Detecting Photorealistic Computer-Generated Images

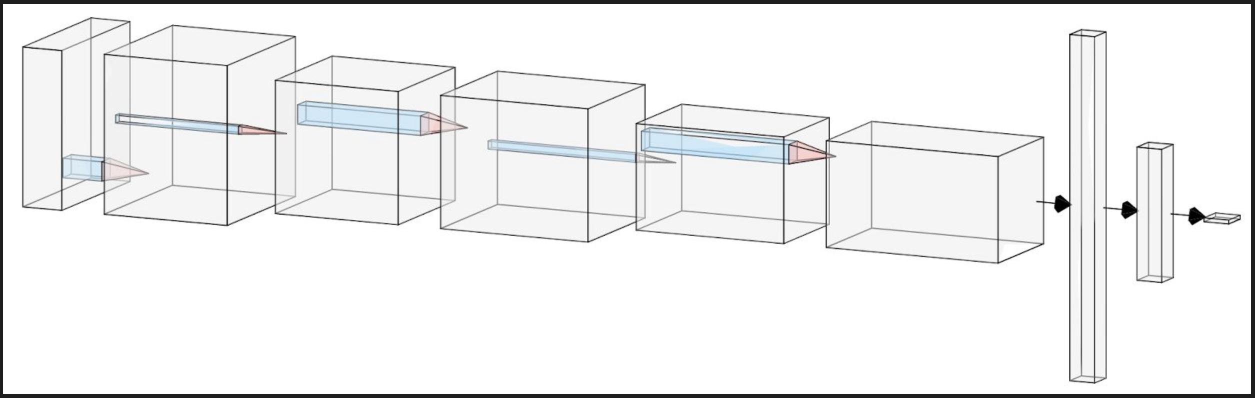
CS 221 Project Report

Shawn Zhang and Gabriel Mukobi
szhang22@stanford.edu gmukobi@stanford.edu
https://github.com/mukobi/CS221-Project
Link to Project Proposal

Model/Algorithm Proposal

Advancing from our baseline's simple logistic regression, we propose a new convolutional neural network to better distinguish between CGI and real photos.

Here we breakdown our proposed neural network:



1. Resizing of the input image as 3 x 64 x 64
2. Convolution + ReLU with 32 (5 x 5) kernels, stride of 2, and padding of 2
3. Max-pooling with (2 x 2) kernel and stride of 2
4. Convolution + ReLU with 32 (5 x 5) kernels, stride of 2, and padding of 2
5. Max-pooling with (2 x 2) kernel and stride of 2
6. Convolution + ReLU with 64 (5 x 5) kernels, stride of 2, and padding of 2
7. Max-pooling with (2 x 2) kernel and stride of 2
8. Flatten to (8192 x 1) vector
9. Fully-connected to (32 x 1) vector
10. Fully-connected to (1 x 1) output
11. Sigmoid of output

When the sigmoid's output is > 0.5, we classify the image as CGI (+1). Otherwise, we classify as real (0).

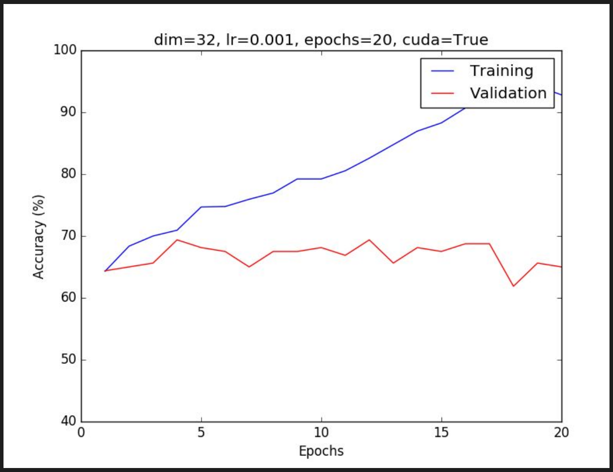
The reason why we decided to use convolutional layers for this visual task is three fold:

