A Study for Universal Adversarial Attacks on Texture Recognition

Yingpeng Deng\(^1\) and Lina J. Karam\(^{1,2}\)

\(^1\)Image, Video and Usability Lab, School of ECEE, Arizona State University, Tempe, AZ, USA
\(^2\)Dept. of ECE, School of Engineering, Lebanese American University, Lebanon
{ypdeng, karam}@asu.edu

Abstract

Given the outstanding progress that convolutional neural networks (CNNs) have made on natural image classification and object recognition problems, it is shown that deep learning methods can achieve very good recognition performance on many texture datasets. However, while CNNs for natural image classification/object recognition tasks have been revealed to be highly vulnerable to various types of adversarial attack methods, the robustness of deep learning methods for texture recognition is yet to be examined. In our paper, we show that there exist small image-agnostic/universal perturbations that can fool the deep learning models with more than 80% of testing fooling rates on all tested texture datasets. The computed perturbations using various attack methods on the tested datasets are generally quasi-imperceptible, containing structured patterns with low, middle and high frequency components.

1 Introduction

Texture is one of the essential features for objects or identities in the physical world. It usually yields visually distinguishable properties to provide mid-level hints for human/machine vision. For example, Convolutional neural networks (CNNs) \([1, 2, 3, 4]\) are regarded as highly effective solutions to ImageNet ILSVRC dataset \([5]\), a challenging large-scale image classification dataset containing more than one million of real-world images categorized into one thousand classes. In the inference process of CNNs, local features are collected and combined by convolutional filter banks from previous layers, where textures in the input image are the valuable local information to collect for the convolutional filters in the first few layers. Moreover, a recent study has shown that
ImageNet-trained CNNs rely on textures rather than shapes to recognize images/objects [6], which can indicate that textures play a pivotal role in the success of the CNN models for object recognition.

In fact, texture recognition is already a widely explored topic. Before the prevalence of CNNs, researchers try to extract or design discriminative features from texture patterns via various kinds of methods, such as Scale-Invariant Feature Transform (SIFT) [7], Bag-of-Words (BoWs) [8], Vector of Locally Aggregated Descriptors (VLAD) [9], Fisher Vector (FV) [10] and so forth. Given the superior performance of CNNs on object recognition problems since the early 2010s [11], Cimpoi et al. [12] took advantage of pretrained CNN features and processed them with different encoders to achieve cutting-edge performance on texture recognition problems. Furthermore, inspired by dictionary learning, Zhang et al. [13] proposed a Deep Texture Encoding Network (DeepTEN), making end-to-end learning possible, followed by some variants to improve DeepTEN [14, 15].

Despite the outstanding performances of CNNs, some researchers have casted doubt on their robustness against small perturbations [16]. Specifically, it has been shown that there exist small and image-agnostic perturbations, called universal perturbations, that can fool CNN classifiers, resulting in a significant decrease in classification accuracy over a natural image set [17]. However, to the best of our knowledge, the performance of deep neural networks (DNNs) under universal adversarial attacks has not been studied for texture recognition. Given the properties of repetitive/redundant information and spatially invariant representation in texture images, what is the effect of universal perturbations on the performance of DNNs for texture image recognition? Can the performance be severely degraded by universal perturbations that are computed with much less training images and classes than those computed on ImageNet? Do the adopted modules and architectures in the deep learning models for texture recognition really improve the ability and robustness of the baseline DNN models? In order to answer these questions, we train four available deep learning models on six texture datasets and conduct adversarial attack experiments with existing universal attack algorithms. We perform the universal attacks with the assumption that we have full knowledge about the targeted model (white-box attack), such as its architecture, weight and gradient information, and that relevant training data is available (data-dependent). After the computation of the perturbation, we use testing data to evaluate the attack performance. The rest of our paper is organized as follows. Section 2 presents an overview of related work on texture recognition and adversarial attacks. Our experimental setup and obtained performance results are described in Section 3. Finally a conclusion is given in Section 4.
2 Related Work

2.1 Texture Recognition

Here we only consider the end-to-end deep-learning models for texture recognition, which most of the existing universal attack methods were proposed to fool. Inspired by previous dictionary learning and residual encoding techniques such as VLAD and Fisher Vector, Zhang et al. [13] proposed a learnable residual encoding layer, where dictionary codewords and assigned weights can both be updated via backpropagation, and plugged it before the fully connected layer of the ResNet structure. Further, to take the local spatial information into consideration, Xue et al. [14] combined the features after the global average pooling layer with the outputs of the encoding layer proposed in [13] through a bilinear model [15] in their deep encoding pooling (DEP) network. Recently, Hu et al. [15] presented the idea of multi-level texture encoding and representation (MuLTER) by assembling multi-stage features extracted using the DEP module after each residual block before the final fully connected layer. Intuitively, all these architectures are derived from the vanilla ResNet architecture.

