Photo-Sketching: Evaluation Project

Michael Tran, Victor Chen

Abstract—Edges, boundaries and contours are important subjects of study in both computer graphics and computer vision. On one hand, they are the 2D elements that convey 3D shapes, on the other hand, they are indicative of occlusion events and thus separation of objects or semantic concepts. In this paper, we aim to leverage an existing framework titled PhotoSketch which generates contour drawings (boundary-like drawings), that capture the outline of a visual scene. We use these contours to propose a solution to problems that plague neural networks, namely adversarial images that aim to trick a neural network into misclassification. We also extend our classification task to gender classification using image contours.

Index Terms—Computer Vision, PhotoSketch, Contours, Deep Learning, Deep Neural Networks, Convolutional Neural Networks, Adversarial Images, Gender Classification

1 INTRODUCTION

Edge-like visual representation appearing in form of image edges, object boundaries, line drawings and pictorial scripts, is of great research interest in both computer vision and computer graphics. Contour drawings of images contain object boundaries and salient inner edges such as occluding contours and salient background edges. These sets of visual cues convey 3D perspective, length, width, thickness, and depth. This paper explores, studies, and evaluates the possibilities of image contours.

Deep learning algorithms such as Convolutional Neural Networks (CNNs) are powerful models that are trained on terabytes of data to be able to classify an image. CNNs consider the entire image when classifying and can recognize complex shapes and patterns no matter where they appear in the image. In many image recognition tasks, they can equal or even beat human performance. However, it is very easy to fool a neural network by simply changing a few pixels in an image to be darker or lighter. An adversarial (or hacked) image can cause a neural network to misclassify an image that it would, in normal circumstances, classify correctly.

In this paper, we will evaluate the potentials of using image contours to prevent hacked images from tricking convolutional neural networks. This evaluation depends on the effectiveness of a neural network trained on contoured images, which we test using real images and their contour-generated counterparts. We will also extend our classification task to gender classification and evaluate the potential of simply using image contours to determine a person's gender.

2 PHOTOSKETCH

Mentian et Al. [1] proposes a learning-based method named PhotoSketch that resolves diversity in the annotation, and unlike boundary detectors, can work with imperfect alignment of the annotation and the actual ground truth.

Specifically, they propose a contour generation algorithm to output contour drawings given input images. The generation process involves identifiying salient boundaries and is connected with the salient boundary detection in computer vision.

2.1 Dataset

The dataset consists of 5000 high-quality drawings made by humans of 1000 outdoor images crawled from Adobe Stock. To collect this dataset, they used a popular crowdsourcing platform called Amazon Mechanical Turk. [2] Turkers were allowed to trace over a fainted background image to ensure that drawings are roughly boundary aligned.

2.2 Method

The paper leverages a conditional generative adversarial network (cGAN), and a novel MM-loss (Min-Mean-loss) to generate the contours.

For cGANs, the generator aims to generate "real" images conditioned on the input images. Adversarially, the discriminator network is trained to tell the generated images from the actual ground truth images. Typically, this method expects a 1-to-1 mapping between the two domains. However, to accommodate the extra images in each training example, they used a MM-loss to account for this.

In the novel MM-loss, two different aggregation functions are used for the generator and discriminator respectively. The "mean" aggregate function asks the discriminator to learn from all modalities in the target domain, and treat those modalities with equal importance. The "min" aggregate function allows the generator to adaptively pick the most suitable modality to generate on-the-fly.

2.3 Model

In this paper, we will be using the original authors pretrained model. This model is trained on the dataset mentioned in the previous section, which mainly contains pictures of humans and dogs. Thus, the contours generated will be most accurate when generating contours of humans and dogs, however, we have tested it on other types of images with success.

3 Hacked Images

Generating hacked images is essentially the same as "generating an adversarial example". It is intentionally crafting a piece of data such that a machine learning model will misclassify it as something completely different. This can be used for something harmless such as a prank, but it can also be used for something malicious such as uploading an image (e.g. a pornographic image) that violates a websites terms of services. A deep neural network would normally be the first line of defence and able to catch something like that; however, a hacked image is a way to bypass that defence. Hacked images can even fool neural networks even when they are printed out on a piece paper! [3] This means that hacked images can not only fool systems that upload an image file directly, but also fool physical cameras or scanners.