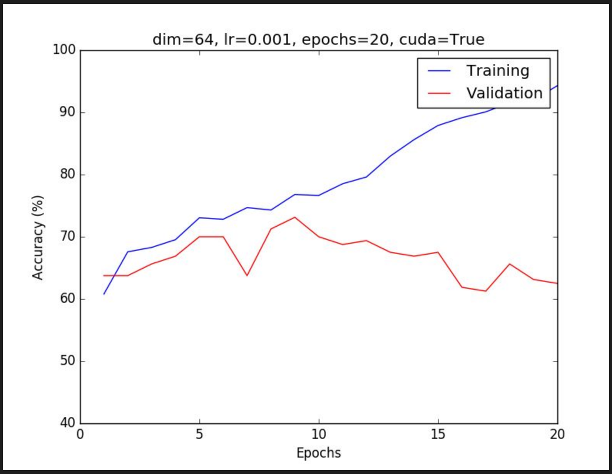
- Ability to use spatial context
- Translation invariance
- Ability to have a lot less parameters, since CNN's share weights

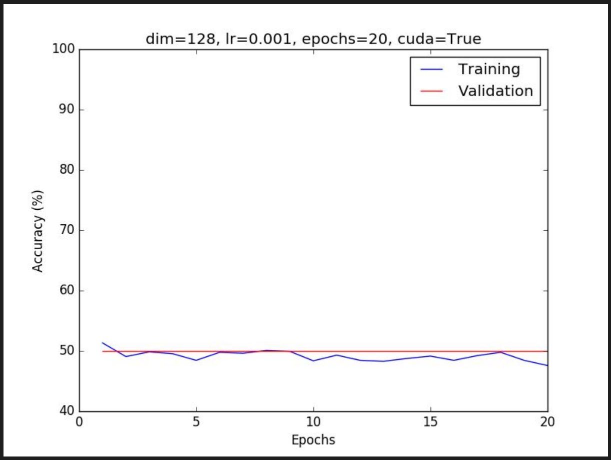
With that in mind, we have high hopes that this more complex neural network will learn better and obtain accuracies close to our oracle.

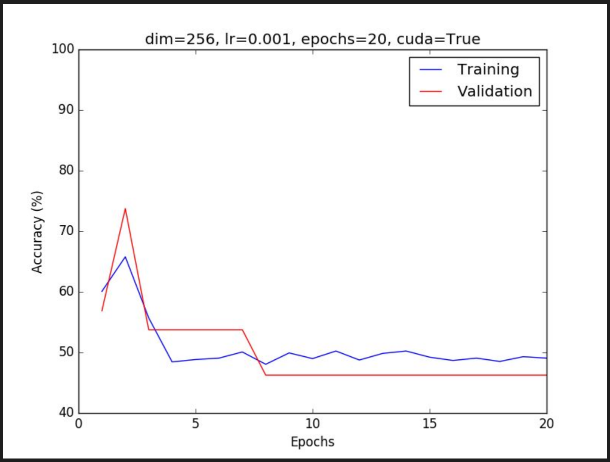
Experimentation: Hyperparameter Tuning

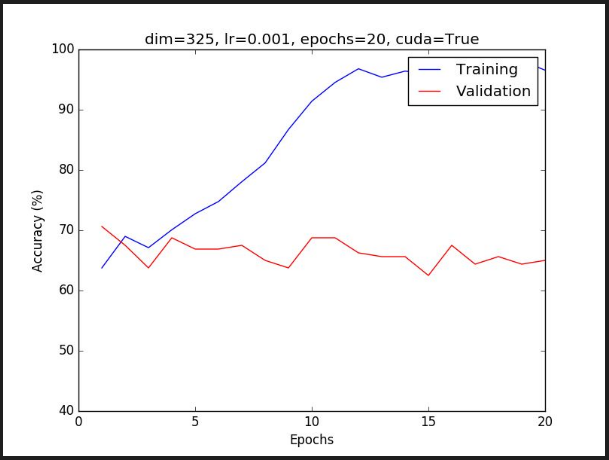
To train our neural network in hopes of obtaining better accuracy, we split our dataset to be 80% training, 10% validation, and 10% testing. We then focused on playing around with the following two hyperparameters: (1) learning rate and (2) input dimension. Here are a few plots that demonstrate our experimentation:

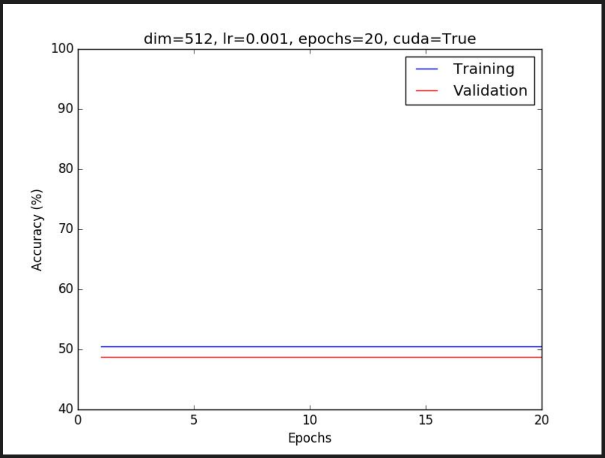




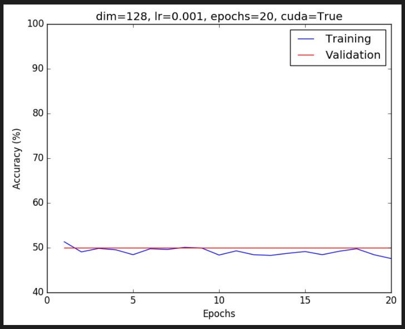


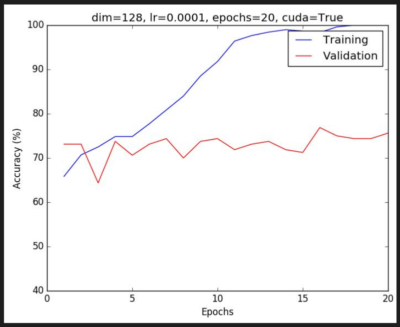


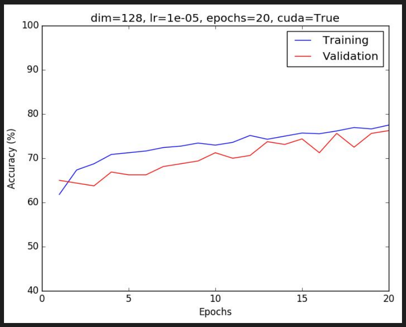




Above, we compared training and validation accuracies with a constant learning rate of 0.001 but a variable input dimension (32, 64, 128, 256, 325, 512).
Below, we compared the accuracies this time with a constant input dimension (128) but with a variable learning rate (0.001, 0.0001, 0.00001).







There are a few observations to be made here:

- These plots show that occasionally there is an early convergence problem, where the neural network gets stuck in a local minimum or saddle-point.
- It appears that input dimension does not have a large influence in affecting accuracy.
- The learning rate of 0.0001 seems to work best for learning.
- We are experiencing a variance issue, i.e. there is a huge gap between our training and our validation accuracy. We will need to address this in the future.

Looking back to our baseline classifier, we actually see a significant boost in our training accuracy (69% vs. 100%). However, our convolutional neural network appears to overfit the data when we compare our validation accuracy (69% vs. 75%). With our more complex model, this is probably to be expected.