2.2 Universal Attacks

White-box universal perturbations that are computed using relevant training data can produce the stronger attack ability than other kinds of universal perturbations such as data-independent [19] and black-box ones [20]. Based on the image-dependent attack algorithm DeepFool [21], Moosavi-Dezfooli et al. [17] produced universal adversarial perturbations (UAP) by accumulating the image-dependent updates iteratively over the training images to fool the targeted CNN model. Generative adversarial networks were adopted to generate the universal perturbation in [22, 23], where the adversarial networks were set as the targeted models. Poursaeed et al. [22] trained the generative network using a fixed random pattern as input to produce the generative adversarial perturbation (GAP), while Mopuri et al. [23] introduced both fooling and diversity objectives as the loss function of their network for adversary generation (NAG) to learn the perturbation distribution in the latent space. Recently, the authors of [24] and [25] used mini-batch based stochastic projected gradient descent (sPGD) during training by maximizing the average loss over each mini-batch, enlightened by another image-dependent attack algorithm called iterative gradient sign method (IGSM) or projected gradient descent (PGD) [26]. Most recently, Deng and Karam [27] proposed an enhanced universal projected gradient descent (UPGD) based on the UAP framework by replacing DeepFool with the stronger PGD attack and boosting the computation process with momentum.

3 Experiments

We consider six texture datasets in order to examine the recognition performance of the deep learning models under various universal attacks. The Ma-
Table 1: The numbers of data used for training the networks (training set), computing the perturbation (attacking set) and testing (testing set).

| Dataset   | training set | attacking set | testing set |
|-----------|--------------|---------------|-------------|
| MINC [28] | 48,875       | 5,750         | 5,750       |
| GTOS [29, 14] | 93,945       | 9,381         | 6,066       |
| DTD [30]  | 3,760        | 1,880         | 1,880       |
| 4DLF [31] | 840          | 360           | 360         |
| FMD [32]  | 900          | 300           | 100         |
| KTH [33]  | 2,376        | 231           | 2,376       |

The Materials in Context (MINC) Database [28] is a large real-world material dataset. In our work, we adopt its publicly available subset MINC-2500 with provided train-test split. There are 23 classes with 2500 images for each class. Xue et al. created the Ground Terrain in Outdoor Scenes (GTOS) [29] dataset with 31 classes of over 90,000 ground terrain images and the GTOS-mobile [14] with the same classes but with a much smaller number of images (around 6,000 images). As in [14], we adopt the GTOS dataset as the training set and test the trained models on the GTOS-mobile dataset. In the rest of our paper, we use MINC and GTOS to refer to the MINC-2500 and the combined dataset of GTOS and GTOS-mobile, respectively.

We also adopt some smaller texture datasets. The Describable Textures Database (DTD) [30] includes 47 categories with 120 images per category. The 4D light-field (4DLF) material dataset [31] consists of 1200 images in total for 12 different categories. An angular resolution of $7 \times 7$ is used for each image in the dataset and we only use the one where $(u, v) = (-3, 3)$ in our experiments. The Flickr Material Dataset (FMD) [32] has 10 material classes and 100 images per class. The KTH-TIPS-2b (KTH) [33] comprises 11 texture classes, with four samples per class and 108 images per sample.

We evaluate the performance of ResNet [1], DeepTEN [13], DEP [14] and MuLTER [15] on the six texture datasets. For each dataset, the training set is used to train the models for texture recognition, the attacking set includes images that are randomly sampled from the training set for computing the perturbations, and the testing set is to evaluate the performance without and with universal attacks. The number of images for each purpose is listed in Table 1. We use the train-test split that is either provided in the dataset or suggested in [13]. To extract the attacking set, we randomly sample 10% of the images in the training image set for the GTOS/KTH dataset, one-third of the training images for the FMD dataset and a number of training images that is equal to the number of testing images for MINC, DTD and 4DLF separately.

