Large-scale Continual Road Inspection: Visual Infrastructure Assessment in the Wild

Ke Ma[1]
kemma@cs.stonybrook.edu
Minh Hoai[1]
minhhoai@cs.stonybrook.edu
Dimitris Samaras[12]
samaras@cs.stonybrook.edu

[1] Computer Vision Lab
Stony Brook University
Stony Brook, NY, USA

[2] CentraleSupélec.
Université Paris-Saclay
France

Abstract

This work develops a method to inspect the quality of pavement conditions based on images captured from moving vehicles. This task is challenging because the appearance of road surfaces varies tremendously, depending on the construction materials (e.g., concrete, asphalt), the weather conditions (e.g., rain, snow), the illumination conditions (e.g., sunny, shadow), and the interference of other structures (e.g., manholes, road marks). This problem is amplified by the lack of a sufficiently large and diverse dataset for training a pavement classifier. Our first contribution in this paper is the development of a method to create a large-scale dataset of pavement images. Specifically, using map and GPS information, we match the ratings by government inspectors found in public databases to Google Street View images, creating a dataset containing more than 700K images from 70K street segments. We use the dataset to develop a deep-learning method for road assessment, which is based on Convolutional Neural Networks, Fisher Vector encoding, and UnderBagging random forests. This method achieves an accuracy of 58.2% and significantly outperforms various other texture classification methods.

1 Introduction

This paper is motivated by the lack of a reliable method to tackle a challenging real-world problem: image-based road degradation assessment. Solving this problem is essential to improving road condition monitoring. Currently, this task is the responsibility of local and state maintenance departments, and they rely on pavement specialists or dedicated vehicles to physically visit and inspect damages. This approach is labor intensive and therefore cannot be performed regularly. In New York for example, state roads are only inspected about once a year. One possible solution is to crowdsource the monitoring task to road users. Unfortunately, road users are not trained to provide objective opinions, they tend to be negative and report that all roads are bad. We envision here an alternative crowd-sourcing solution where instead of sending opinions, road users send images from their mobile phones or dashcams. These images will then be analyzed and road conditions evaluated in an automated process. Figure 1 provides the overview of our proposed work toward this direction.

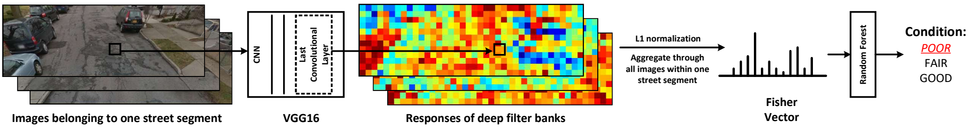

Images belonging to one street segment
CNN
Last Convolutional Layer
VGG16
Responses of deep filter banks
L1 normalization
Aggregate through all images within one street segment
Fisher Vector
Random Forest
Condition:
POOR
FAIR
GOOD

Figure 1: **Large-scale automatic visual road inspection.** The proposed method for automatic pavement rating is based on convolutional neural networks, Fisher Vector encoding, and random forest classifiers.

At a first glance, image-based pavement inspection seems to be a straightforward application of computer vision, where existing image and texture classification methods (e.g., [5, 19, 24, 41, 43, 44, 45, 47]) can be used. Unfortunately, this problem is challenging due to several reasons: 1) There is no sufficiently large and diverse dataset to train a deep-learning pavement classifier. 2) We need a method for fine-grained visual classification. Conventional texture classification or material recognition methods are designed to distinguish between categories of different materials. The appearance difference between two materials is much more pronounced than the difference between a material and its degraded version. What distinguishes between a fair and a good road might be a few cracks that do not occur uniformly. 3) Some levels of road degradation are rare, posing challenges to the creation of balanced training sets. 4) Labeling the level of degradation is more subjective than labeling distinct material categories. 5) We need to classify images taken in the wild, under diverse environmental conditions. Unlike many texture classification studies, we need to analyze images that do not just contain pavement. Crowd sourced images in the wild will contain structures unrelated to pavement conditions, such as cars, trees, and manholes.

To solve the first problem, we leverage maintenance records of transportation infrastructure publicly available online. Meanwhile, infrastructure images are easily accessible via map services such as Google Street View. So we can exploit these two data troves to create a dataset for material weathering classification. This is our **first contribution** in this paper. In order to localize infrastructure, records always contain coordinates from Global Positioning System (GPS), which we can utilize to fetch the corresponding images from Google Street View. Unlike other datasets where images are collected first and workers label the images later, we match a pre-existing set of labels to a set of images based on GPS information.