3.1 Hacked Image Generation

The generation of hacked images is suprisingly simple, we just need to change a few pixels in an image to be darker or lighter. One would expect that changing a couple of pixels on an image would not matter to a deep neural network. However, in a famous paper in 2014 by C. Szegedy et al. [4] discovered that it isn't always true. If one knew exactly which pixels to change and exactly how much to change them, you can intentionally force the neural network to predict the wrong output without making any obvious changes to the human eye.

3.2 Pipeline

For this task, we use and aim to trick the Inception v3 image recognition model. This model was created by Google and has been shown to attain greater than 78.1% accuracy on the ImageNet dataset. [5] The model itself is made up of symmetric and asymmetric building blocks, including convolutions, average pooling, max pooling, concats, dropouts, and fully connected layers.

The following is a pipeline of how we will generate the hacked images:

1. Feed in an image.
2. Check Inception v3's prediction and see how far off the image is from the fake prediction.
3. Tweak the image using back-propagation to make the final prediction slightly closer to the fake prediction.
4. Repeat steps 1-3 with the same image until the Inception v3 network gives us the fake prediction we want.

One thing to keep in mind is that we cannot allow any single pixel to be adjusted without any limitations, or else the changes to the image can be drastic enough that it can be seen by the human eye. These changes will show up as discolored spots or wavy areas. To prevent these distortions, we add a constraint to our algorithm: no one single pixel in the hacked image can ever be changed by more than a tiny amount from the original image (we use 0.01%). This forces the algorithm to tweak the image in a way that still fools the neural network without having any obvious changes from the original image.

3.3 Results

For this paper, we force every input image we want to hack to be classified as a toaster. Fig. 1 and Fig. 2 below shows two seemingly identical images that Inception v3 believes to be totally different. We note that Inception v3 does not classify humans, but for all intents and purposes, we show that we can still force the network into thinking the image of a human is a toaster. Fig. 3 shows that we can even improve the networks confidence of a toaster image being a toaster.

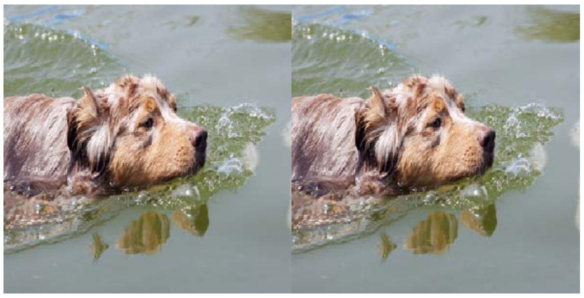

Fig. 1. – Left: Welsh_springer_spaniel with 26.75% confidence. Right: Toaster with 92.83% confidence.

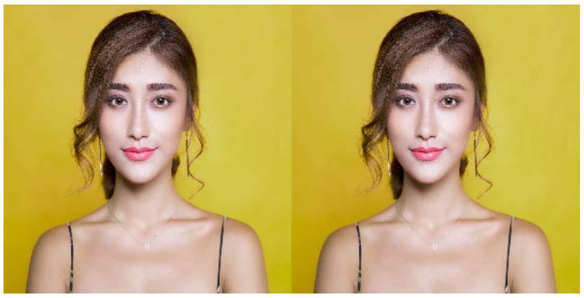

Fig. 2. – Left: Wig with 60.11% confidence. Right: Toaster with 99.56% confidence.

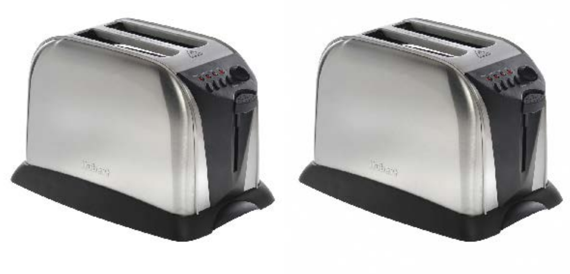

Fig. 3. – Left: Toaster with 98.26% confidence. Right: Toaster with 100% confidence.