3.1 Implementations

We train the four deep learning models on each of the six texture datasets and then conduct the mentioned universal attacks on 24 trained classifiers. Similar
to the training process of deep learning models, the perturbations are computed only on each of the attacking sets, whose images are randomly sampled from the corresponding training set, and then are evaluated on the testing data.

Training. The deep learning models are finetuned via transfer learning based on the ResNet backbones which are pretrained on ImageNet. We follow the suggestions of backbones on different datasets in [13, 14]. In regard to the network backbone, we use pretrained ResNet50 for MINC, DTD, 4DLF and FMD and pretrained ResNet18 for GTOS and KTH. Following the same training strategies in [14, 15], we train our models with only single-size images. Similar to the data augmentation strategies in [13], the input images are resized to 256 × 256 and then randomly cropped to 224 × 224, followed by a random horizontal flipping. Standard color augmentation and PCA-based noise are used as in [11]. For DeepTEN, DEP and MuLTER, the number of codewords is set to 8 for the ResNet18 backbone and to 32 for the ResNet50 backbone. The learning rate is initialized to 0.01 and decays every 10 epochs by a factor of 0.1, with a momentum coefficient of 0.9. The training process stops after 30 epochs.

Attacking. Generally, no data augmentation is used while computing the universal perturbations. But in our implementation, given the small number of data for 4DLF, FMD and KTH, we use random cropping and horizontal flipping to prevent overfitting problems for perturbations. We perform UAP [17], GAP [22], sPGD [25] and UPGD [27] on the trained models. In [22], the authors mention two ways to optimize the perturbation - (1) minimizing the least-likely class loss between the original prediction with the least probability and the perturbed prediction; (2) maximizing the original loss between the perturbation prediction and the true target. We refer to the former as GAP-llc and to the latter as GAP-tar. For GAP and sPGD, we use a mini-batch size of 32 when computing the perturbation. The defaulted parameters are used for all the considered attack methods, except for UPGD whose hyperparameters (i.e., initial learning rate and decay factor, momentum, etc.) were varied to maximize the performance for the computed perturbations in terms of fooling rate. Typically, we stop the training process of the perturbation and report the results when the fooling rates are unable to increase by more than 0.5% within 5 epochs for all the attack methods. For UAP, we choose its best results in terms of fooling rate within 20 epochs given that its learning curves were observed to fluctuate drastically. All perturbations are constrained within an $l_\infty$ norm of 10 on 8-bit images (about 0.04 for the normalized image scale [0, 1]).

3.2 Results
3.2.1 Performance Comparison Results Without Attacks

We show the classification accuracies of each model without attacks on the six texture datasets in Table 2. For MINC and GTOS datasets, it seems that the

---

1 According to [34], the adopted ResNet50 backbones are slightly different from [4], where the first convolutional layer with a kernel size of 7 is replaced by three cascaded 3 × 3 convolutional layers.
Table 2: Top-1 accuracies on the testing sets for texture recognition. Bold number in each column indicates best performance on the corresponding dataset.

|       | MINC   | GTOS   | DTD    | 4DLF   | FMD    | KTH    |
|-------|--------|--------|--------|--------|--------|--------|
| ResNet| 81.5   | 80.1   | 71.4   | 72.5   | 70.0   | 79.9   |
| DeepTEN| 81.7   | 77.8   | 61.2   | 79.2   | 82.0   | **83.5** |
| DEP   | 81.5   | 77.5   | **73.0** | 82.2   | 81.0   | 79.9   |
| MuLTER | 81.8   | 77.9   | 71.6   | **83.6** | **84.0** | 80.6   |

Table 3: Fooling rates by the universal attacks against different models. Bold number indicates best performance on the corresponding model in each column within the same dataset and italic number denotes best performance on the corresponding dataset over different models and attack methods.

