free and appear quite realistic, which forces the CNN model to pick up not only the artifacts of the segmentation and blending algorithms, but also the compatibility between the visual content of the inserted object and its surrounding scene. Different from previous work [15], we do not manually remove any structurally inconsistent images. We find that composites generated by *FullySupervised* are usually correct with regards to semantics and geometry, but sometimes suffer from inconsistent lighting and color. *PartiallySupervised* also often generates meaningful scenes, but sometimes tends to paste an object into parts of another object. While *Unsupervised* tends to generate scenes with incorrect semantics, the number of scenes that can be generated is not restricted by the limited amount of human annotation.

Ranking of Training Images Interestingly, our trained CNN model is able to rank visually appealing image composites higher than unrealistic photos with visual artifacts. In Figure 5, we use our model to rank the training composites by their realism score prediction. The top row shows high-quality composites that are difficult for humans to spot while the bottom row shows poor composites due to incorrect segmentation and color inconsistency. We demonstrate that our model matches to human perception with quantitative experiments in Section 6.

4. Improving Image Composites

Let $f(I;\theta)$ be our trained CNN classifier model predicting the visual realism of an image I. We can use this classifier to guide an image compositing method to produce more realistic outputs. This optimization not only improves object composition, but also reveals many of the properties of our learned realism model.

We formulate the object composition process as $I_g = \alpha \cdot g(F) + (1-\alpha) \cdot B$ where F is the source object, B is the background scene, and $\alpha \in [0,1]$ is the alpha mask for the foreground object. For this task, we assume that the foreground object is well segmented and placed at a reasonable location. The color adjustment model $g(\cdot)$ adjusts the visual properties of the foreground to be compatible with the background image. Color plays an important role in the object composition process [15]. Even if an object fits well to the scene, the inconsistent lighting will destroy the illusion of realism.

The goal of a color adjustment is to optimize the adjustment model $g(\cdot)$, such that the resulting composite appears realistic. We express this in the following objective function:

$$E(g,F) = -f(I_g;\theta) + w \cdot E_{reg}(g), \tag{1}$$

where f measures the visual realism of the composite and E_{reg} imposes a regularizer on the space of possible adjustments. A desired image composite should be realistic while staying true to identity of the original object (e.g. do not turn a white horse to be yellow). The weight w controls the relative importance between the two terms (we set it to $w = 50$ in all our experiments). We apply a very simple brightness and contrast model to the source object F for each channel independently. For each pixel we map the foreground color values $F^p = (c_1^p, c_2^p, c_3^p)$ to $g(F^p) = (\lambda_1 c_1^p + \beta_1, \lambda_2 c_2^p + \beta_2, \lambda_3 c_3^p + \beta_3)$. The regularization term for this model can be formulated as:

$$E_{reg}(g) = \frac{1}{N} \sum_p \Big(\|I_g^p - I_0^p\|_2 + \sum_{i,j} \|(\lambda_i - 1) \cdot c_i^p + \beta_i - (\lambda_j - 1) \cdot c_j^p - \beta_j\|_2 \Big) \tag{2}$$

where N is the number of foreground pixels in the image, and $I_0 = \alpha \cdot F + (1-\alpha) \cdot B$ is the composite image without recoloring, I_g^p and I_0^p denotes the color values for pixel p in the composite image. The first term penalizes large change between the original object and recolored

object, and the second term discourages independent color channel variations (roughly hue change).

Note that the discriminative model θ has been trained and fixed during this optimization.

Optimizing Color Compatibility We would like to optimize color adjustment function $g^* = \arg\min_g E(g, F)$. Our objective (Equation 1) is differentiable, if the color adjustment function g is also differentiable. This allows us to optimize for color adjustment using gradient-descent.

