10-26-2018 Keras API Functions

1
```
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)))
```

2
```
model.add(layers.MaxPooling2D((2, 2)))
```

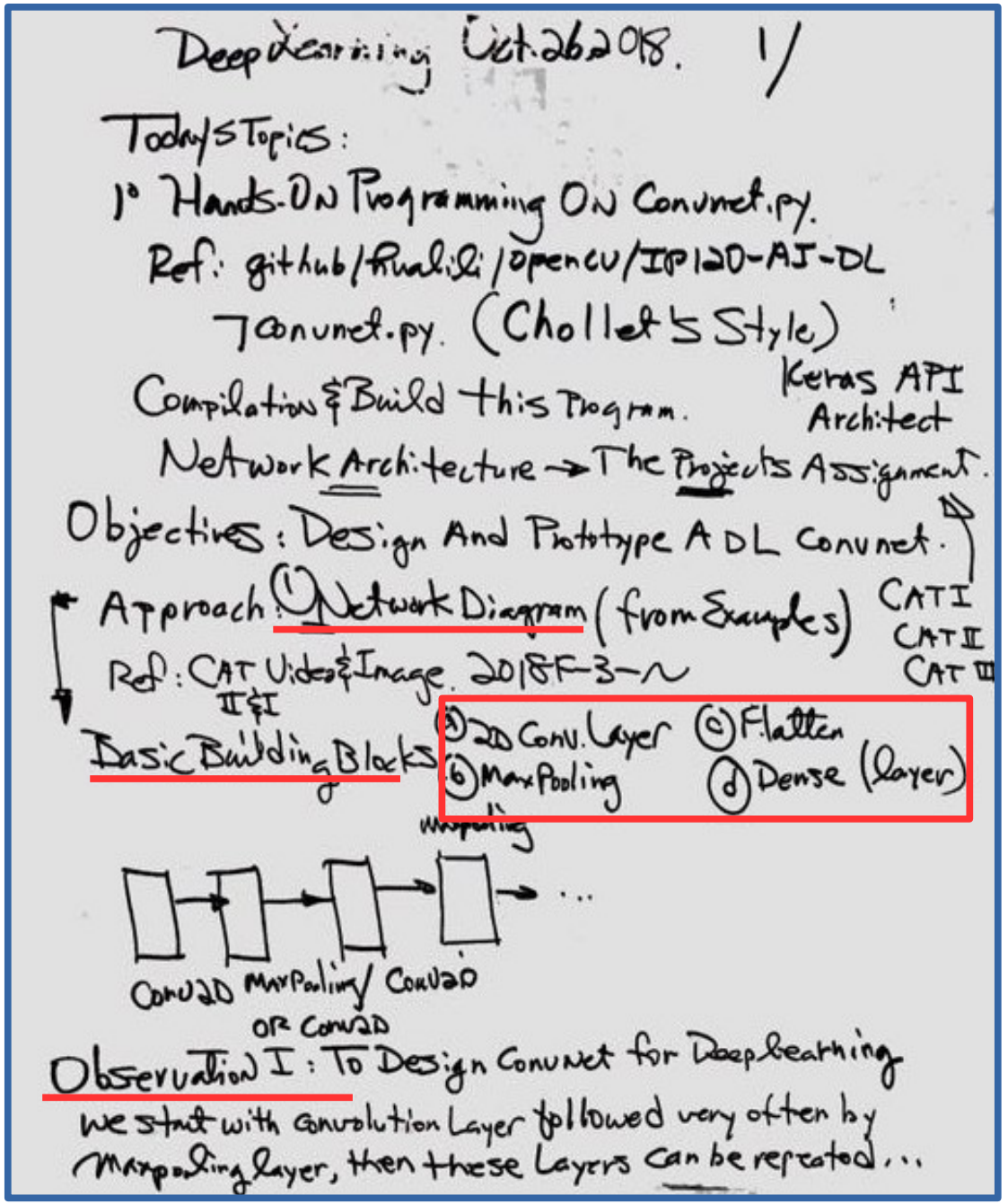
3
```
model.add(layers.Flatten())
```

4
```
model.add(layers.Dense(64, activation='relu'))
```

5
```
tf.keras.layers.Dropout(0.2)(fc_1)
```

6
```
model.summary()
```

10-26-2018 ConvNet Architecture Design



Step 1. Repeating conv2D( ) layer and MaxPooling( ) layer

Example

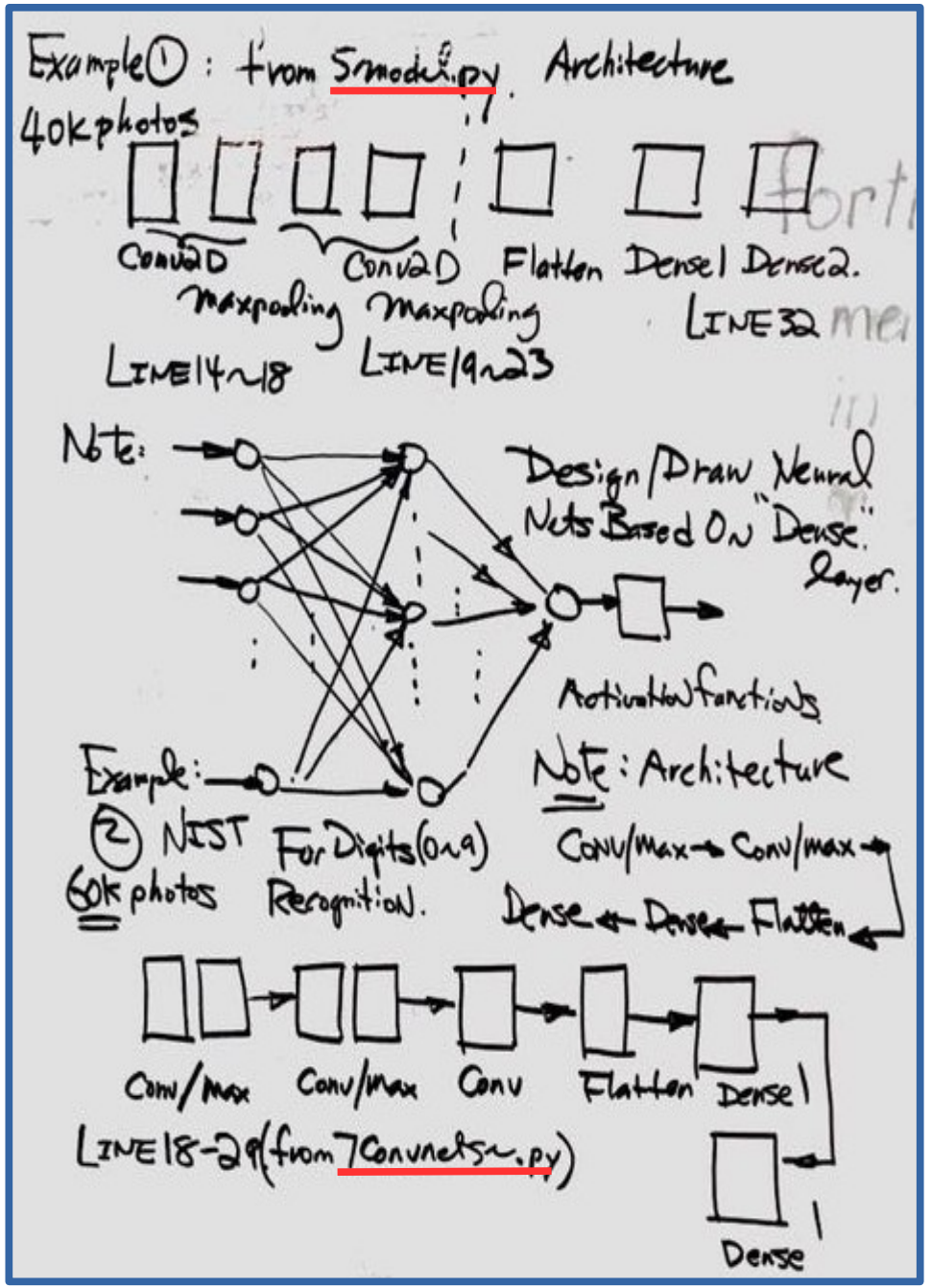
1
```
model.add(layers.Conv2D(32, (3, 3),
activation='relu', input_shape=(28, 28, 1)))
```