|       | ResNet | DeepTEN | DEP | MuLTER | mean | ResNet | DeepTEN | DEP | MuLTER | mean |
|-------|--------|---------|-----|--------|------|--------|---------|-----|--------|------|
| UAP   | 86.6   | 73.1    | 65.8 | 66.4   | 71.5 | 51.0   | 27.7   | 31.6 | 39.9   | 57.3 |
| GAP-lc| 55.7   | 75.3    | 61.4 | 76.0   | 87.1 | 69.7   | 75.6   | 81.1 | 72.2   | 74.7 |
| GAP-tar| **94.1** | **94.5** | **94.2** | 94.4 | 69.5 | **81.5** | **72.5** | **79.0** | 75.6 |
| sPGD  | 93.8   | 94.3    | 93.1 | 93.4   | 93.7 | 51.8   | 74.9   | 70.0 | 76.3   | 68.3 |
| UPGD  | 93.4   | 93.7    | 93.1 | 93.7   | 93.5 | **78.0** | 77.5   | 74.7 | 77.8   | **76.4** |

3.2.2 Attacking Results

In Table 3, we list the fooling rate results on the considered six texture datasets. It can be observed that UAP exhibits a large performance gap in terms of fooling rate as compared to the other attack methods, with a significantly lower performance on the GTOS and DTD datasets. For the two GAP variants, GAP-lc results in significantly lower fooling rates as compared to GAP-tar,
while GAP-tar results in the best performances on most targeted models for the MINC, GTOS, DTD and FMD datasets. As for sPGD and UPGD, they produce competitive results that are close on average to the top performing GAP-tar attack method on the considered datasets.

In addition, GAP-tar, sPGD and UPGD can easily fool all the four models that are trained for the MINC dataset on over 93% of the testing images. Concerning the vulnerability of the targeted models, attacking DeepTEN yields the highest fooling rates on four (MINC, GTOS, DTD and FMD) out of the six datasets. Overall, the highest fooling rate on each dataset can exceed 80%, meaning that an effective universal perturbation generally exists for the evalu-
3.2.3 Visual Examples

Figure 1 and 2 show, respectively, some visual examples of images perturbed by GAP-tar and sPGD against four models for each dataset (Section 3.2.2). Typically, perturbations are quasi-imperceptible given the relatively small $l_{\infty}$ norm. The perturbations for the GTOS dataset consist of colored patterns and blobs, which are more perceivable than the other types of patterns. According to Figure 3, most of the perturbations show dense texture patterns, which
Figure 3: Visualization of perturbations computed using different universal attack methods (first row) against DEP models on the six datasets (first column). The perturbations are rescaled for visibility.

are considered as middle/high frequency components, while some perturbations especially for the GTOS dataset also include colored structured patterns (e.g., spots and strips).

4 Conclusion

To evaluate the robustness of deep learning models for texture recognition against universal attacks, we train four deep learning models (ResNet, DeepTEN, DEP and MuLTER) on six texture datasets (MINC, GTOS, DTD, 4DLF, FMD and KTH) separately and conduct five existing universal attack methods (UAP, GAP-lle, GAP-tar, sPGD and UPGD) by targeting all the models. The ob-
served results show that highly effective universal perturbations exist over all tested datasets regardless of training data size and class number, achieving fooling rates of more than 80%. We also observe that these perturbations contain various components like colored flat regions (low frequency), intense lines (middle frequency) and dense textures (high frequency).

References

[1] Karen Simonyan and Andrew Zisserman, “Very deep convolutional networks for large-scale image recognition,” in International Conference on Learning Representations, 2015.

[2] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich, “Going deeper with convolutions,” in IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.

[3] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna, “Rethinking the inception architecture for computer vision,” in IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2818–2826.

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, “Deep residual learning for image recognition,” in IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[5] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei, “ImageNet large scale visual recognition challenge,” International Journal of Computer Vision, vol. 115, no. 3, pp. 211–252, 2015.

[6] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel, “Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness,” in International Conference on Learning Representations, 2018.

[7] David G Lowe, “Distinctive image features from scale-invariant keypoints,” International Journal of Computer Vision, vol. 60, no. 2, pp. 91–110, 2004.

[8] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray, “Visual categorization with bags of keypoints,” in Workshop on statistical learning in computer vision, ECCV. Prague, 2004, vol. 1, pp. 1–2.