We demonstrate the effectiveness of this dataset creation method by building a pavement condition rating dataset for New York City. The records come from open data at the Department of Transportation (DoT). Our **second contribution** is this dataset. This is the first public large-scale dataset designed for studies in classifying material degradation levels, rather than classifying different material categories. This is also the first large-scale dataset for pavement condition rating studies.

The third contribution of this paper is a deep learning method that handles this challenging dataset. We utilize Fisher Vectors with Convolutional Neural Networks (FV-CNN), a superior method for texture classification to extract descriptors. Since street segments of different lengths have different numbers of images, we use Fisher Vectors to encode these street segments into fixed length descriptors. This orderless pooling also helps to solve the problem of degradation levels not being uniformly distributed along the street. We demonstrate that: 1) limiting the receptive field of CNN by processing image patches, and 2) $L1$ normalization of the features of the last convolutional layer safeguard that the network is not attracted by the irrelevant objects like pedestrians and vehicles on the road. Finally, we address the imbalance of data categories by using *UnderBagging* [15], also known as *EasyEnsemble* [28], to train a random forest, where each tree is trained using a balanced subset.

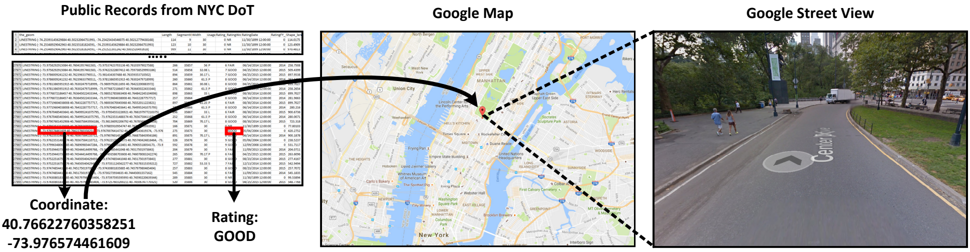


Figure 2: **Dataset creation pipeline.** We start from publicly available pavement condition rating records from New York City's Department of Transportation. The street view corresponding to the longitude and latitude coordinates is fetched automatically via the Google Street View API and added to the dataset.

2 Previous Work

Material Datasets. Existing material datasets can be roughly divided into two categories: i) the images contain only the labeled material, or ii) the images contain unrelated "material in the wild" content. For example, the CUReT dataset [12] has over 60 different materials, under 200 different viewing and illumination conditions, supporting the development of 3D textons [24]. A more diverse dataset is KTH-TIPS [6], which consists of material samples at multiple scales and multiple instances per category. The STAF dataset [18] records time-varying appearance of 26 material instances in 30 instances in time. More recently, the dataset for fine-grained material recognition in [21] focuses on fabric subtypes such as cotton, denim, and silk. The images in these datasets are taken under controlled environments. Compared to them, the FMD dataset [36] is a great step towards cataloging materials in the wild. It contains 10 material categories with 100 examples each. The images were drawn from Flickr and captured under unknown real-world conditions. However, FMD is too small to cover the diversity of the real-world materials and the hand-picked images are subjective. Similar to FMD, 4DLF [46] is a dataset of 12 categories, each with 100 images captured with a light field camera. The OpenSurfaces (OS) dataset [3] was the first large-scale dataset for material recognition and segmentation from natural images. It contained 105,000 images and was later curated by Cimpoi et al. [9] and augmented with some of the 47 attributes from DTD dataset [8] to produce the OSA dataset [9], a new dataset of 53,915 images. The MINC dataset [4] further increases the size of the materials dataset by an order of magnitude. It contains 3 million images. Rare categories are well sampled and diversity is also maintained.