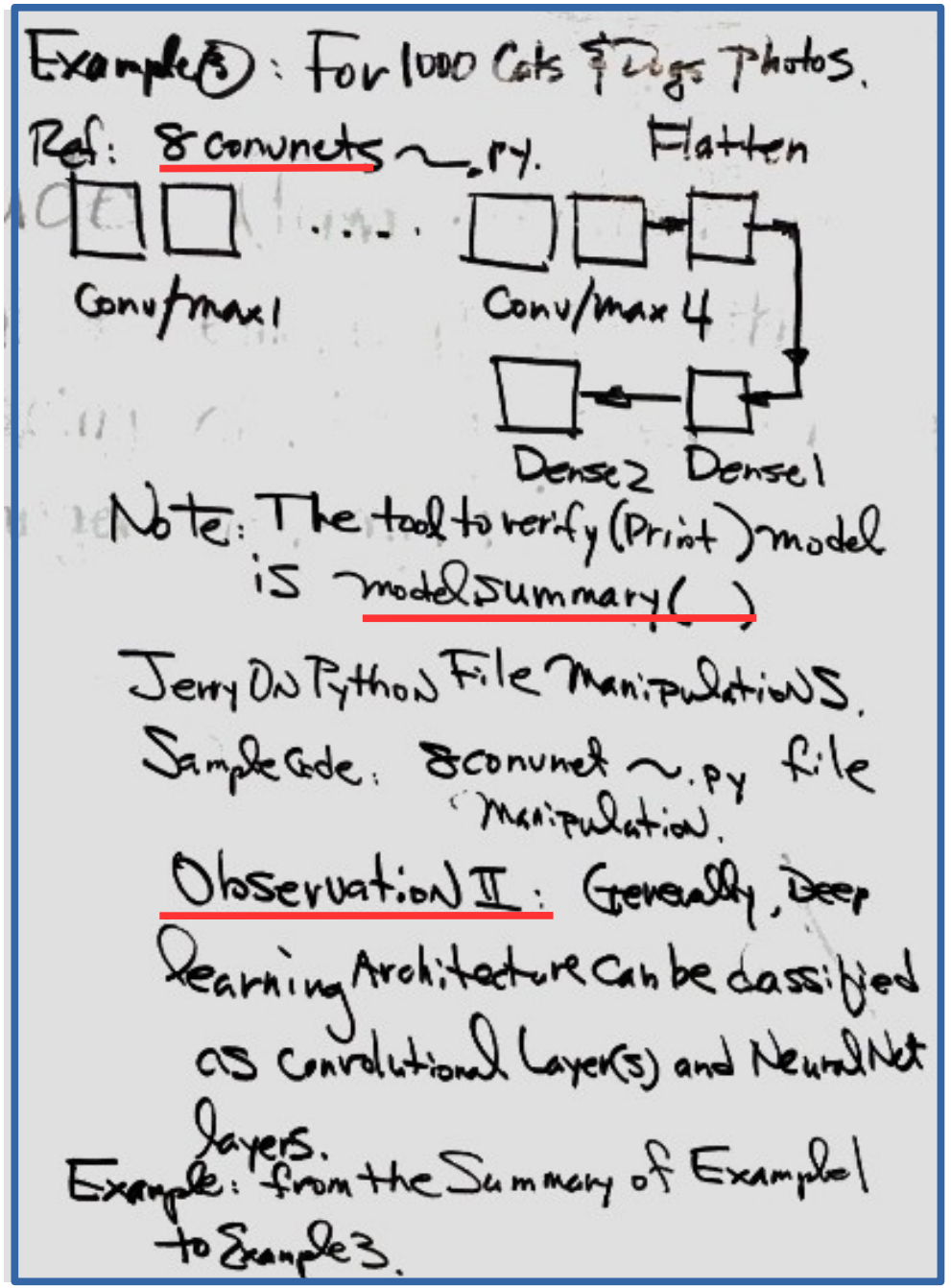
2
```
model.add(layers.MaxPooling2D((2, 2)))
```

6
```
model.summary()
```

To display the network architecture

10-26-2018 3 ConvNet Examples





Convnet MNIST Architecture

```
Using TensorFlow backend.
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 26, 26, 32)        320
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 13, 13, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 11, 11, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 5, 5, 64)          0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 3, 3, 64)          36928
=================================================================
Total params: 55,744
Trainable params: 55,744
Non-trainable params: 0
```

