Material Recognition in the Wild with the Materials in Context Database

Sean Bell* Paul Upchurch* Noah Snavely Kavita Bala
Department of Computer Science, Cornell University
{sbell,paulu,snavely,kb}@cs.cornell.edu

Abstract

Recognizing materials in real-world images is a challenging task. Real-world materials have rich surface texture, geometry, lighting conditions, and clutter, which combine to make the problem particularly difficult. In this paper, we introduce a new, large-scale, open dataset of materials in the wild, the ***Materials in Context Database*** *(MINC), and combine this dataset with deep learning to achieve material recognition and segmentation of images in the wild.*

MINC is an order of magnitude larger than previous material databases, while being more diverse and well-sampled across its 23 categories. Using MINC, we train convolutional neural networks (CNNs) for two tasks: classifying materials from patches, and simultaneous material recognition and segmentation in full images. For patch-based classification on MINC we found that the best performing CNN architectures can achieve 85.2% mean class accuracy. We convert these trained CNN classifiers into an efficient fully convolutional framework combined with a fully connected conditional random field (CRF) to predict the material at every pixel in an image, achieving 73.1% mean class accuracy. Our experiments demonstrate that having a large, well-sampled dataset such as MINC is crucial for real-world material recognition and segmentation.

1. Introduction

Material recognition plays a critical role in our understanding of and interactions with the world. To tell whether a surface is easy to walk on, or what kind of grip to use to pick up an object, we must recognize the materials that make up our surroundings. Automatic material recognition can be useful in a variety of applications, including robotics, product search, and image editing for interior design. But recognizing materials in real-world images is very challenging. Many categories of materials, such as fabric or wood, are visually very rich and span a diverse range of appearances. Materials can further vary in appearance due to lighting and shape. Some categories, such as plastic and ceramic, are often smooth and featureless, requiring reasoning about subtle cues or context to differentiate between them.

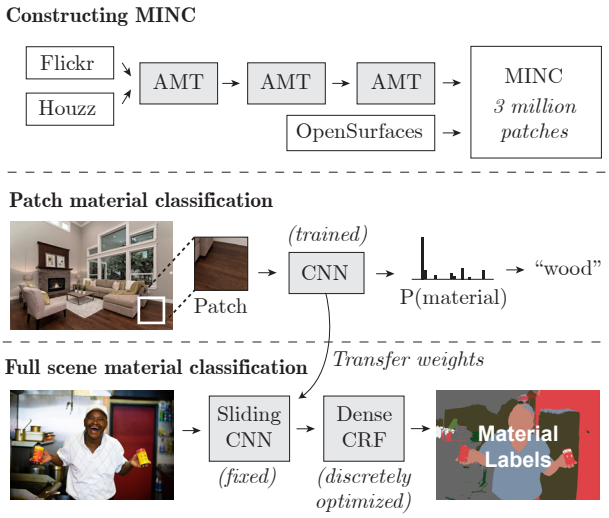


Figure 1. **Overview. (a)** We construct a new dataset by combining OpenSurfaces [1] with a novel three-stage Amazon Mechanical Turk (AMT) pipeline. **(b)** We train various CNNs on patches from MINC to predict material labels. **(c)** We transfer the weights to a fully convolutional CNN to efficiently generate a probability map across the image; we then use a fully connected CRF to predict the material at every pixel.

Large-scale datasets (e.g., ImageNet [21], SUN [31, 19] and Places [34]) combined with convolutional neural networks (CNNs) have been key to recent breakthroughs in object recognition and scene classification. Material recognition is similarly poised for advancement through large-scale data and learning. To date, progress in material recognition has been facilitated by moderate-sized datasets like the Flickr Material Database (FMD) [26]. FMD contains ten material categories, each with 100 samples drawn from Flickr photos. These images were carefully selected to illustrate a wide range of appearances for these categories. FMD has been used in research on new features and learning methods for material perception and recognition [17, 10, 20, 25]. While FMD was an important step towards material recognition, it is not sufficient for classifying materials in real-world im-

*Authors contributed equally

agery. This is due to the relatively small set of categories, the relatively small number of images per category, and also because the dataset has been designed around hand-picked iconic images of materials. The OpenSurfaces dataset [1] addresses some of these problems by introducing 105,000 material segmentations from real-world images, and is significantly larger than FMD. However, in OpenSurfaces many material categories are under-sampled, with only tens of images.

A major contribution of our paper is a new, well-sampled material dataset, called the **Materials in Context Database** (MINC), with **3 million** material samples. MINC is more diverse, has more examples in less common categories, and is much larger than existing datasets. MINC draws data from both Flickr images, which include many "regular" scenes, as well as Houzz images from professional photographers of staged interiors. These sources of images each have different characteristics that together increase the range of materials that can be recognized. See Figure 2 for examples of our data. We make our full dataset available online at http://minc.cs.cornell.edu/.

We use this data for material recognition by training different CNN architectures on this new dataset. We perform experiments that illustrate the effect of network architecture, image context, and training data size on subregions (i.e., patches) of a full scene image. Further, we build on our patch classification results and demonstrate simultaneous material recognition and segmentation of an image by performing dense classification over the image with a fully connected conditional random field (CRF) model [12]. By replacing the fully connected layers of the CNN with convolutional layers [24], the computational burden is significantly lower than a naive sliding window approach.

In summary, we make two new contributions:

- We introduce a new material dataset, MINC, and 3-stage crowdsourcing pipeline for efficiently collecting millions of click labels (Section 3.2).
- Our new semantic segmentation method combines a fully-connected CRF with unary predictions based on CNN learned features (Section 4.2) for simultaneous material recognition and segmentation.