[9] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez, “Aggregating local descriptors into a compact image representation,” in IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2010, pp. 3304–3311.
[10] Florent Perronnin, Jorge Sánchez, and Thomas Mensink, “Improving the fisher kernel for large-scale image classification,” in *European Conference on Computer Vision*. Springer, 2010, pp. 143–156.

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[12] Mircea Cimpoi, Subhransu Maji, and Andrea Vedaldi, “Deep filter banks for texture recognition and segmentation,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3828–3836.

[13] Hang Zhang, Jia Xue, and Kristin Dana, “Deep TEN: Texture encoding network,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 708–717.

[14] Jia Xue, Hang Zhang, and Kristin Dana, “Deep texture manifold for ground terrain recognition,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 558–567.

[15] Yuting Hu, Zhiling Long, and Ghassan AlRegib, “Multi-level texture encoding and representation (MuLTER) based on deep neural networks,” in *IEEE International Conference on Image Processing*. IEEE, 2019, pp. 4410–4414.

[16] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus, “Intriguing properties of neural networks,” *International Conference on Learning Representations*, 2014.

[17] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard, “Universal adversarial perturbations,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1765–1773.

[18] William T Freeman and Joshua B Tenenbaum, “Learning bilinear models for two-factor problems in vision,” in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1997, pp. 554–560.

[19] Hong Liu, Rongrong Ji, Jie Li, Baochang Zhang, Yue Gao, Yongjian Wu, and Feiyue Huang, “Universal adversarial perturbation via prior driven uncertainty approximation,” in *IEEE International Conference on Computer Vision*, 2019, pp. 2941–2949.

[20] Yusuke Tsuzuku and Issei Sato, “On the structural sensitivity of deep convolutional networks to the directions of fourier basis functions,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 51–60.
[21] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard, “DeepFool: A simple and accurate method to fool deep neural networks,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2574–2582.

[22] Omid Poursaeed, Isay Katsman, Bicheng Gao, and Serge Belongie, “Generative adversarial perturbations,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4422–4431.

[23] Konda Reddy Mopuri, Utkarsh Ojha, Utsav Garg, and R Venkatesh Babu, “NAG: Network for adversary generation,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 742–751.

[24] Chaithanya Kumar Mumudii, Thomas Brox, and Jan Hendrik Metzen, “Defending against universal perturbations with shared adversarial training,” in *IEEE International Conference on Computer Vision*, 2019, pp. 4928–4937.

[25] Ali Shafahi, Mahyar Najibi, Zheng Xu, John P Dickerson, Larry S Davis, and Tom Goldstein, “Universal adversarial training,” in *AAAI*, 2020, pp. 5636–5643.

[26] Alexey Kurakin, Ian Goodfellow, and Samy Bengio, “Adversarial examples in the physical world,” *arXiv preprint arXiv:1607.02533*, 2016.

[27] Yingpeng Deng and Lina J. Karam, “Universal adversarial attack via enhanced projected gradient descent,” in *IEEE International Conference on Image Processing*, 2020, pp. 1241–1245.

[28] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala, “Material recognition in the wild with the materials in context database,” in *IEEE conference on Computer Vision and Pattern Recognition*, 2015, pp. 3479–3487.

[29] Jia Xue, Hang Zhang, Kristin Dana, and Ko Nishino, “Differential angular imaging for material recognition,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 764–773.

[30] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi, “Describing textures in the wild,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3606–3613.

[31] Ting-Chun Wang, Jun-Yan Zhu, Ebi Hiroaki, Mannmohan Chandraker, Alexei A Efros, and Ravi Ramamoorthi, “A 4d light-field dataset and cnn architectures for material recognition,” in *European Conference on Computer Vision*. Springer, 2016, pp. 121–138.

[32] Lavanya Sharan, Ce Liu, Ruth Rosenholtz, and Edward H Adelson, “Recognizing materials using perceptually inspired features,” *International Journal of Computer Vision*, vol. 103, no. 3, pp. 348–371, 2013.
[33] Barbara Caputo, Eric Hayman, and P Mallikarjuna, “Class-specific materia-
rial categorisation,” in *IEEE International Conference on Computer Vi-
sion*. IEEE, 2005, vol. 2, pp. 1597–1604.

[34] Hang Zhang, “Pytorch-encoding,” https://github.com/zhanghang1989/PyTorch-
Encoding/blob/master/encoding/models/backbone/resnet.py.