10-26-2018 Comparison of 3 ConvNets

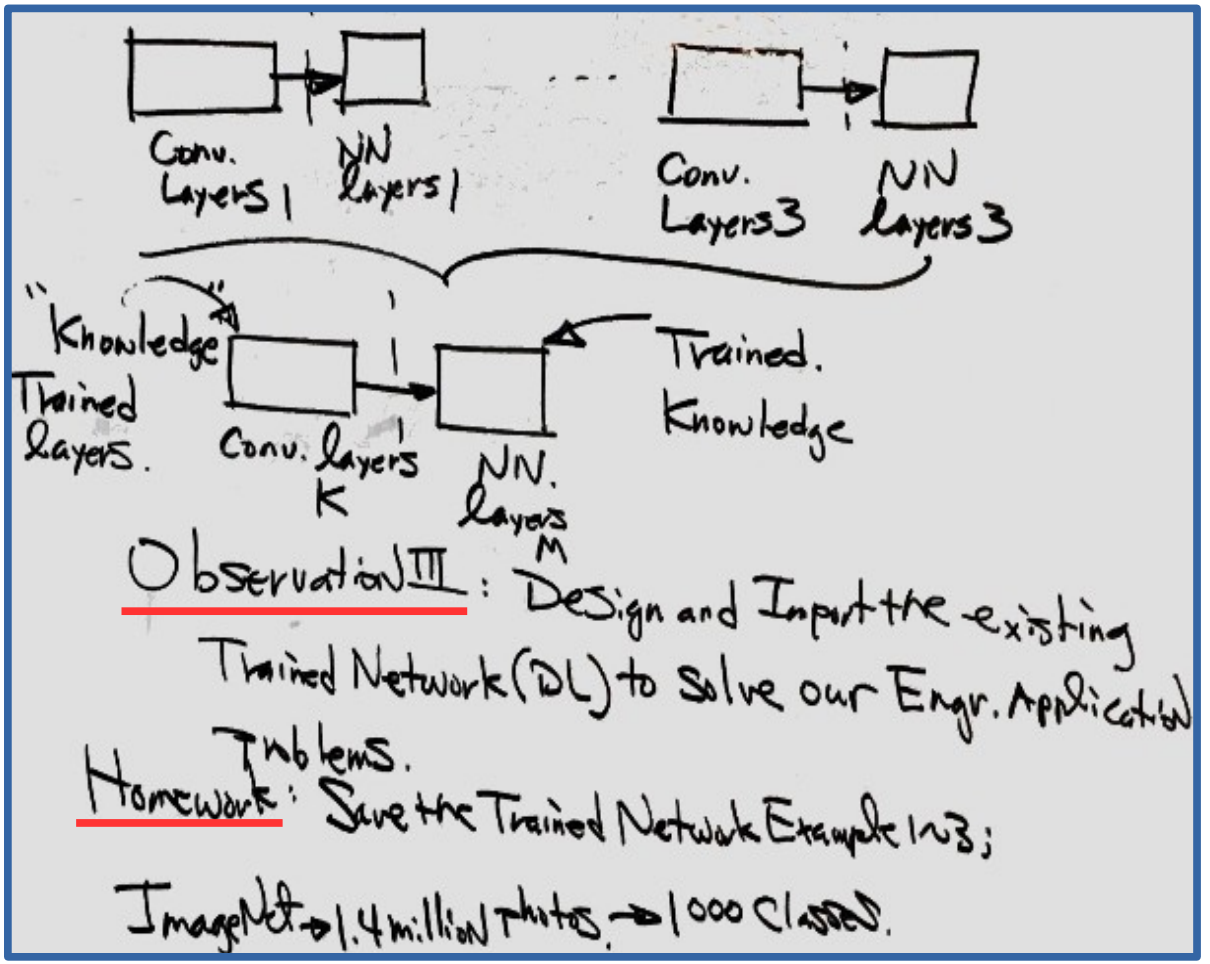
Table 1. Three well trained convnet examples

1 convnet1	Satellite imagery	40K images	ConvPool1+ConvPool2+Flatten+Den1+Den2
2 convnet2	NIST digits 0-9	60K images	ConvPool1+ConvPool2+Conv+Flatten+Den1+Den2
3 convnet3	Chollet cat-dog	1K images	ConvPool1+ConvPool2+ConvPool3 +ConvPool4 +Flatten+Den1+Den2

Table 2. Sample code for 3 well trained convnets

Network	Programs
convnet1	5model.py
convnet2	7convnets-NumeralDetection-ch05.py
convnet3	8convnets-SmallData-cats-dogs-ch05.py

10-26-2018 3 ConvNet Examples



Additional Well Developed Convnets

1 convnet1	Satellite imagery
2 convnet2	NIST digits 0-9
3 convnet3	Chollet cat-dog

	Typical applications	Architecture	No. of Images	Leader
Xception InceptionV3 ResNet50 VGG16 VGG19 MobileNet				

10-26-2018 Numerals Output Design

Example: NIST 10 digits (0-9) convnet Dense layer

```
from keras import models
from keras import layers

network = models.Sequential()
network.add(layers.Dense(512, activation='relu', input_shape=(28 * 28,)))
network.add(layers.Dense(10, activation='softmax'))
```

Sample code : https://github.com/fchollet/deep-learning-with-python-notebooks

The network consists of 2 Dense layers, densely-connected ("fully-connected") neural layers. The output layer is a 10-way "softmax" layer, it returns an array of 10 probability scores (summing to 1). Each score will be the probability that the current digit image belongs to one of our 10 digit classes.