3.4 Image Contours and Hacked Images

Given the original image and hacked image, we feed both images into the PhotoSketch generator. Fig. 4 and Fig. 5 shows the resultant image contours.

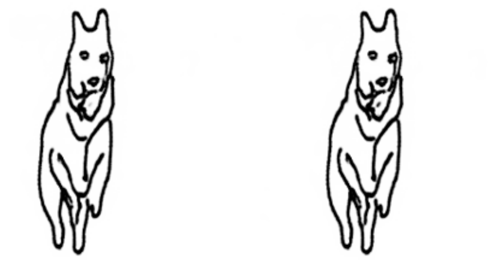

Fig. 4. – Left: Image contours of original image. Right: Image contours of hacked image.

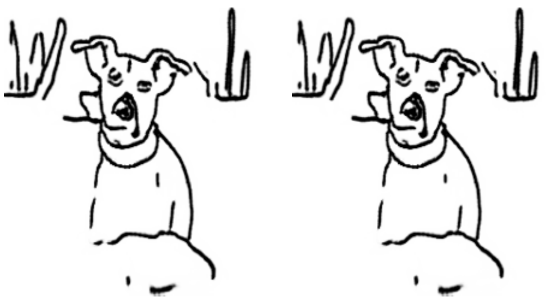

Fig. 5. – Left: Image contours of original image. Right: Image contours of hacked image.

At first glance, the image contours of Fig. 5 may look identical, but there are slight differences in the contours (some contour lines are longer in one image than the other). Overall, we can say that (at least from the human eye's perspective) the contours for hacked and non-hacked images are almost identical.

4 Neural Networks and Contour Images For Hacked Image Validation

Hacked images may be able to fool a neural network into misclassification, but would their image contours be able to fool a neural network? We surmise that since image contours remained almost identical when passing original images and hacked images to PhotoSketch, the image contours could at the very least serve as a validation test or sanity check for neural networks.

4.1 Method

To test our hypothesis, we decided to create a Convolutional Neural Network (CNN) trained on image contours. We used Google's TensorFlow as our CNN framework. Our CNNs architechture was composed of 4 convoultional and max pooling layers followed by a fully connected layer. We train the CNN using the original images contours of three distinct classes: dogs, humans, and toasters.

4.2 Dataset

For our dataset, we took pictures of dogs and humans from the PhotoSketch validation dataset. To help balance the dataset, we added additional photos of humans from the stock photo website Pexels. [7] We also added images of toasters to our dataset found on Google Image Search. [8] In total, we had 123 images of dogs, 102 images of humans, and 21 images of toasters for a total of 246 images.

4.3 Pipeline

The following is the pipeline we follow to generate the results:

1. Hack all 246 images such that Inception v3 believes that they are all toasters.
2. Generate image contours for both hacked images and non-hacked images.
3. Split the non-hacked image contours into training/testing/validation sets (80/10/10).
4. Train the CNN using the data sets.
5. Get validation accuracy from the CNN.
6. Test our final trained CNN on all 246 hacked image contours and report accuracy.

4.4 Results

Table 1 shows the results of our CNN trained on image contours. We note that our accuracy is very dependent on the quality of contours that PhotoSketch generates. If PhotoSketch improves with more quality contours either by more training data or algorithm enhancements, it is reasonable to believe that our accuracies would also increase.

Dataset	Accuracy
(Non-hacked) Validation Split	76%
Hacked Image Contours	96%

Table 1 – CNN accuracies on Image Contours.

From Table 1, we can see that image contours can do a decent job at object classification with 76% accuracy for three classes. Hacked image contours had 96% accuracy which shows us that hacked images contours have almost no real differences than their original image counterparts. Therefore, given our results, it is reasonable to say that image contours can be used for classification, and that it can be used as a validation test or a sanity check for neural networks as it cannot be fooled by hacked images.