Texture Classification. Texture reflects material properties of objects thus material recognition is often addressed as texture classification. Most studies focus on improving texture descriptors and orderless pooling. Early work used low-level image cues to describe texture: 3D textons [24], MR8 filter banks [41], and Local Binary Patterns [33]. Later SIFT [29] and dense SIFT features were widely used as texture descriptors. Recently, two texture descriptors were developed based on Convolutional Neural Networks (CNNs): FC-CNN [16] and FV-CNN [9]. FC-CNN extracts the output of the penultimate Fully-Connected layer while FV-CNN only uses the outputs of the last convolutional layer. To aggregate these features throughout an image and encode them into a fixed length feature vector, different pooling methods have been proposed. The three most popular orderless pooling methods are Bag of Words (BoW) [11], Vectors of Linearly Aggregated Descriptors (VLAD) [20] and Fisher

Vectors (FV) [34]. BoW is solely based on counting. VLAD uses first order statistics of the descriptors whereas FV also uses second order statistics. In many applications, FV achieves the best results [21]. More recently, bilinear orderless pooling [26] unifies all three orderless pooling methods mentioned above. The bilinear CNN model is shown to be superior to FV-CNN in fine-grained objects recognition but achieves similar result to FV-CNN in texture recognition [25]. A recent study [27] shows that FV-CNN is still the state of the art method for texture classification, especially for materials with large appearance variation.

Pavement Condition Rating. Attempts to introduce computer vision into pavement condition rating go back to 1990 [35]. Recent work includes pavement crack assessment [32, 40] and pothole localization via stereo vision [22]. However, these studies still rely on low level image cues such as LBP with hard thresholds on the response map [17, 31, 38] and are evaluated on small private datasets, which discourages comparison and generalization.

3 Dataset

We have collected an image dataset of road surfaces that contains more than 700K images from about 70K street segments. The dataset was created by downloading Google Street View images that are referenced in road condition reports. This is the first large-scale dataset for road surface inspection. Although we are not the first to use Google Maps to create a dataset, we follow a novel workflow for using the Google Street View API. Compared to some datasets (e.g., datasets for 3D city modeling [30, 39], image localization [48], and privacy protection [14]) where the data is first collected and subsequently labeled by human annotators, we start from the available annotation and retrieve images that correspond to the annotation, as shown in Fig. 2. Recently, similar methods were also applied by Lee et al. [23] and Arietta et al. [1] to create datasets. They focus more on the visual attributes of urban architecture.

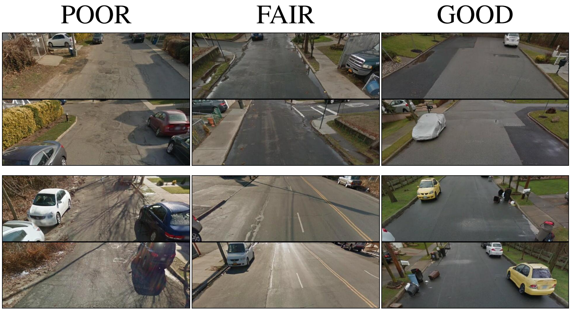


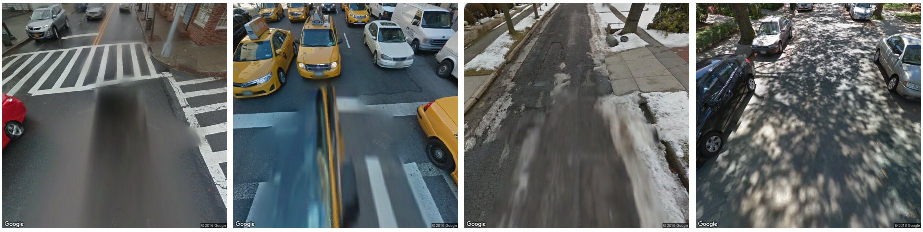
Figure 3: **Examples of pavement images**. For each condition, two street segments are shown, each with two images. Images were cropped from the original 640×640 images.

	Images	Streets
POOR	4650 (0.6%)	518 (0.7%)
FAIR	200553 (28.2%)	20612 (28.8%)
GOOD	506317 (71.2%)	50471 (70.5%)
Total	711520 (100%)	71601 (100%)

Table 1: **Data statistics** – the numbers and percentages of images and street segments in each pavement condition.

3.1 Data Sources

Roads are generally inspected by local and state governments' maintenance departments, and inspection reports are usually available to the public. For our dataset, we use the public records from New York City Department of Transportation. There are two benefits for using NYC data. First, the records of NYC contain GPS data for each street inspected. GPS data



(a) Rainy day (b) Too many cars (c) Snow cover (d) Tree shadow

Figure 4: **Image clutter and nuisance conditions.**

can be used in Google Street View to localize images. Second, as one of the biggest cities in the world, Google Street View images of New York are updated more frequently than other areas. We can easily find many images in Manhattan that were captured in 2016. This reduces the time gap between image acquisition and the corresponding pavement rating for more accurate labels.