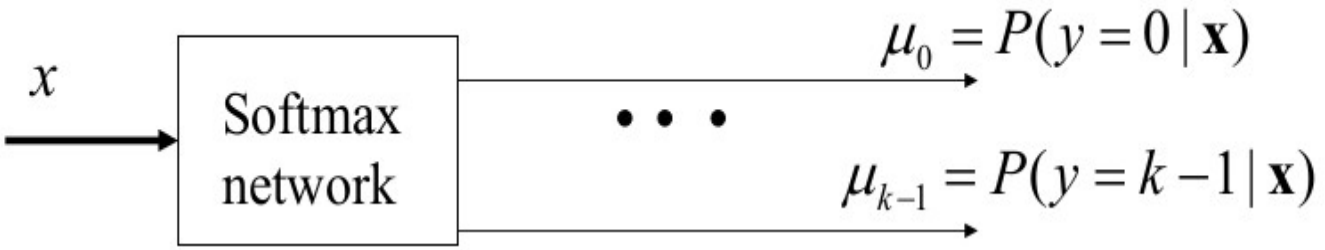


Illustration of the softmax block diagram from Milos Hauskrecht, milos@cs.pitt.edu,5329 Sennott Square

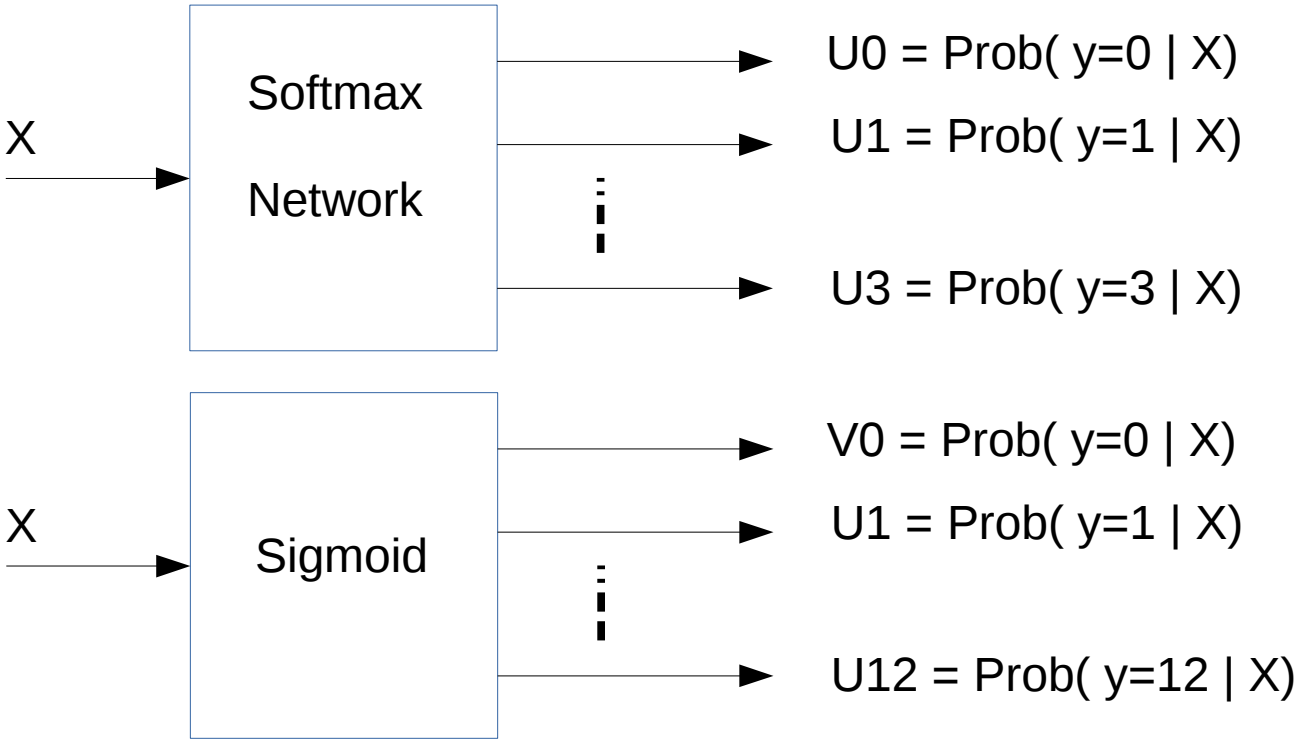
10-26-2018 Satellite Output Design

Example: Satellite Imagery Classification by convnet Dense layer

```
#separate outputs for the weather and the ground labels
weather_output = tf.keras.layers.Dense(4,
                                       activation='softmax',
                                       name='weather')(fc_2)
ground_output = tf.keras.layers.Dense(13,
                                      activation='sigmoid',
                                      name='ground')(fc_2)
```

Sample program: 5model.py

```
4
model.add(layers.Dense(64, activation='relu'))
```

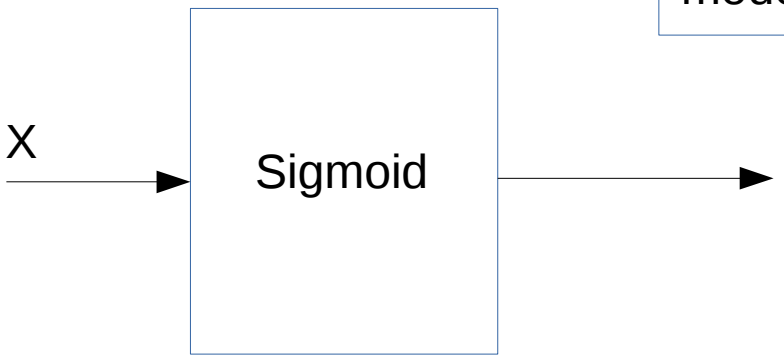


10-26-2018 Cats-Dogs Output Design

Example: Cats-dogs Imagery Classification by convnet Dense layer

Sample program: 8convnets-SmallData-cats-dogs-ch05

```
model.add(layers.Flatten())
model.add(layers.Dense(512, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```



10-26-2018 Kaggle Data and NIST Data

Example: Satellite Imagery Classification by convnet Dense layer

```
class KagglePlanetSequence(tf.keras.utils.Sequence):
```

Sample program: 5model.py

```
# original_dataset_dir = '/Users/fchollet/Downloads/kaggle_original_data'
original_dataset_dir = 'kaggle-cats-dogs'