2. Prior Work

Material Databases. Much of the early work on material recognition focused on classifying specific instances of textures or material samples. For instance, the CUReT [4] database contains 61 material samples, each captured under 205 different lighting and viewing conditions. This led to research on the task of instance-level texture or material classification [15, 30], and an appreciation of the challenges of building features that are invariant to pose and illumination. Later, databases with more diverse examples from each material category began to appear, such as KTH-TIPS [9, 2], and led explorations of how to generalize from one example of a material to another—from one sample of wood to a completely different sample, for instance. Real-world texture attributes have also recently been explored [3].

In the domain of categorical material databases, Sharan *et al*. released FMD [26] (described above). Subsequently, Bell *et al*. released OpenSurfaces [1] which contains over 20,000 real-world scenes labeled with both materials and objects, using a multi-stage crowdsourcing pipeline. Because OpenSurfaces images are drawn from consumer photos on Flickr, material samples have real-world context, in contrast to prior databases (CUReT, KTH-TIPS, FMD) which feature cropped stand-alone samples. While OpenSurfaces is a good starting point for a material database, we substantially expand it with millions of new labels.

Material recognition. Much prior work on material recognition has focused on the classification problem (categorizing an image patch into a set of material categories), often using hand-designed image features. For FMD, Liu *et al*. [17] introduced reflectance-based edge features in conjunction with other general image features. Hu *et al*. [10] proposed features based on variances of oriented gradients. Qi *et al*. [20] introduced a pairwise local binary pattern (LBP) feature. Li *et al*. [16] synthesized a dataset based on KTH-TIPS2 and built a classifier from LBP and dense SIFT. Timofte *et al*. [29] proposed a classification framework with minimal parameter optimization. Schwartz and Nishino [23] introduced material traits that incorporate learned convolutional auto-encoder features. Recently, Cimpoi *et al*. [3] developed a CNN and improved Fisher vector (IFV) classifier that achieves state-of-the-art results on FMD and KTH-TIPS2. Finally, it has been shown that jointly predicting objects and materials can improve performance [10, 33].

Convolutional neural networks. While CNNs have been around for a few decades, with early successes such as LeNet [14], they have only recently led to state-of-the-art results in object classification and detection, leading to enormous progress. Driven by the ILSVRC challenge [21], we have seen many successful CNN architectures [32, 24, 28, 27], led by the work of Krizhevsky *et al*. on their SuperVision (a.k.a. AlexNet) network [13], with more recent architectures including GoogLeNet [28]. In addition to image classification, CNNs are the state-of-the-art for detection and localization of objects, with recent work including R-CNNs [7], Overfeat [24], and VGG [27]. Finally, relevant to our goal of per-pixel material segmentation, Farabet *et al*. [6] use a multi-scale CNN to predict the class at every pixel in a segmentation. Oquab *et al*. [18] employ a sliding window approach to localize patch classification of objects. We build on this body of work in deep learning to solve our problem of material recognition and segmentation.

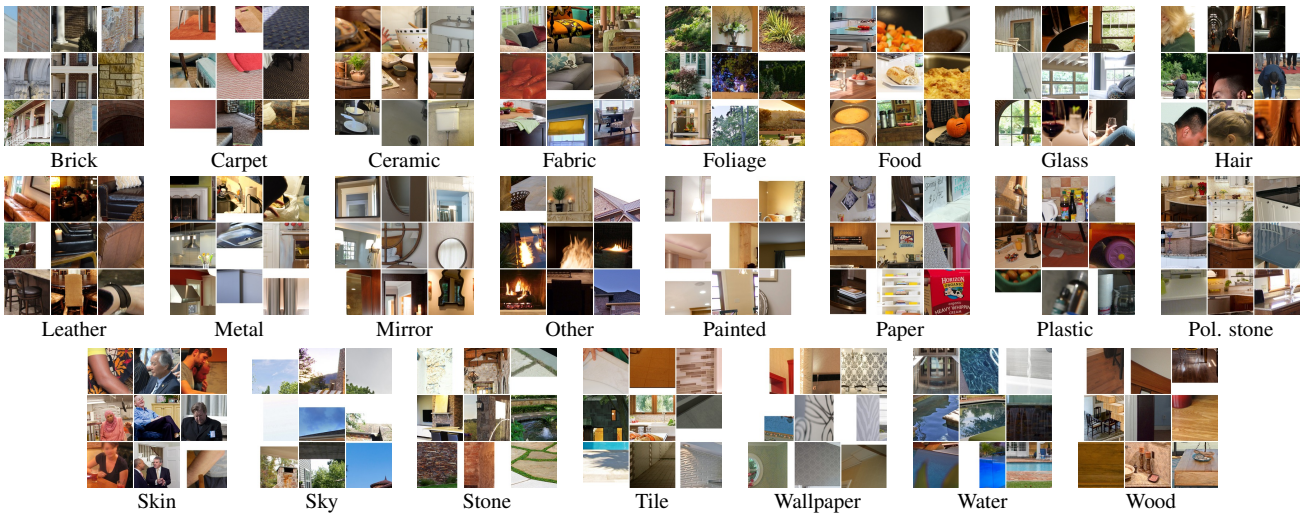


Figure 2. Example patches from all 23 categories of the **Materials in Context Database** (MINC). Note that we sample patches so that the patch center is the material in question (and not necessarily the entire patch). See Table 1 for the size of each category.