To optimize the function, we decompose the gradient into $\frac{\partial E}{\partial g} = -\frac{\partial f(I_g,\theta)}{\partial I_g} \cdot \frac{\partial I_g}{\partial g} + \frac{\partial E_{reg}}{\partial g}$. Notice that $-\frac{\partial f(I_g,\theta)}{\partial I_g}$ can be computed through backpropagation of CNN model from the loss layer to the image layer while the other parts have a simple close form of gradient. See supplemental material for the gradient derivation. We optimize the cost function using L-BFGS-B [2]. Since the objective is non-convex, we start from multiple random initializations and output the solution with the minimal cost.

In Section 6.1, we compare our model to existing methods, and show that our method generates perceptually better composites. Although our color adjustment model is relatively simple, our learned CNN model provides guidance towards better color compatible composite.

Selecting Best-fitting Objects Imagine that a user would like to place a car on a street scene (e.g. as in [16]). Which car should she choose? We could choose an object $F^* = \arg\min_F E(g, F)$. For this, we essentially generate a composite image for each candidate car instance and select the object with minimum cost function (Equation 1). We show our model can select more suitable objects for composition task in Section 6.2.

5. Implementation

CNN Training We used the VGG model [28] from the authors' website, which is trained on ImageNet [5]. We then fine-tune the VGG Net on our binary classification task (natural photos vs. composites). We optimize the CNN model using SGD. The learning rate α is initialized to 0.0001 and reduced by factor 0.1 after $10,000$ iterations. We set the learning rate for $fc8$ layer to be 10 times higher than the lower layers. The momentum is 0.9, the batch size 50, and the maximum number of iterations $25,000$.

Dataset Generation For annotated objects and object proposals in the LabelMe dataset [26], we only consider objects whose pixels occupy between $5\% \sim 50\%$ of image pixels. For human annotation, we exclude occluded objects whose object label strings contain the words "part", "occlude", "regions" and "crop".

6. Experiments

We first evaluate our trained CNN model in terms of classifying realistic photos vs. unrealistic ones.

Methods without object mask	
Color Palette [15] (no mask)	0.61
VGG Net [28] + SVM	0.76
PlaceCNN [34] + SVM	0.75
AlexNet [14] + SVM	0.73
RealismCNN	0.84
RealismCNN + SVM	**0.88**
Human	0.91
Methods using object mask	
Reinhard *et al.* [23]	0.66
Lalonde and Efros [15] (with mask)	0.81

Table 1: Area under ROC curve comparing our method against previous methods [15, 23]. Note that several methods take advantage of human annotation (object mask) as additional input while our method assumes no knowledge of the object mask.

Evaluation Dataset We use a public dataset of 719 images introduced by Lalonde and Efros [15], which comprises of 180 natural photographs, 359 unrealistic composites, and 180 realistic composites. The images were manually labeled by three human observers with normal color vision. All methods are evaluated on a binary realistic vs. unrealistic classification task with 359 unrealistic photos versus 360 realistic photos (which include natural images plus realistic composites). Our method assigns a visual realism score to each photo. Area under ROC curve is used to evaluate the classification performance. We call our method *RealismCNN*. Although trained on a different loss function (i.e. classifying natural photos vs. automatically generated image composites), with no human annotations for visual realism, our model outperforms previous methods that build on matching low-level visual statistics including color std/mean [23], color palette, texture and color histogram [15]. Notice that Lalonde and Efros [15] also requires a mask for the inserted object, making the task much easier, but less useful.

Supervised Training Without any human annotation for visual realism, our model already outperforms previous methods. But it would be more interesting to see how our *RealismCNN* model improves with a small additional amount of human realism labeling. For this, we use the human annotation (realistic photos vs. unrealistic photos) provided by [15], and train a linear SVM classifier [3] on top of the $fc7$ layer's 4096 dimensional features extracted by our *RealismCNN* model, which is a common way to adapt a pre-trained deep model to a relatively small dataset. We call this *RealismCNN + SVM*. Figure 6 shows a few composites ranked with this model. In practice, $fc6$ and $fc7$ layers give similar performance, and higher compared to lower layers. We evaluate our SVM model using 10-fold cross-