# The directory to store smaller dataset
base_dir = 'kaggle-cats-dogs-small'
os.mkdir(base_dir)
```

Sample program: 8convnets-SmallData-cats-dogs-ch05

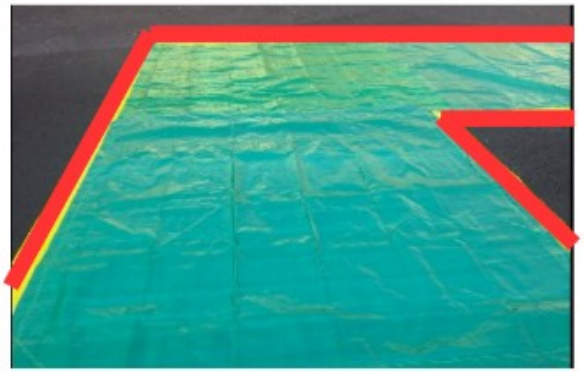
10-26-2018 Satellite Input Data Design

Example: Satellite Imagery Classification by convnet Dense layer

Sample program: 5model.py

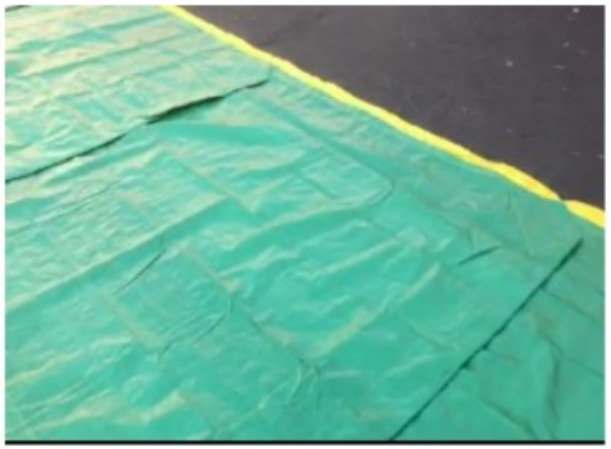
```
import tensorflow as tf
IM_SIZE = 128

image_input = tf.keras.Input(shape=(IM_SIZE, IM_SIZE, 3), name='input_layer')