3. The Materials in Context Database (MINC)

We now describe the methodology that went into building our new material database. Why a new database? We needed a dataset with the following properties:

- **Size**: It should be sufficiently large that learning methods can generalize beyond the training set.
- **Well-sampled**: Rare categories should be represented with a large number of examples.
- **Diversity**: Images should span a wide range of appearances of each material in real-world settings.
- **Number of categories**: It should contain many different materials found in the real world.

3.1. Sources of data

We decided to start with the public, crowdsourced OpenSurfaces dataset [1] as the seed for MINC since it is drawn from Flickr imagery of everyday, real-world scenes with reasonable diversity. Furthermore, it has a large number of categories and the most samples of all prior databases.

While OpenSurfaces data is a good start, it has a few limitations. Many categories in OpenSurfaces are not well sampled. While the largest category, *wood*, has nearly 20K samples, smaller categories, such as *water*, have only tens of examples. This imbalance is due to the way the OpenSurfaces dataset was annotated; workers on Amazon Mechanical Turk (AMT) were free to choose any material subregion to segment. Workers often gravitated towards certain common types of materials or salient objects, rather than being encouraged to label a diverse set of materials. Further, the images come from a single source (Flickr).

We decided to augment OpenSurfaces with substantially more data, especially for underrepresented material categories, with the initial goal of gathering at least 10K samples per material category. We decided to gather this data from another source of imagery, professional photos on the interior design website Houzz (houzz.com). Our motivation for using this different source of data was that, despite Houzz photos being more "staged" (relative to Flickr photos), they actually represent a larger variety of materials. For instance, Houzz photos contain a wide range of types of polished stone. With these sources of image data, we now describe how we gather material annotations.

3.2. Segments, Clicks, and Patches

What specific kinds of material annotations make for a good database? How should we collect these annotations? The type of annotations to collect is guided in large part by the tasks we wish to generate training data for. For some tasks such as scene recognition, whole-image labels can suffice [31, 34]. For object detection, labeled bounding boxes as in PASCAL are often used [5]. For segmentation or scene parsing tasks, per-pixel segmentations are required [22, 8]. Each style of annotation comes with a cost proportional to its complexity. For materials, we decided to focus on two problems, guided by prior work:

- **Patch material classification.** Given an image patch, what kind of material is it at the center?
- **Full scene material classification.** Given a full image, produce a full per-pixel segmentation and labeling. Also known as *semantic segmentation* or *scene parsing* (but in our case, focused on materials). Note that classification can be a component of segmentation, e.g., with sliding window approaches.

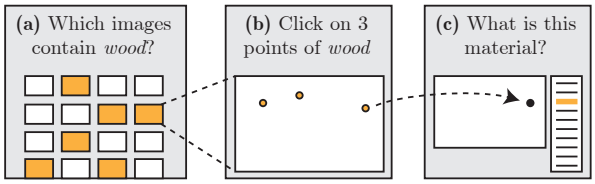


Figure 3. **AMT pipeline schematic for collecting clicks. (a)** Workers filter by images that contain a certain material, **(b)** workers click on materials, and **(c)** workers validate click locations by re-labeling each point. Example responses are shown in orange.

Segments. OpenSurfaces contains material segmentations—carefully drawn polygons that enclose same-material regions. To form the basis of MINC, we selected OpenSurfaces segments with high confidence (inter-worker agreement) and manually curated segments with low confidence, giving a total of 72K shapes. To better balance the categories, we manually segmented a few hundred extra samples for *sky*, *foliage* and *water*.

Since some of the OpenSurfaces categories are difficult for humans, we consolidated these categories. We found that many AMT workers could not disambiguate *stone* from *concrete*, *clear plastic* from *opaque plastic*, and *granite* from *marble*. Therefore, we merged these into *stone*, *plastic*, and *polished stone* respectively. Without this merging, many ground truth examples in these categories would be incorrect. The final list of 23 categories is shown in Table 1. The category *other* is different in that it was created by combining various smaller categories.

Clicks. Since we want to expand our dataset to millions of samples, we decided to augment OpenSurfaces segments by collecting *clicks*: single points in an image along with a material label, which are much cheaper and faster to collect. Figure 3 shows our pipeline for collecting clicks.

Initially, we tried asking workers to click on examples of a given material in a photo. However, we found that workers would get frustrated if the material was absent in too many of the photos. Thus, we added an initial first stage where workers filter out such photos. To increase the accuracy of our labels, we verify the click labels by asking different workers to specify the material for each click without providing them with the label from the previous stage.

To ensure that we obtain high quality annotations and avoid collecting labels from workers who are not making an effort, we include secret known answers (sentinels) in the first and third stages, and block workers with an accuracy below 50% and 85% respectively. We do not use sentinels in the second stage since it would require per-pixel ground truth labels, and it turned out not to be necessary. Workers generally performed all three tasks so we could identify bad workers in the first or third task.

Patches	Category	Patches	Category	Patches	Category
564,891	Wood	114,085	Polished stone	35,246	Skin
465,076	Painted	98,891	Carpet	29,616	Stone
397,982	Fabric	83,644	Leather	28,108	Ceramic
216,368	Glass	75,084	Mirror	26,103	Hair
188,491	Metal	64,454	Brick	25,498	Food
147,346	Tile	55,364	Water	23,779	Paper
142,150	Sky	39,612	Other	14,954	Wallpaper
120,957	Foliage	38,975	Plastic		

Table 1. **MINC patch counts by category.** Patches were created from both OpenSurfaces segments and our newly collected *clicks*. See Section 3.2 for details.