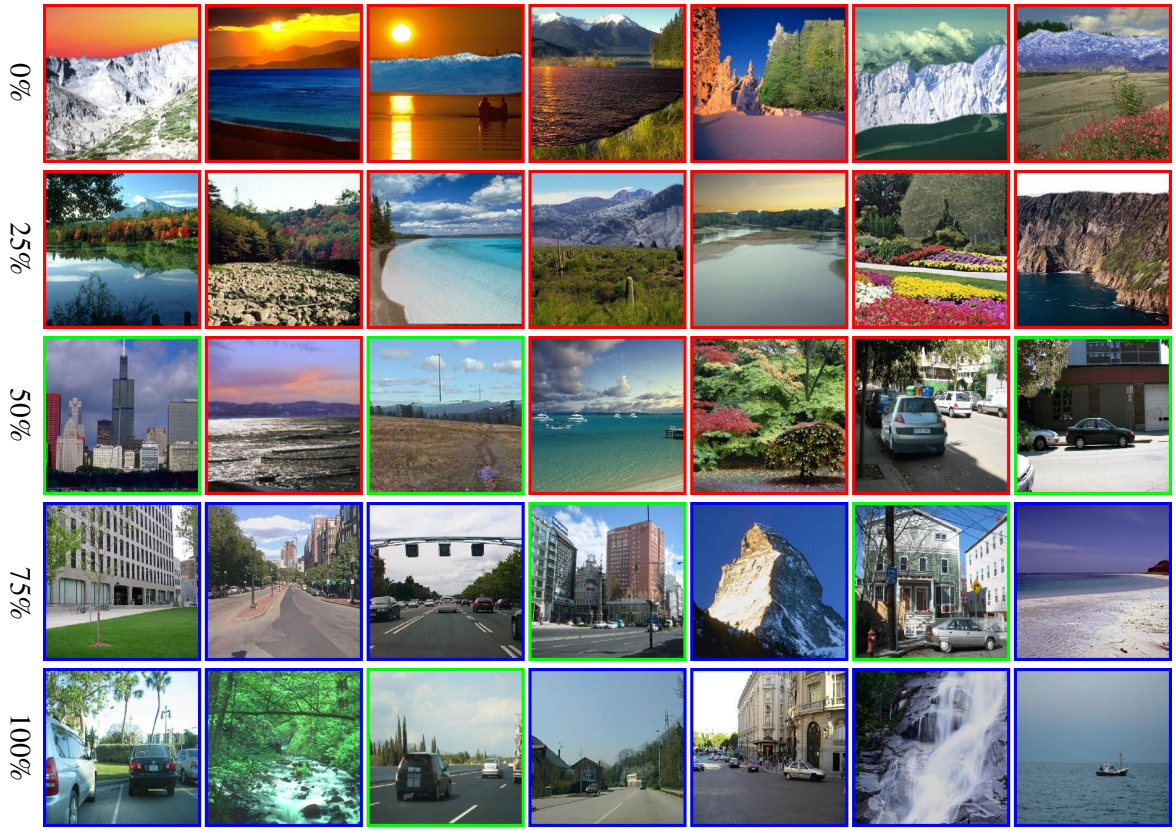

Figure 6: Ranking of photos according to our model's visual realism prediction. The color of image border encodes the human annotation: green: realistic composites; red: unrealistic composites; blue: natural photos. The different rows contain composites corresponding to different rank percentiles of scores predicted with *RealismCNN + SVM*.

validation. This adaptation further improves the accuracy of visual realism prediction. As shown in Table 1, *RealismCNN + SVM* (0.88) outperforms existing methods by a large margin. We also compare our SVM model with other SVM models trained on convolutional activation features ($fc7$ layer) extracted from different CNN models including AlexNet [14] (0.75), PlaceCNN [34] (0.73) and original VGG Net [28] (0.76). As shown in Table 1, our *Realism + SVM* model reports much better results, which suggests that training a discriminative model using natural photos, and automatically generated image composites can help learn better feature representation for predicting visual realism.