The inspection records of New York City have many entries, but the most relevant pieces information for our purpose are: longitude, latitude, rating score, rating category, and rating date. In particular, longitude and latitude coordinates are used in Google Street View API to fetch images and the rating date is used to estimate the time gap between the rating scores and the images. Along with the records, the meta-data provides a conversion table for converting from the rating score to the rating category. The rating score ranges from 1 to 10, corresponding to three rating categories: "good" (8 to 10), "fair" (4 to 7), and "poor" (1 to 3). We choose rating categories over rating scores, because rating categories are less sensitive to time gaps. Therefore, this is a three-class classification problem.

The Google Street View API allows automatic image retrieval by specifying desired GPS coordinates and several other image and camera parameters. We used the following parameter settings: the desired output image size is 640×640, the field of view is 90°, the heading angle is the direction of the street, and the pitch angle is $-50°$.

3.2 Dataset properties

The pavement report of New York City contains 81,209 street segments. There are 71,601 rated street segments available on Google Street View. We collected a dataset of 711,520 images for those segments, summarized in Tab. 1:

Image resolution and region of interest. Each image has a resolution of 640×640 pixels. The image was retrieved at a $-50°$ pitch angle, and the bottom of the original image might have contained the front of the vehicle on which the camera is mounted. Post-processing often leaves discernible artifacts, as shown in Fig. 4 (a) and (b), hence we crop the image and only retain the top 640×224 area.

Image clutter. The images do not only contain pavement. They often contain other structures such as cars, pedestrians, trees, and manholes. This makes our dataset challenging and different from other texture datasets like OS [3] and KHT-TIPS [6], where the entire images show the texture patterns or the segmentation mask for the target texture area is available. This is one reason why the direct application of the state-of-the-art texture classification method [9] yields poor performance. One can consider a smaller image region, but it is also not guaranteed to solely contain pavement. More importantly, this will limit our consideration to a narrow part of a street that might not reflect the entire road condition. Pavement degradation can appear non-uniformly on a road.

Subtle inter-class differences. Images in our dataset are taken under diverse environmental conditions, unlike other texture datasets collected under controlled conditions [12, 21]. In our dataset, images from the same category can look drastically different, depending on the construction materials (e.g., concrete, asphalt, composite) and weather and illumination conditions (e.g., sunny, snow, shadow). Inter-class differences can be very subtle compared to intra-class differences. The difference between fair and good conditions might be a few cracks that do not occur uniformly in an image, which are more subtle than shadows and line markings (Fig. 4). As shown in Sec. 5, these distractions severely affect the performance of texture classification methods.

Label noise. The estimated time gap between when an image was taken and when it was rated is 1.2 year (estimated on a small subset of the data). This time gap can lead to a discrepancy between some images and their rating labels.

Class imbalance. Only 0.7% of the pavement data is rated poor. The other two categories, fair and good, correspond to 28.8% and 70.5% of the data respectively. Given this imbalanced data distribution, the classifier might be biased against returning the POOR label. Because one purpose of our work is to monitor road conditions and identify roads in poor condition that need to be repaired, class imbalance must be carefully taken into account.

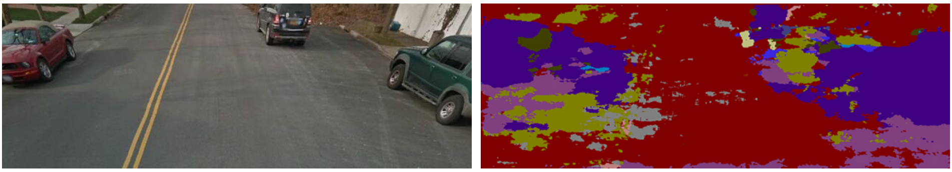
4 Pavement classification method

We develop a pavement classification method based on FV-CNN [9], a method that combines CNN features and Fisher Vector encoding [34]. FV-CNN is the state-of-the-art method for texture classification, especially for coarse-grain texture classification [25, 27]. It inputs an image (at any size) and extracts the features after the last convolutional layer. The output is a feature block of the size $m \times n \times p$ where m and n are determined by the size of the input image, and p is the number of output channels. The output can be considered as multiple p-dimensional feature vectors. The feature vectors for multiple images of a road segment are pooled and encoded using Fisher Vector, yielding a fixed length descriptor. The descriptor is L_2 normalized and fed into a linear SVM classifier.