# Some convolutional layers
conv_1 = tf.keras.layers.Conv2D(32,
                                kernel_size=(3, 3),
                                padding='same',
                                activation='relu')(image_input)
conv_1 = tf.keras.layers.MaxPooling2D(padding='same')(conv_1)
```

CAT-II Path Primitives

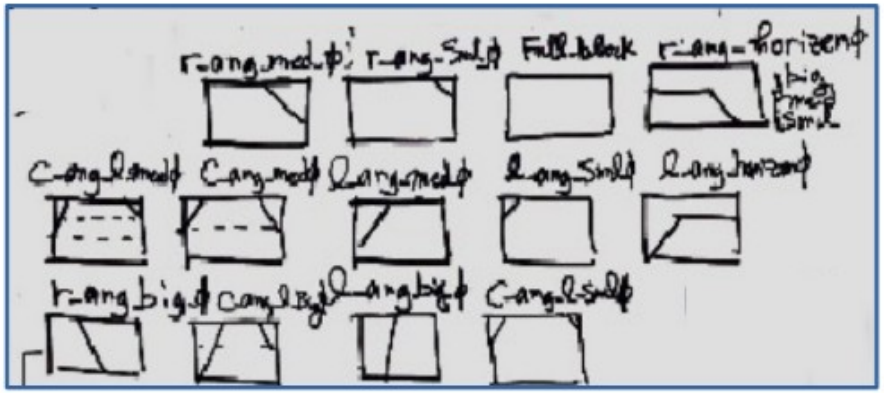


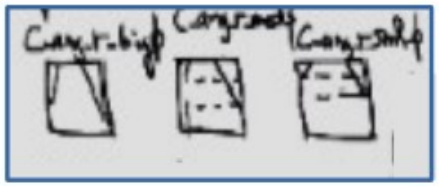






9-25-2018 Primitive Features for CAT-II Path Classification

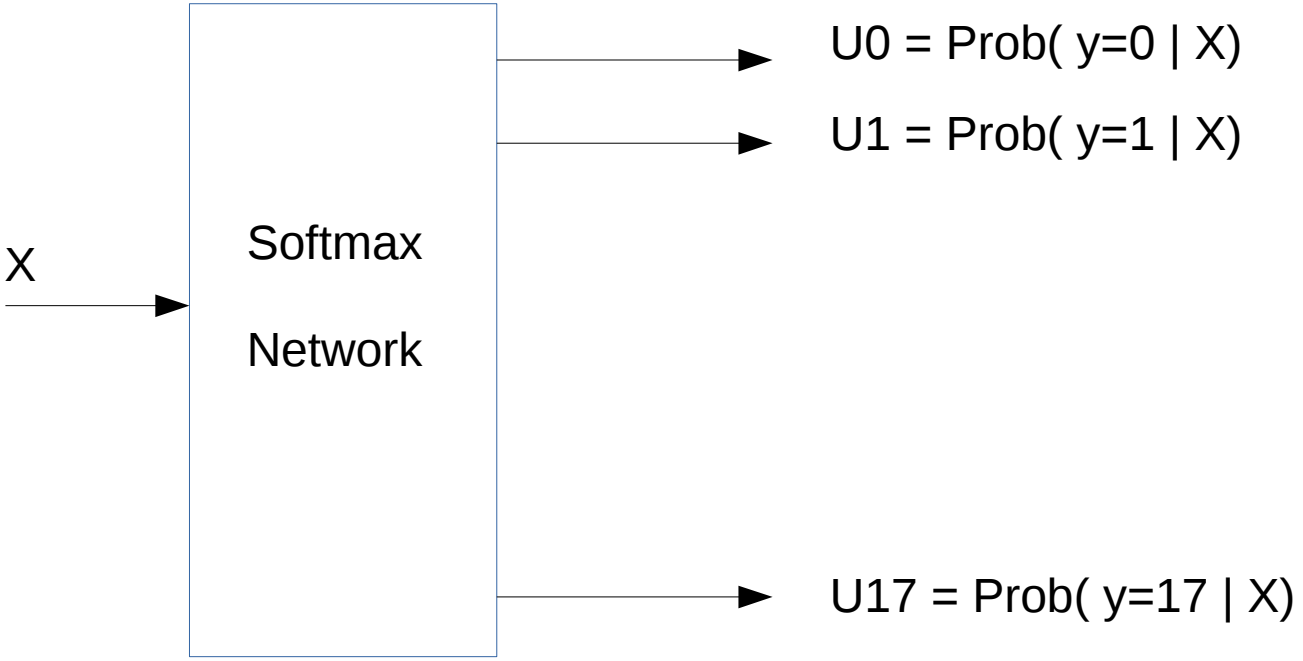




1. r_ang_big	right angle big
2. r_ang_med	right angle medium
3. r_ang_sml	right angle small
4. r_ang_hor	right angle horizon
5. l_ang_big	left angle big
6. l_ang_med	left angle medium
7. l_ang_sml	left angle small
8. l_ang_hor	left angle horizon
9. cr_ang_big	centeral-right big
10. cr_ang_med	central angle medium
11. cr_ang_sml	central angle small
12. cl_ang_big	centeral-right angle
13. cl_ang_med	central angle medium
14. cl_ang_sml	central angle small
15. c_ang_big	centeral-angle big
16. c_ang_med	central-ang medium
17. c_ang_sml	central-ang small
18. full	full block

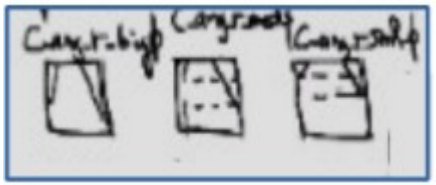
Harry Li, Ph.D. 2018

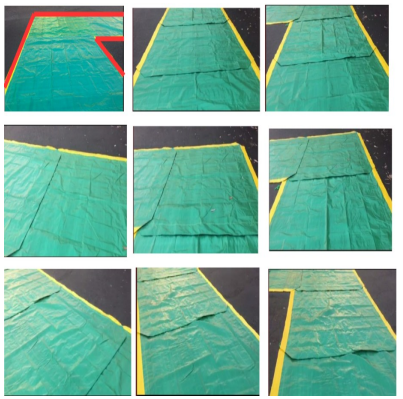
10-26-2018 Softmax Output Design for CATIII



Where Prob(y=0 | X) + Prob(y=1 | X) + … + Prob(y=17 | X) = 1

1. r_ang_big	right angle big
2. r_ang_med	right angle medium
3. r_ang_sml	right angle small
4. r_ang_hor	right angle horizon
5. l_ang_big	left angle big
6. l_ang_med	left angle medium
7. l_ang_sml	left angle small
8. l_ang_hor	left angle horizon
9. cr_ang_big	centeral-right big
10. cr_ang_med	central angle medium
11. cr_ang_sml	central angle small
12. cl_ang_big	centeral-right angle
13. cl_ang_med	central angle medium
14. cl_ang_sml	central angle small
15. c_ang_big	centeral-angle big
16. c_ang_med	central-ang medium
17. c_ang_sml	central-ang small
18. full	full block





Harry Li, Ph.D. 2018

Oct-Nov-2018 Keras API Functions

1
```
import input_data
```

2
```
model.save('harryTest.h5')
```

3
```
from keras.models import load_model
```

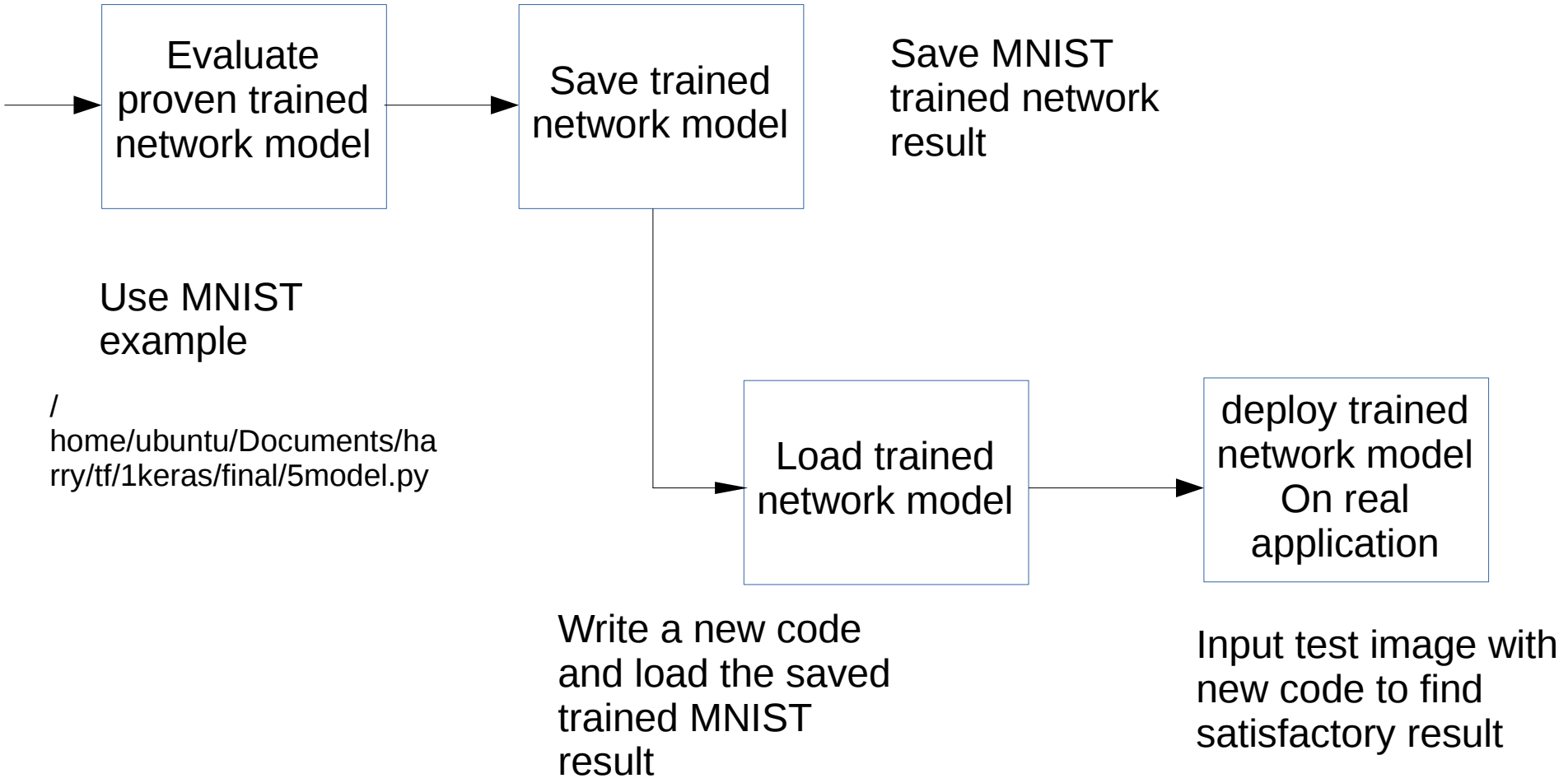
4
```
model = load_model('harryTest.h5')
```

5
```
del model
```

6
```
import h5py
h5py.run_tests()
```

7
```
result_arr = model.predict(test_image)
```

10-29-2018 Road Map to Deploy CNN



10-29-2018 Save Keras Model

https://www.google.com/search?hl=en&ei=kFDZW__GM8SS0wLLhJSICg&q=save+model+json+keras&oq=save+Model+%28JSON%29&gs_l=psy-ab.1.0.0i22i30l7.4409.4409..7830...0.0..0.66.66.1.....0....1j2..gws-wiz.....0i71.fyoCDOfN8UU

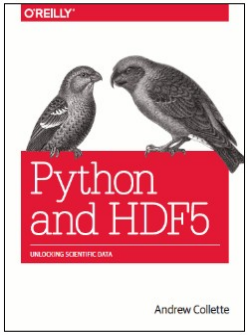
You can use model.save(filepath) to save a Keras model into a single HDF5 file which will contain: the architecture of the model, allowing to re-create the model. the weights of the model. the training configuration (loss, optimizer) the state of the optimizer, allowing to resume training exactly where you left off.

10-30-2018 Prepare h5py to Save Trained Network

https://www.quora.com/How-do-I-save-a-convolution-neural-network-model-after-training-I-am-working-in-Python

Keras for saving a model is just one line of code after training

Just make sure to have HDF5 for Python, use h5py package, which is a Pythonic interface to the HDF5 binary data format, which stores huge amounts of numerical data, and easily to be manipulated with NumPy. The files created can be exchanged including programs like IDL and MATLAB.



http://www.h5py.org/

First, install h5py, see the url link to the right and just follow the installation instructions, after the installation is done, be sure to use the following command to check:

```
import h5py
h5py.run_tests()
```

```
ubuntu@ubuntu-ThinkPad-Yoga-14:~/Documents$ python
Python 3.4.3 (default, Nov 28 2017, 16:41:13)
[GCC 4.8.4] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import h5py
>>> h5py.run_tests()
.....................................................x.................
.........x.....................................s...s......ss............
.....................................................ssssss..............
..........x....x.........................x......x.....................
.........
----------------------------------------------------------------------
Ran 457 tests in 0.632s

OK (skipped=14, expected failures=6)
<unittest.runner.TextTestResult run=457 errors=0 failures=0>
>>>
```

10-30-2018 Save Trained Network with h5py

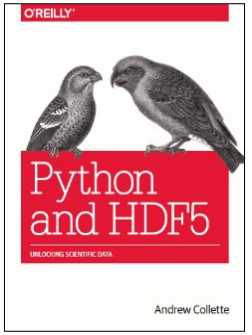
https://www.quora.com/How-do-I-save-a-convolution-neural-network-model-after-training-I-am-working-in-Python

Keras for saving a model is just one line of code after training

Model.save('harry_convnet_mnist.h5')

Example:

```
scores = model.evaluate(X_train,y_train,verbose=0)
print("Accuracy on train set: %.2f%%" % (scores[1]*100))
scores = model.evaluate(X_validation,y_validation,verbose=0)
print("Accuracy on validation set: %.2f%%" % (scores[1]*100))
scores = model.evaluate(X_test,y_test,verbose=0)
print("Accuracy on test set: %.2f%%" % (scores[1]*100))
model.save('ajits_model_d6_256.h5')
```



http://www.h5py.org/

Example: 7-1convnets-NumeralDet-saveTrained.py

```
#-------------save trained model---------*
import h5py
model.save('harryTest.h5')
#-end
```

Just add 2 lines of code

10-30-2018 Load Trained Network

https://stackoverflow.com/questions/35074549/how-to-load-a-model-from-an-hdf5-file-in-keras

Keras for loading a model is just one line of code

If you stored the complete model, not only the weights, in the HDF5 file, then load is as simple as

Example:

```
#----------load trained model--------------------*
from keras.models import load_model
model = load_model('harryTest.h5')
model.summary() #check the model
```

```
>>> model.summary()
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 26, 26, 32)        320
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 13, 13, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 11, 11, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 5, 5, 64)          0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 3, 3, 64)          36928
_________________________________________________________________
flatten_1 (Flatten)          (None, 576)               0
_________________________________________________________________
dense_1 (Dense)              (None, 64)                36928
_________________________________________________________________
dense_2 (Dense)              (None, 10)                650
=================================================================
Total params: 93,322
Trainable params: 93,322
Non-trainable params: 0
```

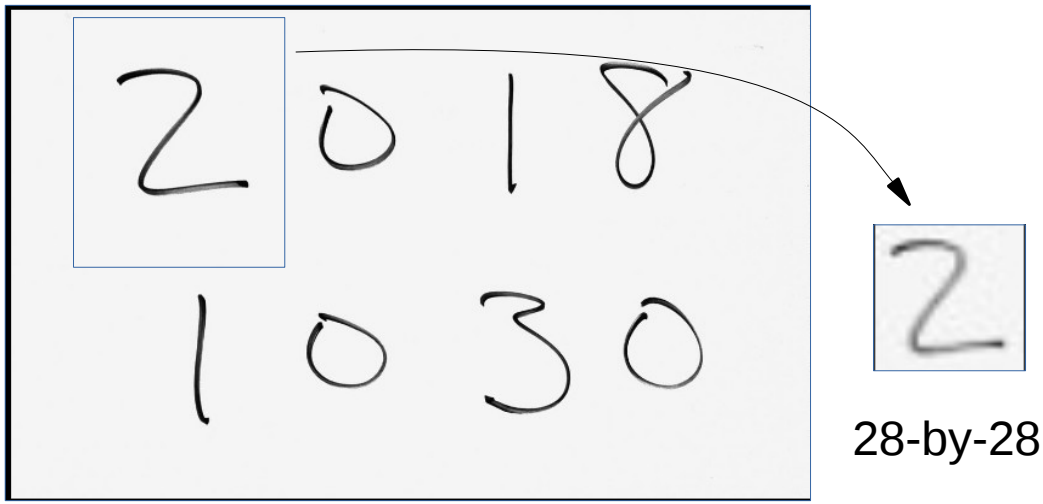
Prepare Image For Testing Trained MNIST convnet

Find out the input image size to MNIST convnet, from 2 sources: (1) from 7convnets-NumeralDetection-ch05.py (code below) (2) from the code, the dimension of the conv2d_1, kernel size is 3x3, from model.summary, the output shape is 26x26.

Example:

```
model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)))
```

So, (1) use screen capture program to locate ROI as shown below, and then (2) we wrote opencv program to take any size input image and convert it to 28-by-28.



28-by-28

