GANs

Using the Discriminator as a Classifier

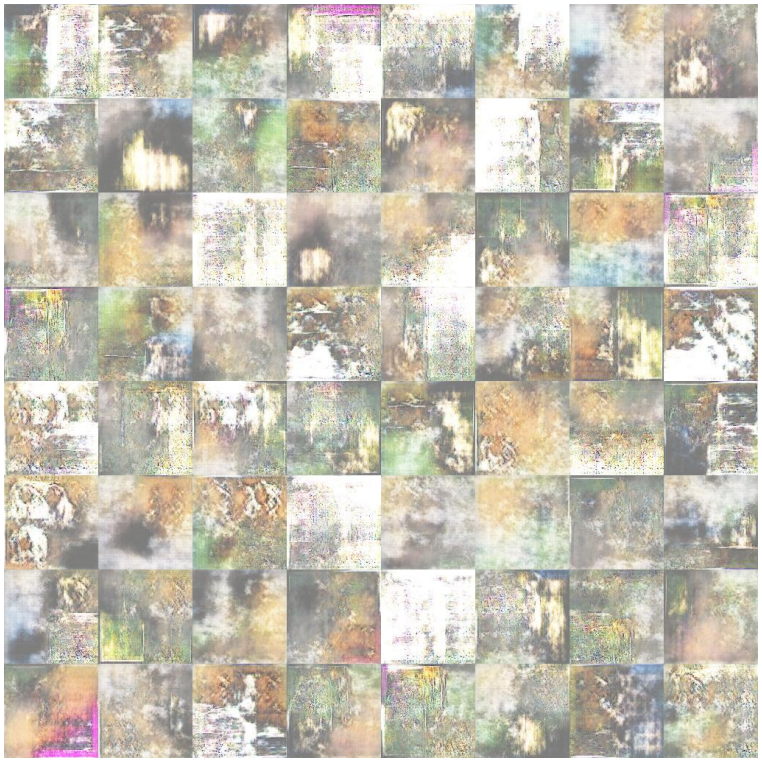
Arvi Gjoka, Rishabh Ranawat and Yun Bin Zhang

Motivation

Resilience of CNN networks like VGG, ResNet has been shown to be poor against adversarial attacks. These attacks targets gradients in the input image to try to get the greatest response from a network.

The discriminator of a GAN can be a CNN. Since it is being trained on adversarial examples, let’s examine how it performs as a CNN. If possible, let’s see if it provides any benefit to adversarial attacks.

Training a DCGAN

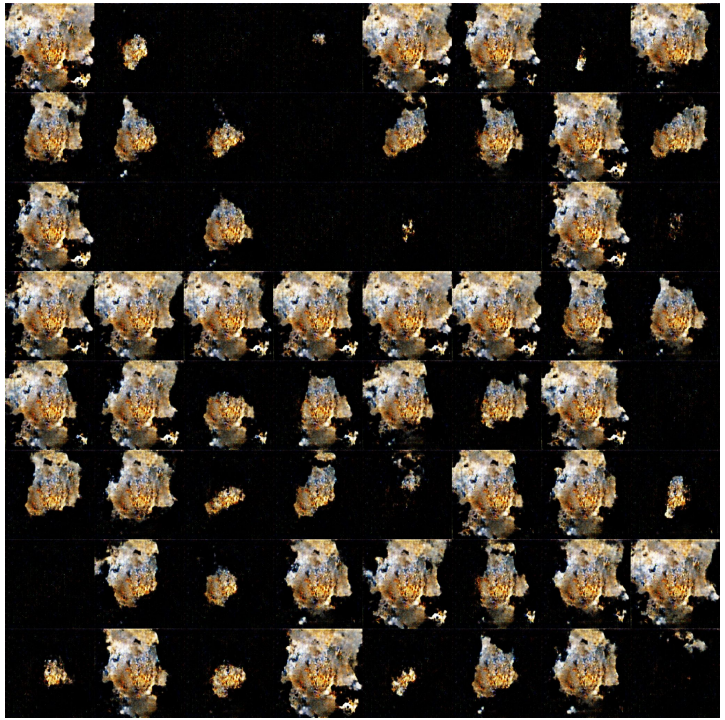


Examples

Dataset: ImageNet “Big Cats” ~ 1300
Epochs: ~4000
Batch Size: 128
Image Size: 256

Additional Preprocessing

Examples



Result

FCNN to segment the cats.
Graph Cuts for background segmentation

Dataset: ImageNet “Big Cats” ~ 1300
Epochs: ~6000
Batch Size: 128
Image Size: 256

Bad results, non smooth

Label Smoothing on a DCGAN

Dataset: Kaggle Cats ~12500
Epochs: 99
Batch Size: 128
Image Size: 64

Change scoring from 0-1 scoring to 0.1-0.9 scoring

Reduce image size

Smooth results, but abstraction was not generated

Further testing

Dataset: Kaggle Cats ~12500
Epochs: ~1100
Batch Size: 128
Image Size: 64

Smoother results, closer to “cat-like” images

However, still no clear generated cats

Training Progression

Classification Using VGG

Accuracy: 79%

Image Size: 64x64

Epoch: 50

Kaggle Cats and Dogs Dataset:

- 20000 for training
- 4000 for validation
- 1000 for test

```
    model.add(Conv2D(64, (3, 3)))
    model.add(Activation('relu'))
    model.add(MaxPooling2D(pool_size=(2, 2)))

    model.add(Flatten())
    model.add(Dense(64))
    model.add(Activation('relu'))
    model.add(Dropout(0.5))
    model.add(Dense(1))
    model.add(Activation('sigmoid'))

    if(weights_path):
        model.load_weights(weights_path)
    model.compile(loss='binary_crossentropy',
              optimizer='rmsprop',
              metrics=['accuracy'])
    return model
```

In [124]:
```
model = VGG_16('first_try.h5',64,64)
```

In [125]:
```
datagen = ImageDataGenerator(rescale=1. / 255)
generator = datagen.flow_from_directory(
        'newData/PetImages/',
        target_size=(64, 64),
        batch_size=16,
        class_mode='binary',
        shuffle=False)
```

```
Found 1000 images belonging to 2 classes.
```

In [134]:
```
evaluation = model.evaluate_generator(generator,steps=100)
```

In [135]:
```
print("Accuracy: %.3f"%(evaluation[1]))
```

```
Accuracy: 0.787
```

Classification Using Discriminator

Epochs	Seen Cats (Dataset size, Average confi	Unseen Cats	Dogs
48	12468, 50.47%	1846, 14%	12461, 14%
99	12468, 56.43%	1846, 8.9%	12461, 8.79 %
300	12468, 70.13%	1846, 7.34%	12461, 7.73%
600	12468, 64.65%	1846, 5.23%	12461, 5.47%
900	12468, 58.69%	1846, 3.63%	12461, 4.04%
1116	12468, 61.09%	1846, 3.93%	12461, 4.4%,