Unfortunately a direct application of FV-CNN to pavement classification yields poor performance due to the challenges explained in the previous section. We propose to address these challenges with the following improvements:

1. We use image patches instead of the entire image to extract the descriptor. This is to prevent the network from focusing on non-pavement elements. The root of the problem is that the VGG-D [37] network used in our framework was trained to detect objects rather than subtle pavement cracks. If we were to pass the whole image to the network, feature responses corresponding to non-pavement elements such as vehicles and traffic signs would be much stronger than feature responses in the pavement regions. When the Fisher Vectors are normalized, the dynamic range of pavement responses would be severely squashed. The subtle signals that distinguish between different pavement conditions would therefore be dominated by the higher dynamic range of other distraction structures. Thus, we should use image patches instead of the entire image.

We also considered pavement segmentation as a pre-processing step, but found it ineffective. Pavement segmentation is a difficult problem on its own. On one hand, we do not have pixel level annotation of the road images to train a pavement segmentation network. On the other hand, off-the-shelf semantics segmentation methods such as SegNet [2] performs



(a) input image (b) semantic segmentation output

Figure 5: **Poor results of an off-the-shelf pavement segmentation method [2].** Most of the pavement is predicted as building (red pixels).

poorly on our dataset as shown in Fig. 5(b).

2. We normalize the CNN feature vector. This step is critical to reduce distraction impact, by converting feature vectors to the same scale. Since we extract the output after a ReLU operation, which guarantees non-negativity, either L_1 or L_2 normalization is possible. Both L_1 and L_2 normalization improve performance, but L_1 is better.

3. We use *UnderBagging* random forest as the classifier. UnderBagging enables us to train a random forest [10] where each tree is trained using a class-balanced subset: we use all the data points from the class with the smallest number of data points and randomly sample the same number of data points from the other classes. Though each tree is trained with a small number of samples and the data dimension is high, at each split point of the tree, only a small portion of the features is considered. Decisions from multiple trees are fused together, reducing the effect of overfitting. The ensemble of trees, trained with class-balanced datasets, can handle the imbalance of classes well.

For a classifier like SVM, we can also assign different class weights to the training samples. It is equivalent to duplicating the data points of the minority class several times. But this approach is sensitive to the choice of the regularization parameter C. For a large C, the SVM will overfit to the minority class. For a small C, the margin term in the SVM objective will dominate the total loss, and it weakens the effects of the class weights. As will be seen in Sec. 5.3, with FV-CNN patch feature, UnderBagging random forest handles the class imbalance better than SVM.

5 Experiments

5.1 Train/test/validation partition

We divide our dataset into disjoint train, validation, and test subsets. This is done by dividing the street segments instead of individual images; this is to ensure that images of the same street segment are placed in the same subset, maximizing the independence between training and test data. From the total of 71,601 street segments, the first half is used as the test set. For the second half, three quarters are assigned to the train set and the remaining is used for validation. The partition is random with one constraint that the proportion of images in each pavement category is preserved. The results reported in Sec. 5.3 are obtained on the test set.

5.2 Descriptors and classifiers

We experiment with three different descriptor types: SIFT, FC-CNN, and FV-CNN, and two types of classifiers: SVMs and random forests.

SIFT. As a baseline, we implement a classifier using dense SIFT followed by pooling by

Model	FV-SIFT SVM	FC-CNN SVM	FV-CNN Image SVM	FV-CNN Patch SVM	FV-CNN Patch L1 SVM	FV-CNN Patch L1 RF
POOR	78.0	68.3	1.2	18.5	33.6	72.2
FAIR	35.8	35.2	41.8	36.1	30.6	50.7
GOOD	46.6	42.6	84.4	86.7	85.9	51.7
AVG	53.5	48.7	42.5	47.1	50.0	**58.2**

Table 2: **Experiment Setup and Results.** The table shows the percentage accuracy for each of the three classes individually, as well as the average accuracy for theb three classes in the last row. "Image" means the input of network is the whole image (640 × 224), while "Patch" means we use 64 × 64 image patches as the network input. "RF" is short for random forest. Models with "L1" are using normalized features.

Fisher Vector at street segment level using VLFeat library [42]. The window size used to extract dense SIFT is 64×64 with a step size of 32×32. For Fisher Vector encoding, we use a Gaussian Mixture Model (GMM) with 256 Gaussians.