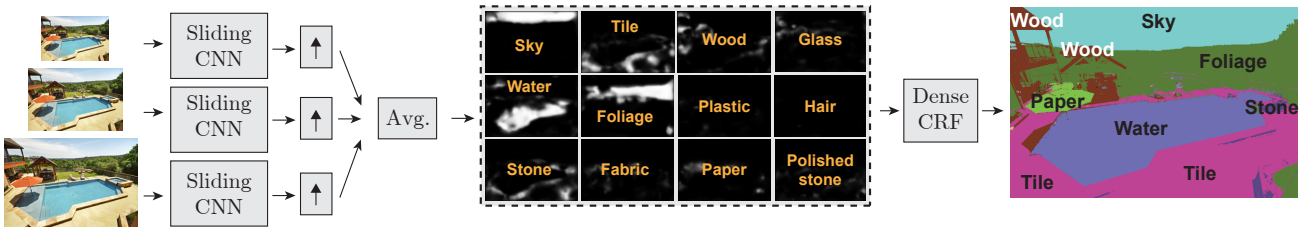
Material clicks were collected for both OpenSurfaces images and the new Houzz images. This allowed us to use labels from OpenSurfaces to generate the sentinel data; we included 4 sentinels per task. With this streamlined pipeline we collected 2,341,473 annotations at an average cost of $0.00306 per annotation (stage 1: $0.02 / 40 images, stage 2: $0.10 / 50 images, 2, stage 3: $0.10 / 50 points).

Patches. Labeled segments and clicks form the core of MINC. For training CNNs and other types of classifiers, it is useful to have data in the form of fixed-sized patches. We convert both forms of data into a unified dataset format: square image patches. We use a *patch center* and *patch scale* (a multiplier of the smaller image dimension) to define the image subregion that makes a patch. For our patch classification experiments, we use 23.3% of the smaller image dimension. Increasing the patch scale provides more context but reduces the spatial resolution. Later in Section 5 we justify our choice with experiments that vary the patch scale for AlexNet.

We place a patch centered around each click label. For each segment, if we were to place a patch at every interior pixel then we would have a very large and redundant dataset. Therefore, we Poisson-disk subsample each segment, separating patch centers by at least 9.1% of the smaller image dimension. These segments generated 655,201 patches (an average of 9.05 patches per segment). In total, we generated 2,996,674 labeled patches from 436,749 images. Patch counts are shown in Table 1, and example patches from various categories are illustrated in Figure 2.

4. Material recognition in real-world images

Our goal is to train a system that recognizes the material at every pixel in an image. We split our training procedure into multiple stages and analyze the performance of the network at each stage. First, we train a CNN that produces a single prediction for a given input patch. Then, we convert the CNN into a sliding window and predict materials on a dense grid across the image. We do this at multiple scales and average to obtain a unary term. Finally, a dense CRF [12] combines the unary term with fully connected pairwise reasoning to output per-pixel material predictions. The entire system is depicted in Figure 1, and described more below.



(a) Multiscale input **(b)** Probability map (12 of 23 shown) **(c)** Predicted labels

Figure 4. **Pipeline for full scene material classification.** An image **(a)** is resized to multiple scales $[1/\sqrt{2}, 1, \sqrt{2}]$. The same sliding CNN predicts a probability map **(b)** across the image for each scale; the results are upsampled and averaged. A fully connected CRF predicts a final label for each pixel **(c)**. This example shows predictions from a single GoogLeNet converted into a sliding CNN (no average pooling).

4.1. Training procedure

MINC contains 3 million patches that we split into training, validation and test sets. Randomly splitting would result in nearly identical patches (e.g., from the same OpenSurfaces segment) being put in training and test, thus inflating the test score. To prevent correlation, we group photos into clusters of near-duplicates, then assign each cluster to one of train, validate or test. We make sure that there are at least 75 segments of each category in the test set to ensure there are enough segments to evaluate segmentation accuracy. To detect near-duplicates, we compare AlexNet CNN features computed from each photo (see the supplemental for details). For exact duplicates, we discard all but one of the copies.

We train all of our CNNs by fine-tuning the network starting from the weights obtained by training on 1.2 million images from ImageNet (ILSVRC2012). When training AlexNet, we use stochastic gradient descent with batchsize 128, dropout rate 0.5, momentum 0.9, and a base learning rate of 10^{-3} that decreases by a factor of 0.25 every 50,000 iterations. For GoogLeNet, we use batchsize 69, dropout 0.4, and learning rate $\alpha_t = 10^{-4}\sqrt{1 - t/250000}$ for iteration t.

Our training set has a different number of examples per class, so we cycle through the classes and randomly sample an example from each class. Failing to properly balance the examples results in a 5.7% drop in mean class accuracy (on the validation set). Further, since it has been shown to reduce overfitting, we randomly augment samples by taking crops (227×227 out of 256×256), horizontal mirror flips, spatial scales in the range $[1/\sqrt{2}, \sqrt{2}]$, aspect ratios from 3:4 to 4:3, and amplitude shifts in $[0.95, 1.05]$. Since we are looking at local regions, we subtract a per-channel mean (R: 124, G: 117, B: 104) rather than a mean image [13].

4.2. Full scene material classification

Figure 4 shows an overview of our method for simultaneously segmenting and recognizing materials. Given a CNN that can classify individual points in the image, we convert it to a sliding window detector and densely classify a grid across the image. Specifically, we replace the last fully connected layers with convolutional layers, so that the network is fully convolutional and can classify images of any shape. After conversion, the weights are fixed and not fine-tuned. With our converted network, the strides of each layer cause the network to output a prediction every 32 pixels. We obtain predictions every 16 pixels by shifting the input image by half-strides (16 pixels). While this appears to require 4x the computation, Sermanet *et al.* [24] showed that the convolutions can be reused and only the pool5 through fc8 layers need to be recomputed for the half-stride shifts. Adding half-strides resulted in a minor 0.2% improvement in mean class accuracy across segments (after applying the dense CRF, described below), and about the same mean class accuracy at click locations.