5 Neural Networks and Contour Images For Gender Classification

Another test we conducted was the effectiveness of a Neural Network trained with contour images for Gender Classification. For this task, we used PhotoSketch to generate contour images and then trained two CNNs using the two datasets, contours and real images. We propose that, the CNN trained on PhotoSketch's generated contour images could still perform classification.

5.1 Method

We used Google's TensorFlow for our CNN framework. Our CNNs architechture was composed of 4 convolutional and max pooling layers followed by fully connected layer.

5.2 Datasets

For this task, we chose a different dataset to focus on human faces. The dataset is the CelebA dataset, provided by Z. Liu [7], a compilation of celebrity face images. This dataset was chosen to test if the salient features are retained in contour images and can be used to make accurate gender predictions. We used a small sample size of 114 female and 106 male celebrity faces. We also had an underlying assumption that the image being given to the network is a human face. That is, only gender classification is the task being tested, not facial recognition. The train/test/validation split was 80/10/10 and the split was the same for both datasets.

5.3 Results

Dataset	Accuracy
CNN - Real Images	86%
CNN - Contour Images	82%

Table. 2 – CNN Real and Contour face images.

The results from testing on the second dataset are straightforward and follow our initial prediction. Contour images can be used to train CNNs, but suffer an effectivity penalty. The CNN trained with real celebrity face images (CNN-RI) performed, on average, slightly better than the CNN trained with contour images (CNN-CI). Table. 2 demonstrates one iteration of testing: the CNN-RI's accuracy was 0.86 and CNN-CI's accuracy was 0.82. Fig. 6 and Fig. 7 demonstrates two sets of images. A real celebrity face image and its PhotoSketch generated counterpart.

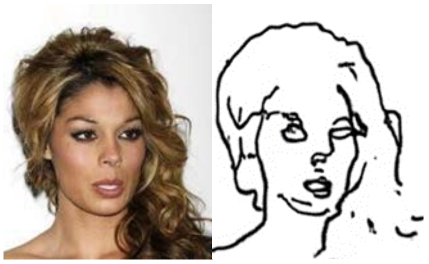

Fig. 6. – Left: Original real image. Right: Contour image

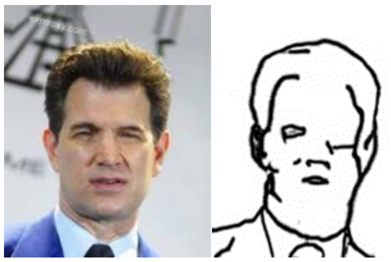

Fig. 7. – Left: Original real image. Right: Contour image

6 Conclusion And Future Work

This paper and its implementation studied the potential uses and effectiveness of image contours. We explored the problem of hacked images that plague neural networks, and how image contours could be leveraged to provide a validation test (or sanity check) on a neural network's classifications. We also explored the ability of neural network classification using only images contours. We then extended our research to use image contours for the gender classification task.

For future works, we would also like to test how well image contours can be leveraged for more difficult tasks such as age recognition.

Acknowledgment

We wish to thank Dr. Kakadiaris and Lei Shi of the University of Houston's Department of Computer Science for their support throughout this project.

References

[1] M. Li, "Photo-Sketching: Inferring Contour Drawing from Images" 2019 IEEE Winter Conference on Applications of Computer Vision. (2019) Available: https://arxiv.org/abs/1901.00542

[2] Amazon Mechanical Turk. [Online] Available: https://www.mturk.com/

[3] Adversarial Examples in the Physical World. [Online] Available: https://www.youtube.com/watch?v=zQ_uMenoBCk

[4] C. Szegedy, "Intriguing properties of neural networks" (2014) Available: https://arxiv.org/abs/1312.6199

[5] Advanced Guide to Inception v3 on Cloud TPI. [Online] Available: https://cloud.google.com/tpu/docs/inception-v3-advanced

[6] Z. Liu, "Deep Learning Face Attributes in the Wild" 2015 Proceedings of International Conference on Computer Vision (ICCV) Available: http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html

[7] Pexels. [Online] Available: https://www.pexels.com/

[8] Google Images [Online] Available: https://www.google.com/imghp?hl=EN