FC-CNN. We implement a classifier that is based on the FC-CNN descriptor. It uses the outputs of the penultimate fully-connected layer as features. The VGG-D model used in all the experiments and the CNN implementation are provided by Keras [7] with a Tensorflow [1] backend. All the images are resized to 224 × 224. The outputs is 4096-dimensional. We first project this high dimension feature vector to 512 dimensions using PCA, preserving 86% of the data variance of the data. We then use Fisher Vector to pool at street segment level. The number of GMM centers is set to 64.

FV-CNN. We consider several variant methods for computing FV-CNN features. To demonstrate the effects of the size of receptive fields, we compare two input options, using 1) the original size 640 × 224 as in [9]; 2) 64 × 64 image patches. To study the effects of normalization, we consider L_1 normalization.

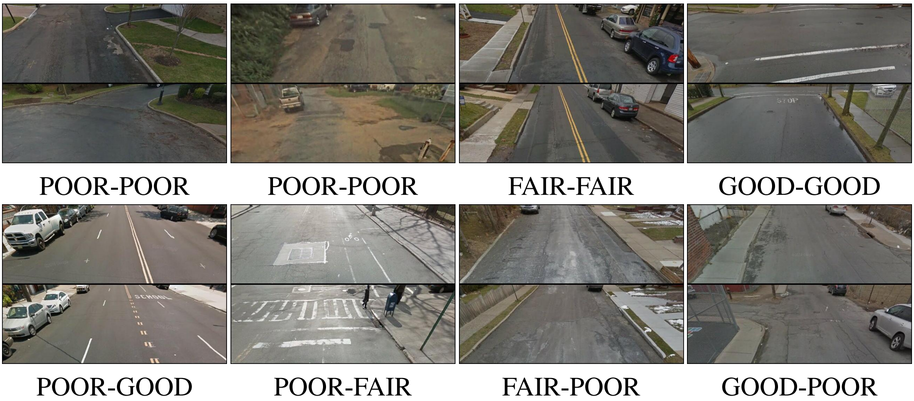
SVM. We experiment with linear SVMs [13]. The best C is found by grid search on the validation set. Only the best results of each method are shown in Section 5.3. The weight for each class used in training is: 97.4 for "poor", 2.4 for "fair" and 1 for "good", which is reciprocal to the number of samples of each class.

Random forests. We propose to use a random forest classifier. We set the number of trees to be 300 and the number of features that each split considers is 256 (the dimension of the feature vector is 65536).

5.3 Results

The results of our experiments are summarized in Tab. 5.2. The average accuracy at chance is 33.3%. The best recognition result is achieved using a random forest classifier, with FV-CNN features, patch input, and L_1 normalization. The classification accuracy for the three categories poor, fair, and good are 72.2%, 50.7% and 51.7%, respectively, and the average accuracy is 58.2%. Switching from random forest to SVM lowers the average accuracy to 50.0%. If L_1 normalization is also removed, the average accuracy drops to 47.1%. Replacing

[1] http://tensorflow.org



POOR-POOR POOR-POOR FAIR-FAIR GOOD-GOOD

POOR-GOOD POOR-FAIR FAIR-POOR GOOD-POOR

Figure 6: **Success and failure cases.** The tag-pair under each image indicates the ground truth and our prediction. For example, POOR-FAIR means the ground truth label is POOR while our prediction is FAIR. So the first row shows the success cases. The second row shows the failure cases.

patches with the whole image at its original size (640 × 224) has an average accuracy of 42.5%, 4.6% lower than when using patches. If we switch features from FV-CNN to FC-CNN, we achieve an average accuracy of 48.7%, 9.5% lower than our best result. SIFT performs better compared to FC-CNN. It achieves 53.3% average accuracy, but still 4.9% lower than the best.

There is still ample space for improvement. Besides the comparison with the expert annotation, we also conducted a user study evaluating a non-expert annotation of this dataset. We randomly selected 300 street segments from the test set, 100 segments per class and ask 7 non experts to label these images. This resulted in an average accuracy if 51.4% with a standard deviation of 3.5%. The best non-expert rater achieved 55.0% accuracy compared to the expert rated ground truth. Subjective rating results diverge drastically among the non-expert raters. Their ratings agree at only 15.3% of all 300 data points. For these 46 data points, their accuracy is 74.1%. The low performance of non-expert raters might be due to the label noise caused by time gap mentioned in Section 3.1, and from the fact that non-experts might not be aware of features that affect pavement quality. The above statistics demonstrate the method we present can surpass the performance of non-expert raters. The diverse and subjective rating results among non-experts also prove that this is a challenging classification problem.