The input image is resized so that a patch maps to a 256x256 square. Thus, for a network trained at patch scale s, the resized input has smaller dimension $d = 256/s$. Note that d is inversely proportional to scale, so increased context leads to lower spatial resolution. We then add padding so that the output probability map is aligned with the input when upsampled. We repeat this at 3 different scales (smaller dimension $d/\sqrt{2}$, d, $d\sqrt{2}$), upsample each output probability map with bilinear interpolation, and average the predictions. To make the next step more efficient, we upsample the output to a fixed smaller dimension of 550.

We then use the dense CRF of Krähenbühl *et al.* [12] to predict a label at every pixel, using the following energy:

$$E(x \mid \mathbf{I}) = \sum_i \psi_i(x_i) + \sum_{i<j} \psi_{ij}(x_i, x_j) \quad (1)$$

$$\psi_i(x_i) = -\log p_i(x_i) \quad (2)$$

$$\psi_{ij}(x_i, x_j) = w_p\, \delta(x_i \neq x_j)\, k(\mathbf{f}_i - \mathbf{f}_j) \quad (3)$$

where ψ_i is the unary energy (negative log of the aggregated softmax probabilities) and ψ_{ij} is the pairwise term that connects every pair of pixels in the image. We use a single pairwise term with a Potts label compatibility term δ weighted by w_p and unit Gaussian kernel k. For the features $\mathbf{f}_i$, we convert the RGB image to $L^*a^*b^*$ and use color (I_i^L, I_i^a, I_i^b) and position (p^x, p^y) as pairwise features for each pixel: $\mathbf{f}_i = \left[\frac{p_i^x}{\theta_p d}, \frac{p_i^y}{\theta_p d}, \frac{I_i^L}{\theta_L}, \frac{I_i^a}{\theta_{ab}}, \frac{I_i^b}{\theta_{ab}}\right]$, where d is the smaller image dimension. Figure 4 shows an example unary term p_i and the resulting segmentation x.

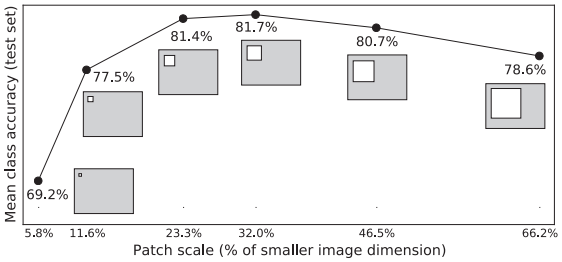


Figure 5. **Varying patch scale**. We train/test patches of different scales (the patch locations do not vary). The optimum is a trade-off between context and spatial resolution. CNN: AlexNet.

Architecture	Validation	Test
AlexNet [13]	82.2%	81.4%
GoogLeNet [28]	**85.9%**	**85.2%**
VGG-16 [27]	85.6%	84.8%

Table 2. **Patch material classification results.** Mean class accuracy for different CNNs trained on MINC. See Section 5.1.

Sky 97.3%	Food 90.4%	Wallpaper 83.4%	Glass 78.5%
Hair 95.3%	Leather 88.2%	Tile 82.7%	Fabric 77.8%
Foliage 95.1%	Other 87.9%	Ceramic 82.7%	Metal 77.7%
Skin 93.9%	Pol. stone 85.8%	Stone 82.7%	Mirror 72.0%
Water 93.6%	Brick 85.1%	Paper 81.8%	Plastic 70.9%
Carpet 91.6%	Painted 84.2%	Wood 81.3%	

Table 3. **Patch test accuracy by category.** CNN: GoogLeNet. See the supplemental material for a full confusion matrix.

5. Experiments and Results

5.1. Patch material classification

In this section, we evaluate the effect of many different design decisions for training methods for material classification and segmentation, including various CNN architectures, patch sizes, and amounts of data.

CNN Architectures. Our ultimate goal is full material segmentation, but we are also interested in exploring which CNN architectures give the best results for classifying single patches. Among the networks and parameter variations we tried we found the best performing networks were AlexNet [13], VGG-16 [27] and GoogLeNet [28]. AlexNet and GoogLeNet are re-implementations by BVLC [11], and VGG-16 is configuration D (a 16 layer network) of [27]. All models were obtained from the Caffe Model Zoo [11]. Our experiments use AlexNet for evaluating material classification design decisions and combinations of AlexNet and GoogLeNet for evaluating material segmentation. Tables 2 and 3 summarize patch material classification results on our dataset. Figure 10 shows correct and incorrect predictions made with high confidence.

Input patch scale. To classify a point in an image we must decide how much context to include around it. The context, expressed as a fraction of image size, is the patch scale. A priori, it is not clear which scale is best since small patches have better spatial resolution, but large patches have more

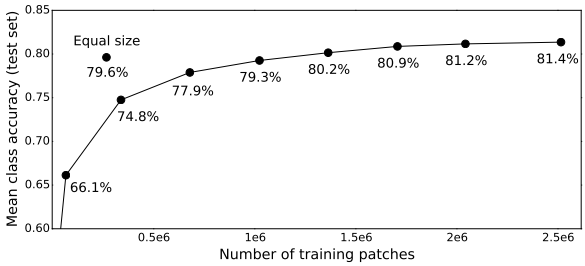


Figure 6. **Varying database size.** Patch accuracy when trained on random subsets of MINC. *Equal size* is using equal samples per category (size determined by smallest category). CNN: AlexNet.