Human Performance Judging an image as photo-realistic or not can be ambiguous even for humans. To measure the human performance on this task, we collected additional annotations for the 719 images in [15] using Amazon Mechanical Turk. We collected on average 13 annotations for each image by asking a simple question "Does this image look realistic?" and allowing the worker to choose one of four options: 1 (definitely unrealistic), 2 (probably unrealistic), 3 (probably realistic) and 4 (definitely realistic). We then average the scores of human response and compare the MT workers' ratings to the "ground truth" labels provided in the original dataset [15]. Humans achieve a score of 0.91 in terms of area under ROC curve, suggesting our model achieves performance that is close to level of human agreement on this dataset.

	RealismCNN	RealismCNN + SVM
FullySupervised	0.84	0.88
PartiallySupervised	0.79	0.84
Unsupervised	0.78	0.84

Table 2: Area under ROC curve comparing different dataset generation procedures. *FullySupervised* uses annotated objects for both source object and target object. *PartiallySupervised* uses annotated objects only for target object, but using object proposals for source object. *Unsupervised* uses object proposals for both cases.

Dataset Generation Procedure The CNN we reported so far was trained on the image composites generated by the *FullySupervised* procedure. In Table 2, we further compare the realism prediction performance when training with other procedures described in Section 3.1. We find that *FullySupervised RealismCNN* gives better results when no human realism labeling is available. With SVM supervised training (using human annotations), the margin between dif-

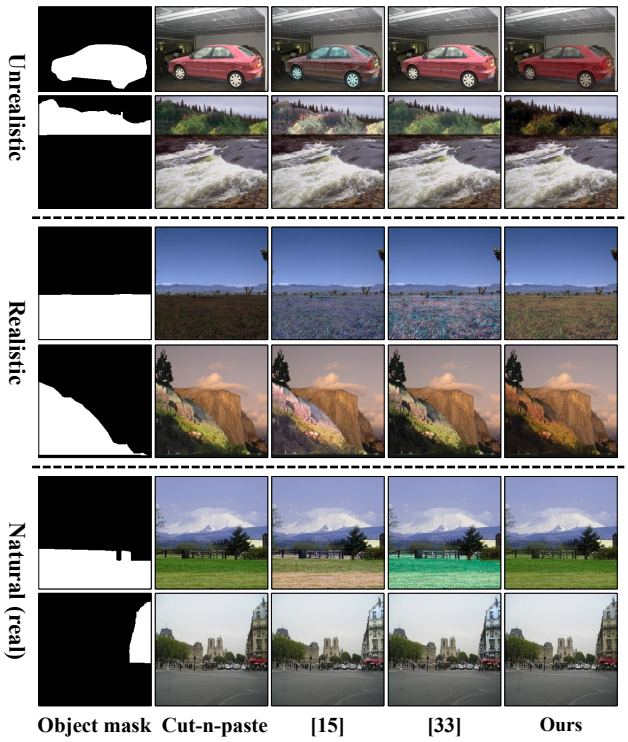

Figure 7: Example composite results: from left to right: objects mask, cut-and-paste, Lalonde and Efros [15], Xue et al. [33] and our method.

ferent dataset generation methods becomes smaller. This suggests that we can learn the feature representation using fully unsupervised data (without any masks), and improve it using small amounts of human rating annotations.

Indoor Scenes The Lalonde and Efros dataset [15] contains mainly photographs of natural outdoor environments. To complement this dataset, we construct a new dataset that contains 720 indoor photos with man-made objects from the LabelMe dataset. Similar to [15], our new dataset contains 180 natural photos, 180 realistic composites, and 360 unrealistic composites. To better model indoor scenes, we train our CNN model on $\sim 21,000$ natural images (both indoor and outdoor) that contain $\sim 42,000$ object instances from more than 200 categories of objects in the LabelMe dataset. We use MTurk to collect human labels for realistic and unrealistic composites (13 annotations per image). Without SVM training, our $RealismCNN$ alone achieves 0.83 on the indoor dataset, which is consistent with our results on the Lalonde and Efros dataset.

