Sample code:

Required dependencies: python-captcha, opencv, python-tensorflow (CPU or GPU)

Generating captchas

```
./generate.py --width 128 --height 64 --length 4 --symbols symbols.txt --count 3200 --output-dir test
```

This generates 3200 128x64 pixel captchas with 4 symbols per captcha, using the set of symbols in the `symbols.txt` file.

The captchas are stored in the folder `test`, which is created if it doesn't exist

To train and validate a neural network, we need two sets of data: a big training set, and a smaller validation set.

The network is trained on the training set, and tested on the validation set, so it is very important that there are no common images that are in both sets.

Training the neural network

```
./train.py --width 128 --height 64 --length 4 --symbols symbols.txt --batch-size 32 --epochs 5 --output-model test.h5 --train-dataset training_data --validate-dataset validation_data
```

Train the neural network for 5 epochs on the data specified. One epoch is one pass through the full dataset.

For the initial problem of captcha recognition, we train the network for 100 epochs.

Our dataset for the initial training is 128000 images, and the validation set 12800 images.

Running the classifier

```
./classify.py  --model-name test --captcha-dir ~/Downloads/validation_data/ --output ~/Downloads/stuff.txt --symbols symbols.txt
```

With `--model-name test` the classifier script will look for a model called `test.json` with weights `test.h5` in the current directory, and load the model up.

The classifier runs all the images in `--captcha-dir` through the model, and saves the file names and the model's guess at captcha contained in the image in the `--output` file.

Generate.py

```
#!/usr/bin/env python3

import os
import numpy
import random
import string
import cv2
import argparse
import captcha.image

def main():
    parser = argparse.ArgumentParser()
    parser.add_argument('--width', help='Width of captcha image', type=int)
    parser.add_argument('--height', help='Height of captcha image', type=int)
    parser.add_argument('--length', help='Length of captchas in characters', type=int)
    parser.add_argument('--count', help='How many captchas to generate', type=int)
    parser.add_argument('--output-dir', help='Where to store the generated captchas', type=str)
    parser.add_argument('--symbols', help='File with the symbols to use in captchas', type=str)
    args = parser.parse_args()

    if args.width is None:
        print("Please specify the captcha image width")
        exit(1)
```

```
if args.height is None:
    print("Please specify the captcha image height")
    exit(1)

if args.length is None:
    print("Please specify the captcha length")
    exit(1)

if args.count is None:
    print("Please specify the captcha count to generate")
    exit(1)

if args.output_dir is None:
    print("Please specify the captcha output directory")
    exit(1)

if args.symbols is None:
    print("Please specify the captcha symbols file")
    exit(1)

captcha_generator = captcha.image.ImageCaptcha(width=args.width, height=args.height)

symbols_file = open(args.symbols, 'r')
captcha_symbols = symbols_file.readline().strip()
symbols_file.close()

print("Generating captchas with symbol set {" + captcha_symbols + "}")

if not os.path.exists(args.output_dir):
    print("Creating output directory " + args.output_dir)
    os.makedirs(args.output_dir)
```

```
    for i in range(args.count):
        random_str = ''.join([random.choice(captcha_symbols) for j in range(args.length)])
        image_path = os.path.join(args.output_dir, random_str+'.png')
        if os.path.exists(image_path):
            version = 1
            while os.path.exists(os.path.join(args.output_dir, random_str + '_' + str(version) + '.png')):
                version += 1
            image_path = os.path.join(args.output_dir, random_str + '_' + str(version) + '.png')

        image = numpy.array(captcha_generator.generate_image(random_str))
        cv2.imwrite(image_path, image)

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

Train.py

./train.py --width 128 --height 64 --length 4 --symbols symbols.txt --batch-size 32 --epochs 5 --output-model test.h5 --train-dataset training_data --validate-dataset validation_data

Train the neural network for 5 epochs on the data specified. One epoch is one pass through the full dataset. For the initial problem of captcha recognition, we typicallytrain the network for 100 epochs. The dataset for the initial training should be 128000 images, and the validation set should be 12800 images.

Code

```
# Build a Keras model given some parameters

# Image sequence

#Scale to 0,1

#target device cpu/gpu
```

Classify.py

./classify.py --model-name test --captcha-dir ~/Downloads/validation_data/ --output ~/Downloads/stuff.txt --symbols symbols.txt

With `--model-name test` the classifier script will look for a model called `test.json` with weights `test.h5` in the current directory, and load the model up. The classifier runs all the images in `--captcha-dir` through the model, and saves the file names and the model's guess at captcha contained in the image in the `--output` file.

Code

Load model

Load and scale image

Classify