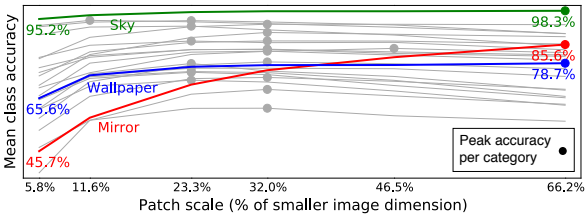


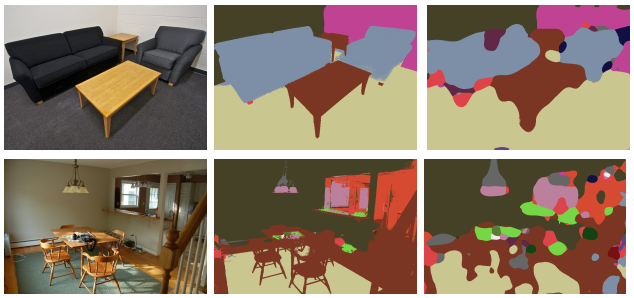
Figure 7. **Accuracy vs patch scale by category**. Dots: peak accuracy for each category; colored lines: *sky*, *wallpaper*, *mirror*; gray lines: other categories. CNN: AlexNet. While most materials are optimally recognized at 23.3% or 32% patch scale, recognition of *sky*, *wallpaper* and *mirror* improve with increasing context.

contextual information. Holding patch centers fixed we varied scale and evaluated classification accuracy with AlexNet. Results and a visualization of patch scales are shown in Figure 5. Scale 32% performs the best. Individual categories had peaks at middle scale with some exceptions; we find that *mirror*, *wallpaper* and *sky* improve with increasing context (Figure 7). We used 23.3% (which has nearly the same accuracy but higher spatial resolution) for our experiments.

Dataset size. To measure the effect of size on patch classification accuracy we trained AlexNet with patches from randomly sampled subsets of all 369,104 training images and tested on our full test set (Figure 6). As expected, using more data improved performance. In addition, we still have not saturated performance with 2.5 million training patches; even higher accuracies may be possible with more training data (though with diminishing returns).

Dataset balance. Although we've shown that more data is better we also find that a balanced dataset is more effective. We trained AlexNet with all patches of our smallest category (*wallpaper*) and randomly sampled the larger categories (the largest, *wood*, being 40x larger) to be equal size. We then measured mean class accuracy on the same full test set. As shown in Figure 6, "Equal size" is more accurate than a dataset of the same size and just 1.7% lower than the full training set (which is 9x larger). This result further demonstrates the value of building up datasets in a balanced manner, focusing on expanding the smallest, least common categories.

Figure 8. **Full-scene material classification examples:** high-accuracy test set predictions by our method. CNN: GoogLeNet (with the average pooling layer removed). Right: legend for material colors. See Table 4 for quantitative evaluation.



Input image (test set) | **(a)** Labels from CRF trained on segments | **(b)** Labels from CRF trained on clicks

Figure 9. **Optimizing for click accuracy leads to sloppy boundaries.** In **(a)**, we optimize for mean class accuracy across segments, resulting in high quality boundaries. In **(b)**, we optimize for mean class accuracy at click locations. Since the clicks are not necessarily close to object boundaries, there is no penalty for sloppy boundaries. CNN: GoogLeNet (without average pooling).

Architecture		**(a)** Segments only		**(b)** Clicks only	
		Class	Total	Class	Total
AlexNet	Scale: 11.6%	64.3%	72.6%	79.9%	77.2%
AlexNet	Scale: 23.3%	69.6%	76.6%	**83.3%**	**81.1%**
AlexNet	Scale: 32.0%	**70.1%**	**77.1%**	83.2%	80.7%
AlexNet	Scale: 46.5%	69.6%	75.4%	80.8%	77.7%
AlexNet	Scale: 66.2%	67.7%	72.0%	77.2%	72.6%
GoogLeNet	7x7 avg. pool	64.4%	71.6%	63.6%	63.4%
GoogLeNet	5x5 avg. pool	67.6%	74.6%	70.9%	69.8%
GoogLeNet	3x3 avg. pool	**70.4%**	77.7%	76.1%	74.7%
GoogLeNet	No avg. pool	**70.4%**	**78.8%**	**79.1%**	**77.4%**
Ensemble	2 CNNs	**73.1%**	**79.8%**	84.5%	83.1%
Ensemble	3 CNNs	**73.1%**	79.3%	**85.9%**	**83.5%**
Ensemble	4 CNNs	72.1%	78.4%	85.8%	83.2%
Ensemble	5 CNNs	71.7%	78.3%	85.5%	83.2%

Table 4. **Full scene material classification results.** Mean class and total accuracy on the test set. When training, we optimize the CRF parameters for mean class accuracy, but report both mean class and total accuracy (mean accuracy across all examples). In one experiment **(a)**, we train and test only on segments; in a separate experiment **(b)**, we train and test only on clicks. Accuracies for segments are averaged across all pixels that fall in that segment.

5.2. Full scene material segmentation

The full test set for our patch dataset contains 41,801 photos, but most of them contain only a few labels. Since we want to evaluate the per-pixel classification performance, we select a subset of 5,000 test photos such that each photo contains a large number of segments and clicks, and small categories are well sampled. We greedily solve for the best such set of photos. We similarly select 2,500 of 25,844 validation photos. Our splits for all experiments are included online with the dataset. To train the CRF for our model, we try various parameter settings (θ_p, θ_{ab}, θ_L, w_p) and select the model that performs best on the validation set. In total, we evaluate 1799 combinations of CNNs and CRF parameters. See the supplemental material for a detailed breakdown.