6.1. Optimizing Color Compatibility

Generating a realistic composite is a challenging problem. Here we show how our model can recolor the object so that it better fits the background.

Dataset, Baselines and Evaluation We use the dataset from [15] that provides a foreground object, its mask, and a background image for each photo. Given an input, we recolor the foreground object using four methods: simple cut-and-paste, Lalonde and Efros [15], Xue et al. [33] and our color adjustment model described in Section 4. We use the *FullySupervised* version of *RealismCNN* model without SVM training. We follow the same evaluation setting as in [33] and use Amazon Mechanical Turk to collect pairwise comparisons between pairs of results (the question we ask is "Given two photos generated by two different methods, which photo looks more realistic?"). We collected in total 43140 pairwise annotations (10 annotations for each pair of methods for all 719 images). We use the Thurstone's Case V Model [32] to obtain a realism score for each method per image from the pairwise annotations, and normalize the scores so that their standard deviation for each image is 1. Finally, we compute the average scores over all the photos. We report these average human rating scores for three categories of images: unrealistic composites, realistic composites and natural photos. We use natural photos for sanity check since an ideal color adjustment algorithm should not modify the color distribution of an object in a natural photo. For natural photos, if no color adjustment is applied, the "cut-and-paste" result does not alter the original photo.

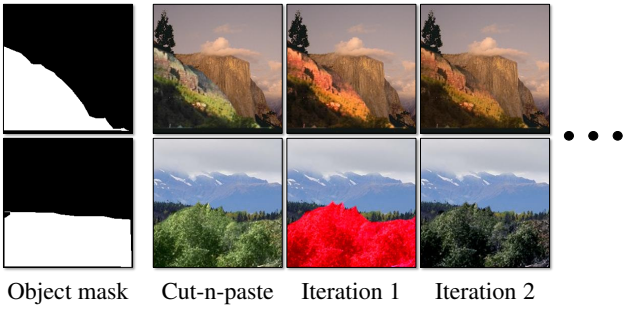

Figure 8: From left to right: object mask, cut-and-paste, results generated by $CNNIter1$ and $CNNIter2$ without the regularization term E_{reg}.

Results Table 3 compares different methods in terms of average human ratings. On average, our method outperforms other existing color adjustment methods. Our method significantly improves the visual realism of unrealistic photos. Interestingly, none of the methods can notably improve realistic composites although our model still performs best among the three color adjustment methods. Having a sense of visual realism informs our color adjustment model as to when, and how much, it should recolor the object. For both realistic composites and natural photos, our method typically does not change much the color distribution since these images are correctly predicted as already being quite realistic. On the other hand, the other two methods try to always match the low-level statistics between the foreground object and background, regardless of how realistic the photo is before recoloring. Figure 7 shows some example results.

Hard Negative Mining We observe that our color optimization method performs poorly for some images once we turn off the regularization term E_{reg}. (See Figure 8 for examples). We think this is because some of the resulting colors (without E_{reg}) never appear in any training data (positive or negative). To avoid this unsatisfactory property, we add newly generated color adjustment results as the negative data, and retrain the CNN with newly added data, similar to hard negative mining in object detection literature [7]. Then we use this new CNN model to recolor the object again. We repeat this process three times, and obtain three CNN models named as $CNNIter1$, $CNNIter2$ and $CNNIter3$. We compare these three models (with E_{reg} added back) using the same MTurk experiment setup, and obtain the following results: $CNNIter1$: -0.162, $CNNIter2$: 0.045, and $CNNIter3$: 0.117. As shown in Figure 8, the hard negative mining avoids extreme coloring, and produces better results in general. We use $CNNIter3$ with E_{reg} to produce the final results in Table 3 and Figure 7.

6.2. Selecting Suitable Object

We can also use our *RealismCNN* model to select the best-fitting object from a database given a location and a background image. In particular, we generate multiple possible candidate composites for one category (e.g. a car) and use our model to select the most realistic one among them.