Next Steps

More Recent Dataset

Extending from our original dataset (which had graphics dating back to 2005), we wish to see how far CGI has evolved over time. We downloaded images from more recent video games from the Level Design Reference Database (http://level-design.org/referencedb/index.php?/category/29). This contains 1832 high-resolution screenshots from modern video games that highlight level-design and in turn, high graphical fidelity. Below are two examples from this dataset:

Of course, we also added more real photos as well using the RAISE Raw Image Dataset (http://loki.disi.unitn.it/RAISE/?fbclid=IwAR3GqOj0EcnMZTBCGx2qijYuUfVTNusxC2MBpKiM-tEL7dow02ZWJ5rNCTs). This dataset contains 8156 high-resolution raw images taken by several photographers on various cameras. Below are two examples from this dataset:

In order to make this dataset smaller (the full uncompressed dataset is 350 GB!) and easier to learn on, we downloaded RAISE 2K which only include 2000 of the images and then used ImageMagick to convert all the raw .TIF images to .jpg files at 80% quality and resized them down so that the minimum dimension is 650 pixels but preserving the aspect ratio of the total image by proportionally scaling the other dimension. This brought the dataset down to 259 MB. We believe training on these higher fidelity video game and photographic images provides a tougher challenge for our machine learning and will produce a more robust model, and so we will train with this going forward.

Regularization: We experienced significant indications of overfitting in our experimentation with our approach. We know this because our training accuracy frequently reached around (or sometimes even equal to) 100% accuracy, but our validation accuracy tended to lag behind around 70-80%. We can address overfitting through regularization or by reducing the complexity of our model, but since our model currently does not seem to be too complex, we will focus on regularization. We plan to introduce batch normalization into our convolutional layers and dropout into our fully connected layers and experiment with the effect of regularizing our model.

Preprocessing: Our dataset contained images of radically different sizes. In order to feed them all into the same neural network, for each experiment, we resized all the images to one common square size (e.g. 32x32 or 512x512 pixels). This method has some complications, however, as it introduces stretching into the images by changing their aspect ratios to squares, and downsampling high-resolution images could cause a loss of high-frequency noise that might be useful in detecting CGI from photos (Yao et al.). Alternatively, we plan on trying image cropping through either randomly cropping each image to a fixed size or applying a systematic crop like 5-crop (yields 5 images produced by cropping in on each of the corners and the middle)

Hyperparameters: We saw large variances in the performance of our model depending on our hyperparameters. The learning rate seemed to be the most important in achieving high accuracies, though we also found that changing the input square dimension from resizing our images did not affect the results too much. We will continue to tweak these hyperparameters as we optimize our model.

Visualization: To better interpret what our classifier is using to distinguish between CGI and real photos, we may wish to try a saliency map to figure out which pixels are most significant. As a result, we can see which factors are a big indicator to revealing CGI, which could be important information to learn.

Bibliography

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6), 84–90. doi: 10.1145/3065386

Ng, T.-T., Chang, S.-F., Hsu, C., & Pepeljugoski, M. (2005). Columbia Photographic Images and Photorealistic Computer Graphics Dataset. *ADVENT Technical Report*, 205-2004-5.

Yao, Ye & Hu, Weitong & Zhang, Wei & Wu, Ting & Shi, Yun-Qing. (2018). Distinguishing Computer-Generated Graphics from Natural Images Based on Sensor Pattern Noise and Deep Learning. Sensors. 18. 10.3390/s18041296. Computer-generated graphics are images generated by computer software. The rapid development of computer graphics technologies has made it easier to generate photorealistic computer graphics, and these graphics are quite difficult to distinguish from natural images by our naked eyes. In this paper, we propose a method based on sensor pattern noise and deep learning to distinguish computer-generated graphics (CG) from natural images (NI). Before being fed into our convolutional neural network (CNN)-based model, these images---including the CG and NI---are clipped into image patches. Furthermore, several high-pass filters (HPF) are used to remove low-frequency signal, which represents the image content. These filters are also used to enhance the residual signal as well as sensor pattern noise introduced by the digital camera device. Different from the traditional methods of distinguishing CG from NI, the proposed method utilizes a five-layer CNN to classify the input image patches. Based on the classification results of the image patches, we deploy a majority vote scheme to obtain the classification results for the full-size images. The experiments have demonstrated that: 1) the proposed method with three high-pass filters can achieve better results than that with only one high-pass filter or no high-pass filter. 2) the proposed method with three high-pass filters achieves 100\% accuracy, although the natural images undergo a JPEG compression with a quality factor of 75.