We evaluate multiple versions of GoogLeNet: both the original architecture and a version with the average pooling layer (at the end) changed to 5x5, 3x3, and 1x1 (i.e. no average pooling). We evaluate AlexNet trained at multiple patch scales (Figure 5). When using an AlexNet trained at a different scale, we keep the same scale for testing. We also experiment with ensembles of GoogLeNet and AlexNet, combined with either arithmetic or geometric mean.

Since we have two types of data, *clicks* and *segments*, we run two sets of experiments: (a) we train and test only on segments, and in a separate experiment (b) we train and test only on clicks. These two training objectives result in very different behavior, as illustrated in Figure 9. In experiment (a), the accuracy across segments are optimized, producing clean boundaries. In experiment (b), the CRF maximizes accuracy only at click locations, thus resulting in sloppy boundaries. As shown in Table 4, the numerical scores for the two experiments are also very different: segments are more challenging than clicks. While clicks are sufficient to train a CNN, they are not sufficient to train a CRF.

Focusing on segmentation accuracy, we see from Table 4(a) that our best single model is GoogLeNet without average pooling (6% better than with pooling). The best ensemble is 2 CNNs: GoogLeNet (no average pooling) and AlexNet (patch scale: 46.5%), combined with arithmetic mean. Larger ensembles perform worse since we are aver-

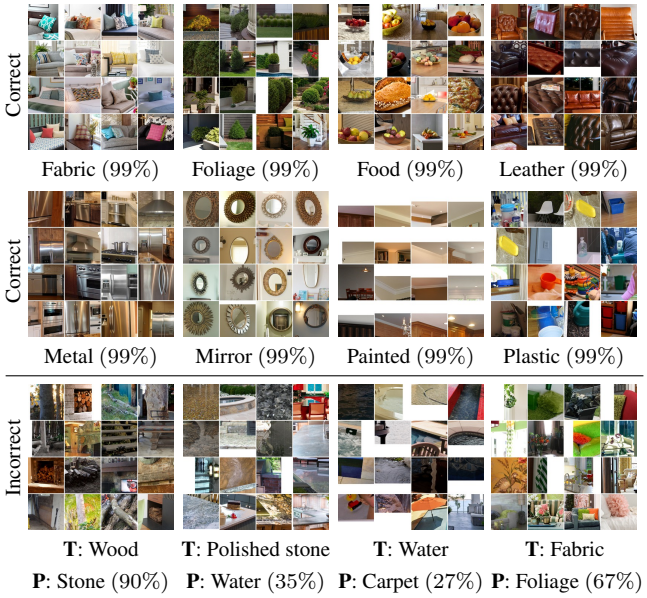


Figure 10. **High confidence predictions.** Top two rows: correct predictions. Bottom row: incorrect predictions (**T**: true, **P**: predicted). Percentages indicate confidence (the predictions shown are at least this confident). CNN: GoogLeNet.

aging worse CNNs. In Figure 8, we show example labeling results on test images.

5.3. Comparing MINC to FMD

Compared to FMD, the size and diversity of MINC is valuable for classifying real-world imagery. Table 5 shows the effect of training on all of FMD and testing on MINC (and vice versa). The results suggests that training on FMD alone is not sufficient for real-world classification. Though it may seem that our dataset is "easy," since the best classifications scores are lower for FMD than for MINC, we find that difficulty is in fact closely tied to dataset size (Section 5.1). Taking 100 random samples per category, AlexNet achieves $54.2 \pm 0.7\%$ on MINC ($64.6 \pm 1.3\%$ when considering only the 10 FMD categories) and 66.5% on FMD.

5.4. Comparing CNNs with prior methods

Cimpoi [3] is the best prior material classification method on FMD. We find that by replacing DeCAF with oversampled AlexNet features we can improve on their FMD results. We then show that on MINC, a finetuned CNN is even better.

To improve on [3], we take their SIFT_IFV, combine it with AlexNet fc7 features, and add oversampling [13] (see supplemental for details). With a linear SVM we achieve $69.6 \pm 0.3\%$ on FMD. Previous results are listed in Table 6.

Having found that SIFT_IFV+fc7 is the new best on FMD, we compare it to a finetuned CNN on a subset of MINC (2500 patches per category, one patch per photo). Fine-tuning AlexNet achieves $76.0 \pm 0.2\%$ whereas

		Test	
		FMD	MINC
Train	FMD	66.5%	26.1%
	MINC	41.7%	85.0%

(10 categories in common)

Table 5. **Cross-dataset experiments.** We train on one dataset and test on another dataset. Since MINC contains 23 categories, we limit MINC to the 10 categories in common. CNN: AlexNet.

Method	**Accuracy**	**Trials**
Sharan *et al.* [25]	$57.1 \pm 0.6\%$	14 splits
Cimpoi *et al.* [3]	$67.1 \pm 0.4\%$	14 splits
Fine-tuned AlexNet	$66.5 \pm 1.5\%$	5 folds
SIFT_IFV+fc7	$\mathbf{69.6 \pm 0.3\%}$	10 splits

Table 6. **FMD experiments.** By replacing DeCAF features with oversampled AlexNet features we improve on the best FMD result.

SIFT_IFV+fc7 achieves $67.4 \pm 0.5\%$ with a linear SVM (oversampling, 5 splits). This experiment shows that a finetuned CNN is a better method for MINC than SIFT_IFV+fc7.