We randomly select 50 images from each of the 15 largest object categories in the LabelMe dataset and build a dataset of 750 background images. For each background photo, we generate 25 candidate composite images by finding 25 source objects (from all other objects in the same category) with the most similar shapes to the target object, as described in Section 3.1. Then the task is to pick the object that fits the background best. We select the foreground object using three methods: using *RealismCNN*, as described in Section 4; select the object with the most similar shape (denoted *Shape*); and randomly select the object from 25 candidates (denoted *Random*).

We follow the same evaluation setting described in Section 6.1. We collect 22500 human annotations, and obtain the following average Human ratings: *RealismCNN*: 0.285, *Shape*: -0.033, and *Random*: -0.252. Figure 9 shows some example results for the different methods. Our method can suggest more suitable objects for the composition task.

7. Conclusion

In this paper, we present a learning approach for characterizing the space of natural images, using a large dataset of automatically created image composites. We show that our learned model can predict whether a given image composite will be perceived as realistic or not by a human observer.

(a) Best-fitting object selected by *RealismCNN*

(b) Object with most similar shape

(c) Random selected objects

Figure 9: For the same photo and the same location, we produce different composites using objects selected by three methods: (a) *RealismCNN*, (b) the object with the most similar shape, and (c) a randomly selected object.

	Unrealistic Composites	Realistic Composites	Natural Photos
cut-and-paste	-0.024	0.263	0.287
[15]	0.123	-0.299	-0.247
[33]	-0.410	-0.242	-0.237
ours	0.311	0.279	0.196

Table 3: Comparison of methods for improving composites by average human ratings. We use the authors' code to produce results for Lalonde and Efros [15] and Xue et al [33]. We follow the same evaluation setting as in [33] and obtain human ratings from pairwise comparisons using Thurstone's Case V Model [32].

Our model can also guide automatic color adjustment and object selection for image compositing.

Many factors play a role in the perception of realism. While our learned model mainly picks up on purely visual cues such as color compatibility, lighting consistency, and segment compatibility, high-level scene cues (semantics, scene layout, perspective) are also important factors. Our current model is not capable of capturing these cues as we generate composites by replacing the object with an object from the same category and with a similar shape. Further investigation in these high-level cues will be required.

Acknowledgements We thank Jean-François Lalonde and Xue Su for help with running their code. This work was sponsored in part by ONR MURI N000141010934, an Adobe research grant, a NVIDIA hardware grant and an Intel research grant. J.-Y. Zhu was supported by Facebook Graduate Fellowship.

References

[1] P. J. Burt and E. H. Adelson. A multiresolution spline with application to image mosaics. *ACM Trans. on Graphics*, 2(4):217–236, 1983. 2

[2] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995. 5

[3] C.-C. Chang and C.-J. Lin. Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):27, 2011. 5

[4] S. Darabi, E. Shechtman, C. Barnes, D. B. Goldman, and P. Sen. Image melding: combining inconsistent images using patch-based synthesis. *ACM Trans. on Graphics*, 31(4), 2012. 2

[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*, pages 248–255. IEEE, 2009. 3, 5

[6] M. Elad and M. Aharon. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Trans. Image Process*, 15(12):3736–3745, 2006. 2

[7] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *PAMI*, 32(9):1627–1645, 2010. 8

[8] J. Hays and A. A. Efros. Scene completion using millions of photographs. *ACM Trans. on Graphics*, 26(3), 2007. 2

[9] Y. Hel-Or and D. Shaked. A discriminative approach for wavelet denoising. *IEEE Trans. Image Process*, 17(4):443–457, 2008. 2

[10] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM Multimedia*, pages 675–678, 2014. 3

[11] M. K. Johnson, K. Dale, S. Avidan, H. Pfister, W. T. Freeman, and W. Matusik. Cg2real: Improving the realism of computer generated images using a large collection of photographs. *IEEE Trans. Vis. Comput. Graphics*, 17(9):1273–1285, 2011. 2