Several success and failure cases are shown in Figure 6. The first label of the label pair below each image is the ground truth rating and the other one is our prediction. The first row shows success cases, and the second row show failure cases. Some of them are likely caused by label noise like the first segment in this row. Some are affected by distractors on the road like in the third segment, where the salt brine that was sprayed on the road for anti-icing appears like a defect.

6 Conclusions and Future Work

We have presented a new approach to create datasets for computer vision tasks. We used public records with GPS data and images from Google Street View to create a large-scale pavement condition rating dataset. This is the first public large-scale dataset for classifying material in different degradation levels. Crucial modifications of FV-CNN, led to an average accuracy of 58.2% on the proposed dataset.

The "label-to-image" approach can also be used to create other datasets. Similar to pavements, condition reports of bridges and tunnels are also publicly available[2]. Images of bridges and tunnels can also be retrieved in a similar way.

In the future, if the time stamp of the StreetView images becomes automatically available, we can also study how the texture changes with time. Time varying texture synthesis can also benefit from this work.

Acknowledgements. This work was supported by the Stony Brook University SensorCAT, FRA DTFR5315C00011, a gift from Adobe, the Partner University Fund 4DVision project, and New York State ITSC. The authors would like to thank Professor Anil Yazici from the Stony Brook University Department of Civil Engineering for fruitful discussion and feedback.

References

[1] Sean M Arietta, Alexei A Efros, Ravi Ramamoorthi, and Maneesh Agrawala. City forensics: Using visual elements to predict non-visual city attributes. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2624–2633, 2014.

[2] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*, 2015.

[3] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. OpenSurfaces: A richly annotated catalog of surface appearance. *ACM Transactions on Graphics (TOG)*, 32 (4):111, 2013.

[4] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. Material recognition in the wild with the materials in context database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[5] Anna Bosch, Andrew Zisserman, and Xavier Munoz. Image classification using random forests and ferns. In *Proceedings of the International Conference on Computer Vision*, 2007.

[6] Barbara Caputo, Eric Hayman, and P Mallikarjuna. Class-specific material categorisation. In *Proceedings of the International Conference on Computer Vision*, 2005.

[7] Franccois Chollet. *Keras*. 2015. `https://github.com/fchollet/keras`.

[8] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[2] http://www.nyc.gov/html/dot/html/infrastructure/annualbridgereport.shtml

[9] Mircea Cimpoi, Subhransu Maji, and Andrea Vedaldi. Deep filter banks for texture recognition and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[10] Antonio Criminisi, Jamie Shotton, and Ender Konukoglu. Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Foundations and Trends® in Computer Graphics and Vision*, 7(2–3):81–227, 2012.

[11] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Proceedings of the European Conference on Computer Vision Workshops*, 2004.

[12] Kristin J Dana, Bram Van Ginneken, Shree K Nayar, and Jan J Koenderink. Reflectance and texture of real-world surfaces. *ACM Transactions on Graphics (TOG)*, 18(1):1–34, 1999.

[13] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. LIBLINEAR: A library for large linear classification. *Journal of machine learning research*, 9:1871–1874, 2008.

[14] Andrea Frome, German Cheung, Ahmad Abdulkader, Marco Zennaro, Bo Wu, Alessandro Bissacco, Hartwig Adam, Hartmut Neven, and Luc Vincent. Large-scale privacy protection in google street view. In *Proceedings of the International Conference on Computer Vision*, 2009.

[15] Mikel Galar, Alberto Fernandez, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera. A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(4):463–484, 2012.

[16] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.

[17] Kasthurirangan Gopalakrishnan, Omar G Smadi, Halil Ceylan, Koray Celik, and Arun K Somani. Machine-Vision-Based Roadway Health Monitoring and Assessment: Development of a Shape-Based Pavement-Crack-Detection Approach. Technical Report DOT F 1700.7 (8-72), U.S. Department of Transportation, 2016.

[18] Jinwei Gu, Chien-I Tu, Ravi Ramamoorthi, Peter Belhumeur, Wojciech Matusik, and Shree Nayar. Time-varying surface appearance: Acquisition, modeling and rendering. *ACM Transactions on Graphics (TOG)*, 25(3):762–771, 2006.

[19] Minh Hoai. Regularized max pooling for image categorization. In *Proceedings of the British Machine Vision Conference*, 2014.

[20] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[21] Christos Kampouris, Stefanos Zafeiriou, Abhijeet Ghosh, and Sotiris Malassiotis. Fine-grained material classification using micro-geometry and reflectance. In *Proceedings of the European Conference on Computer Vision*, 2016.