6. Conclusion

Material recognition is a long-standing, challenging problem. We introduce a new large, open, material database, MINC, that includes a diverse range of materials of everyday scenes and staged designed interiors, and is at least an order of magnitude larger than prior databases. Using this large database we conduct an evaluation of recent deep learning algorithms for simultaneous material classification and segmentation, and achieve results that surpass prior attempts at material recognition.

Some lessons we have learned are:

- Training on a dataset which includes the surrounding context is crucial for real-world material classification.
- Labeled clicks are cheap and sufficient to train a CNN alone. However, to obtain high quality segmentation results, training a CRF on polygons results in much better boundaries than training on clicks.

Many future avenues of work remain. Expanding the dataset to a broader range of categories will require new ways to mine images that have more variety, and new annotation tasks that are cost-effective. Inspired by attributes for textures [3], in the future we would like to identify material attributes and expand our database to include them. We also believe that further exploration of joint material and object classification and segmentation will be fruitful [10] and lead to improvements in both tasks. Our database, trained models, and all experimental results are available online at http://minc.cs.cornell.edu/.

Acknowledgements. This work was supported in part by Google, Amazon AWS for Education, a NSERC PGS-D scholarship, the National Science Foundation (grants IIS-1149393, IIS-1011919, IIS-1161645), and the Intel Science and Technology Center for Visual Computing.

References

[1] S. Bell, P. Upchurch, N. Snavely, and K. Bala. OpenSurfaces: A richly annotated catalog of surface appearance. *ACM Trans. on Graphics (SIGGRAPH)*, 32(4), 2013. 1, 2, 3

[2] B. Caputo, E. Hayman, and P. Mallikarjuna. Class-specific material categorisation. In *ICCV*, pages 1597–1604, 2005. 2

[3] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Describing textures in the wild. In *CVPR*, pages 3606–3613. IEEE, 2014. 2, 8

[4] K. J. Dana, B. Van Ginneken, S. K. Nayar, and J. J. Koenderink. Reflectance and texture of real-world surfaces. *ACM Transactions on Graphics (TOG)*, 18(1):1–34, 1999. 2

[5] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *IJCV*, 88(2):303–338, June 2010. 3

[6] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *PAMI*, 35(8):1915–1929, 2013. 2

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 2

[8] S. Gould, R. Fulton, and D. Koller. Decomposing a scene into geometric and semantically consistent regions. In *ICCV*, 2009. 3

[9] E. Hayman, B. Caputo, M. Fritz, and J. olof Eklundh. On the significance of real-world conditions for material classification. In *ECCV*, 2004. 2

[10] D. Hu, L. Bo, and X. Ren. Toward robust material recognition for everyday objects. In *BMVC*, pages 1–11. Citeseer, 2011. 1, 2, 8

[11] 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 *Proceedings of the ACM International Conference on Multimedia*, pages 675–678. ACM, 2014. 6

[12] P. Krähenbühl and V. Koltun. Parameter learning and convergent inference for dense random fields. In *ICML*, pages 513–521, 2013. 2, 4, 5

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 2, 5, 6, 8

[14] 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*, 1(4):541–551, 1989. 2

[15] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *IJCV*, 43(1):29–44, June 2001. 2

[16] W. Li and M. Fritz. Recognizing materials from virtual examples. In *ECCV*, pages 345–358. Springer, 2012. 2

[17] C. Liu, L. Sharan, E. H. Adelson, and R. Rosenholtz. Exploring features in a bayesian framework for material recognition. In *CVPR*, pages 239–246. IEEE, 2010. 1, 2

[18] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *CVPR*, 2014. 2

[19] G. Patterson, C. Xu, H. Su, and J. Hays. The SUN Attribute Database: Beyond Categories for Deeper Scene Understanding. *IJCV*, 108(1-2):59–81, 2014. 1

[20] X. Qi, R. Xiao, J. Guo, and L. Zhang. Pairwise rotation invariant co-occurrence local binary pattern. In *ECCV*, pages 158–171. Springer, 2012. 1, 2

[21] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge, 2014. 1, 2

[22] 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, May 2008. 3

[23] G. Schwartz and K. Nishino. Visual material traits: Recognizing per-pixel material context. In *Proceedings of the International Conference on Computer Vision Workshops (ICCVW)*, pages 883–890. IEEE, 2013. 2

[24] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *International Conference on Learning Representations (ICLR2014)*. CBLS, April 2014. 2, 5

[25] L. Sharan, C. Liu, R. Rosenholtz, and E. Adelson. Recognizing materials using perceptually inspired features. *IJCV*, 2013. 1, 8

[26] L. Sharan, R. Rosenholtz, and E. Adelson. Material perception: What can you see in a brief glance? *Journal of Vision*, 9(8):784–784, 2009. 1, 2

[27] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2, 6

[28] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CVPR*, 2015. 2, 6

[29] R. Timofte and L. J. Van Gool. A training-free classification framework for textures, writers, and materials. In *BMVC*, pages 1–12, 2012. 2

[30] M. Varma and A. Zisserman. A statistical approach to texture classification from single images. *IJCV*, 62(1-2):61–81, Apr. 2005. 2

[31] J. Xiao, K. A. Ehinger, J. Hays, A. Torralba, and A. Oliva. SUN Database: Exploring a large collection of scene categories. *IJCV*, 2014. 1, 3

[32] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *ECCV*, pages 818–833. Springer, 2014. 2

[33] S. Zheng, M.-M. Cheng, J. Warrell, P. Sturgess, V. Vineet, C. Rother, and P. H. Torr. Dense semantic image segmentation with objects and attributes. In *CVPR*, pages 3214–3221. IEEE, 2014. 2

[34] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using Places database. *NIPS*, 2014. 1, 3