[12] E. Kee, J. F. O'brien, and H. Farid. Exposing photo manipulation from shading and shadows. *ACM Trans. on Graphics*, 33(5):165, 2014. 2

[13] P. Krähenbühl and V. Koltun. Geodesic object proposals. In *ECCV*, pages 725–739. 2014. 3

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012. 5, 6

[15] J.-F. Lalonde and A. A. Efros. Using color compatibility for assessing image realism. In *ICCV*, pages 1–8, 2007. 1, 2, 3, 4, 5, 6, 7, 8

[16] J.-F. Lalonde, D. Hoiem, A. A. Efros, C. Rother, J. Winn, and A. Criminisi. Photo clip art. In *ACM Trans. on Graphics*, volume 26, page 3, 2007. 2, 5

[17] Y. Liu, J. Wang, S. Cho, A. Finkelstein, and S. Rusinkiewicz. A no-reference metric for evaluating the quality of motion deblurring. *ACM Trans. on Graphics*, 32(6):175, 2013. 2

[18] M. Mori, K. F. MacDorman, and N. Kageki. The uncanny valley [from the field]. *Robotics & Automation Magazine, IEEE*, 19(2):98–100, 2012. 1

[19] A. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *arXiv preprint arXiv:1412.1897*, 2014. 1

[20] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. In *ACM Trans. on Graphics*, volume 22, pages 313–318. ACM, 2003. 2

[21] A. C. Popescu and H. Farid. Exposing digital forgeries by detecting traces of resampling. *Signal Processing, IEEE Transactions on*, 53(2):758–767, 2005. 2

[22] J. Portilla, V. Strela, M. J. Wainwright, and E. P. Simoncelli. Image denoising using scale mixtures of gaussians in the wavelet domain. *IEEE Trans. Image Process*, 12:1338–1351, 2003. 2

[23] E. Reinhard, A. O. Akuyz, M. Colbert, C. E. Hughes, and M. OConnor. Real-time color blending of rendered and captured video. In *Interservice/Industry Training, Simulation and Education Conference*, volume 18, 2004. 2, 5

[24] E. Reinhard, M. Ashikhmin, B. Gooch, and P. Shirley. Color transfer between images. *IEEE Comput. Graph. Appl.*, 21(5):34–41, Sept. 2001. 2

[25] L. Ren, A. Patrick, A. A. Efros, J. K. Hodgins, and J. M. Rehg. A data-driven approach to quantifying natural human motion. *ACM Trans. on Graphics*, 24(3):1090–1097, 2005. 2

[26] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: a database and web-based tool for image annotation. *IJCV*, 77(1-3):157–173, 2008. 3, 5

[27] U. Schmidt, C. Rother, S. Nowozin, J. Jancsary, and S. Roth. Discriminative non-blind deblurring. In *CVPR*, June 2013. 2

[28] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3, 5, 6

[29] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 1

[30] M. Tan, J.-F. Lalonde, L. Sharan, H. Rushmeier, and C. O'sullivan. The perception of lighting inconsistencies in composite outdoor scenes. *ACM Trans. Appl. Percept.*, 12(4):18:1–18:18, Sept. 2015. 2

[31] M. W. Tao, M. K. Johnson, and S. Paris. Error-tolerant image compositing. *IJCV*, 103(2):178–189, 2013. 2

[32] K. Tsukida and M. R. Gupta. How to analyze paired comparison data. Technical report, DTIC Document, 2011. 7, 8

[33] S. Xue, A. Agarwala, J. Dorsey, and H. Rushmeier. Understanding and improving the realism of image composites. *ACM Trans. on Graphics*, 31(4):84, 2012. 2, 7, 8

[34] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *NIPS*, pages 487–495, 2014. 5, 6

[35] D. Zoran and Y. Weiss. From learning models of natural image patches to whole image restoration. In *ICCV*, pages 479–486, 2011. 2