[22] Christian Koch, Zhenhua Zhu, Stephanie German Paal, and Ioannis Brilakis. Machine vision techniques for condition assessment of civil infrastructure. In *Integrated Imaging and Vision Techniques for Industrial Inspection*, pages 351–375. 2015.

[23] Stefan Lee, Nicolas Maisonneuve, David Crandall, Alexei Efros, and Josef Sivic. Linking past to present: Discovering style in two centuries of architecture. In *Proceedings of the IEEE International Conference on Computational Photography*, 2015.

[24] Thomas Leung and Jitendra Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International journal of computer vision*, 43(1):29–44, 2001.

[25] Tsung-Yu Lin and Subhransu Maji. Visualizing and Understanding Deep Texture Representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[26] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear CNN models for fine-grained visual recognition. In *Proceedings of the International Conference on Computer Vision*, 2015.

[27] Li Liu, Paul Fieguth, Xiaogang Wang, Matti Pietikäinen, and Dewen Hu. Evaluation of LBP and Deep Texture Descriptors with a New Robustness Benchmark. In *Proceedings of the European Conference on Computer Vision*, 2016.

[28] Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550, 2009.

[29] David G Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision*, 1999.

[30] Branislav Micusik and Jana Kosecka. Piecewise planar city 3D modeling from street view panoramic sequences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[31] A Miraliakbari, S Sok, YO Ouma, and M Hahn. Comparative Evaluation of Pavement Crack Detection Using Kernel-Based Techniques in Asphalt Road Surfaces. *ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pages 689–694, 2016.

[32] Soroush Mokhtari. *Analytical Study of Computer Vision-Based Pavement Crack Quantification Using Machine Learning Techniques.* PhD thesis, University of Central Florida Orlando, Florida, 2015.

[33] Timo Ojala, Matti Pietikäinen, and David Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern recognition*, 29 (1):51–59, 1996.

[34] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *Proceedings of the European Conference on Computer Vision*, 2010.

[35] Stephen G Ritchie. Digital imaging concepts and applications in pavement management. *Journal of transportation engineering*, 116(3):287–298, 1990.

[36] Lavanya Sharan, Ruth Rosenholtz, and Edward Adelson. Material perception: What can you see in a brief glance? *Journal of Vision*, 9(8):784–784, 2009.

[37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[38] Yao Sun, Ezzatollah Salari, and Eb Chou. Automated pavement distress detection using advanced image processing techniques. In *2009 IEEE International Conference on Electro/Information Technology*, pages 373–377, 2009.

[39] Akihiko Torii, Michal Havlena, and Tomas Pajdla. From google street view to 3d city models. In *Proceedings of the International Conference on Computer Vision Workshops*, 2009.

[40] Srivatsan Varadharajan, Sobhagya Jose, Karan Sharma, Lars Wander, and Christoph Mertz. Vision for road inspection. In *Proceedings of the IEEE Workshop on Applications of Computer Vision*, 2014.

[41] Manik Varma and Andrew Zisserman. Texture classification: Are filter banks necessary? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[42] Andrea Vedaldi and Brian Fulkerson. VLFeat: An open and portable library of computer vision algorithms. In *International Conference on Multimedia*, 2010.

[43] Tomas F. Yago Vicente, Minh Hoai, and Dimitris Samaras. Leave-one-out kernel optimization for shadow detection. In *Proceedings of the International Conference on Computer Vision*, 2015.

[44] Tomas F. Yago Vicente, Minh Hoai, and Dimitris Samaras. Noisy label recovery for shadow detection in unfamiliar domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[45] Boyu Wang, Kevin Yager, Dantong Yu, and Minh Hoai. X-ray scattering image classification using deep learning. In *Proceedings of Winter Conference on Applications of Computer Vision*, 2017.

[46] Ting-Chun Wang, Jun-Yan Zhu, Ebi Hiroaki, Manmohan Chandraker, Alexei A Efros, and Ravi Ramamoorthi. A 4D Light-Field Dataset and CNN Architectures for Material Recognition. In *Proceedings of the European Conference on Computer Vision*, 2016.

[47] Zijun Wei and Minh Hoai. Region ranking SVMs for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[48] Amir Roshan Zamir and Mubarak Shah. Accurate image localization based on google maps street view. In *Proceedings of the European Conference on Computer Vision*, 2010.