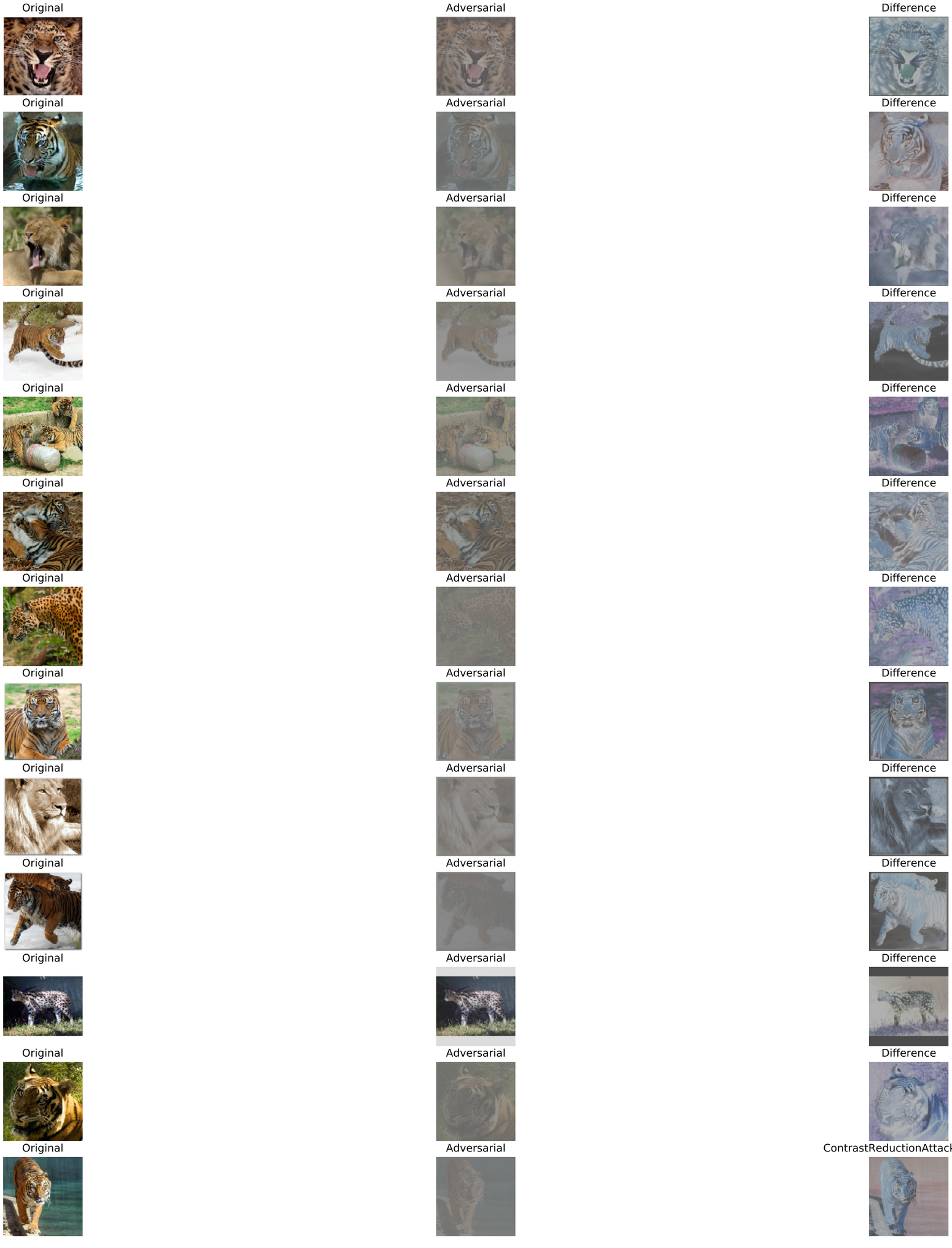
```
def main():
    import foolbox
    import keras
    import numpy as np
    from keras.applications.resnet50 import ResNet50
    from foolbox.criteria import Misclassification
    from keras.preprocessing import image
    from keras.applications.resnet50 import preprocess_input, decode_predictions
#    %matplotlib inline
    import matplotlib
    matplotlib.use('Agg')
    import matplotlib.pyplot as plt
    import numpy as np
    from keras import backend as K
    import os
    os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3'
    K.set_image_dim_ordering('tf')  # this is very important!
    from PIL import Image
    import glob

 #    image_list = image_list[:10]
    # instantiate model
    keras.backend.set_learning_phase(0)
    kmodel = ResNet50(weights='imagenet')
    preprocessing = (np.array([104, 116, 123]), 1)
    fmodel = foolbox.models.KerasModel(kmodel, bounds=(0, 255) )#,
preprocessing=preprocessing)

    # get source image and label
#     image, label = foolbox.utils.imagenet_example()

    # apply attack on source image
    # ::-1 reverses the color channels, because Keras ResNet50 expects BGR instead
of RGB
    criterion = foolbox.criteria.TopKMisclassification(1)

    image_list = []
    for filename in glob.glob('images/*.jpg'):  # assuming gif
        im = Image.open(filename)
        image_list.append(im)

#     image = np.array(image_list[3])
#     fmodel.predictions(image).argmax()
#     plt.figure()
#     plt.imshow(image)
#     plt.show()
#     plt.savefig('test.pdf')

    attack = foolbox.attacks.BlendedUniformNoiseAttack(fmodel,criterion=criterion)
    attack = foolbox.attacks.ContrastReductionAttack(fmodel,criterion=criterion)
    attack = foolbox.attacks.GradientSignAttack(fmodel,criterion=criterion)
    attack = foolbox.attacks.SaliencyMapAttack(fmodel,criterion=criterion)

    plt.figure(figsize=(25, 25))
    len = image_list.__len__()
    i = 1
    for image in image_list:
        try:
            image = np.array(image,dtype='int64')
            #label = kmodel.predict(np.expand_dims(image, 0)).argmax()
```

```
            label = fmodel.predictions(image).argmax()
            print('i is %d' %i)
            print('image label is %d' %label)

            adversarial = attack(image, label)
            #adversarial = attack(image, label)
            label = fmodel.predictions(adversarial).argmax()
            print('adversarial label is %d' %label)

            plt.subplot(len, 3, ((i-1)*3)+1)
            plt.title('Original')
            plt.imshow(image / 255)  # division by 255 to convert [0, 255] to [0,
1]
            plt.axis('off')

            plt.subplot(len, 3, ((i-1)*3)+2)
            plt.title('Adversarial')
            plt.imshow(adversarial / 255)  # ::-1 to convert BGR to RGB
            plt.axis('off')

            plt.subplot(len, 3, ((i-1)*3)+3)
            plt.title('Difference')
            difference = adversarial - image
            plt.imshow(difference / abs(difference).max() * 0.2 + 0.5)
            plt.axis('off')
            i +=1
        except TypeError:
            print('\nCould not create perturbation for sample: %d' %i)
    #plt.show()
    plt.title(attack.name() + ' results')
    plt.savefig( attack.name() + '.pdf')
    plt.close()

if __name__ == '__main__':
    main()
```

Original | Adversarial | Difference

BlendedUniformNoiseAttack results

Original | Adversarial | Difference



ContrastReductionAttack results

Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
GradientSignAttack results


Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
Difference
Original
Adversarial
SaliencyMapAttack results