```
>>> model.summary()
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 26, 26, 32)        320
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 13, 13, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 11, 11, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 5, 5, 64)          0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 3, 3, 64)          36928
_________________________________________________________________
flatten_1 (Flatten)          (None, 576)               0
_________________________________________________________________
dense_1 (Dense)              (None, 64)                36928
_________________________________________________________________
dense_2 (Dense)              (None, 10)                650
=================================================================
Total params: 93,322
Trainable params: 93,322
Non-trainable params: 0
```

OpenCV Program To Resize 28x28 Images

7-3ResizeImage.py

```
import cv2
import numpy as np

gray = cv2.imread('test.png',cv2.IMREAD_GRAYSCALE)
gray = cv2.resize(gray, (28,28))
cv2.imshow('image',gray)
cv2.imwrite('resized_harryTest.jpg',gray, [int(cv2.IMWRITE_JPEG_QUALITY),90])

k = cv2.waitKey(0)
if k == 27:         # wait for ESC key to exit
    cv2.destroyAllWindows()
```

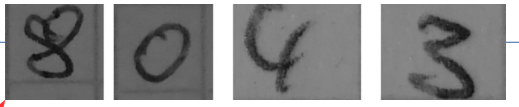
11-2-2018 Deploy MNIST with Test Image (1)

Example:

```
model = load_model('Numeral_detector.h5')
test_image= cv2.imread('resized_LCTest.jpg', cv2.IMREAD_GRAYSCALE)
test_image = np.reshape(test_image, [1, 28, 28, 1])
result_arr = model.predict(test_image)
result = model.predict_classes(test_image)
print(result_arr,' ', result)
```

10-29-2018 Deploy MNIST with Test Image (1)

https://medium.com/@o.kroeger/tensorflow-mnist-and-your-own-handwritten-digits-4d1cd32bbab4



8.png 0.png 4.png 3.png

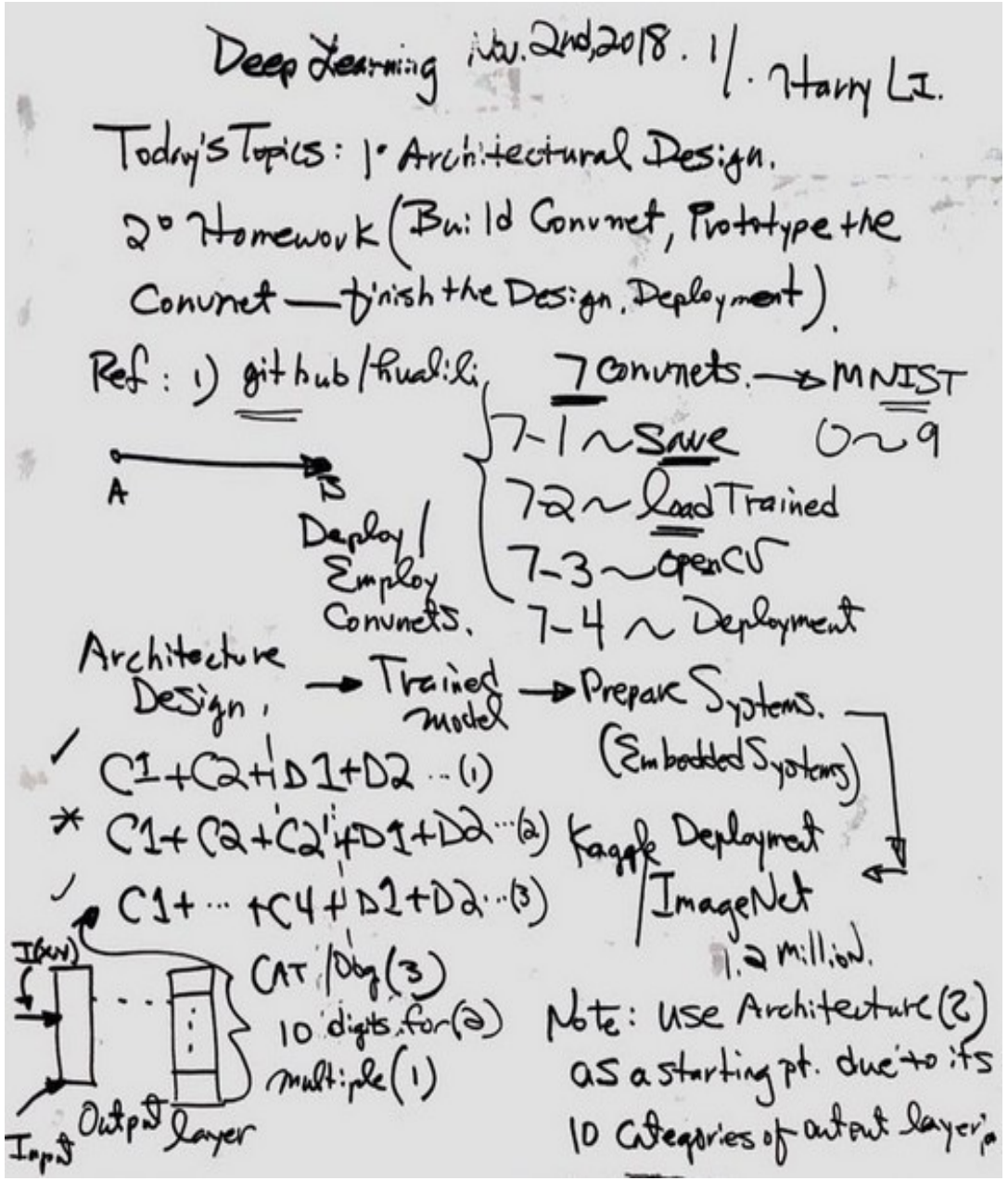
```
# create an array where to store 4 pics with one numeral each
# each image consists of total 784 pixels
images = np.zeros((4,784))
# and the correct values
correct_vals = np.zeros((4,10))

# test images 8, 0, 4, 3
i = 0
for no in [8,0,4,3]:
    gray = cv2.imread("img/blog/own_"+str(no)+".png", cv2.CV_LOAD_IMAGE_GRAYSCALE)

    # resize the images and invert it (black background)
    gray = cv2.resize(255-gray, (28, 28))

    # save the processed images
    cv2.imwrite("pro-img/image_"+str(no)+".png", gray)
    """
    all images in the training set ranges from 0-1
    not from 0-255 so divide the flatten images
    (a one dimensional vector with 784 pixels)
    """
    flatten = gray.flatten() / 255.0
```

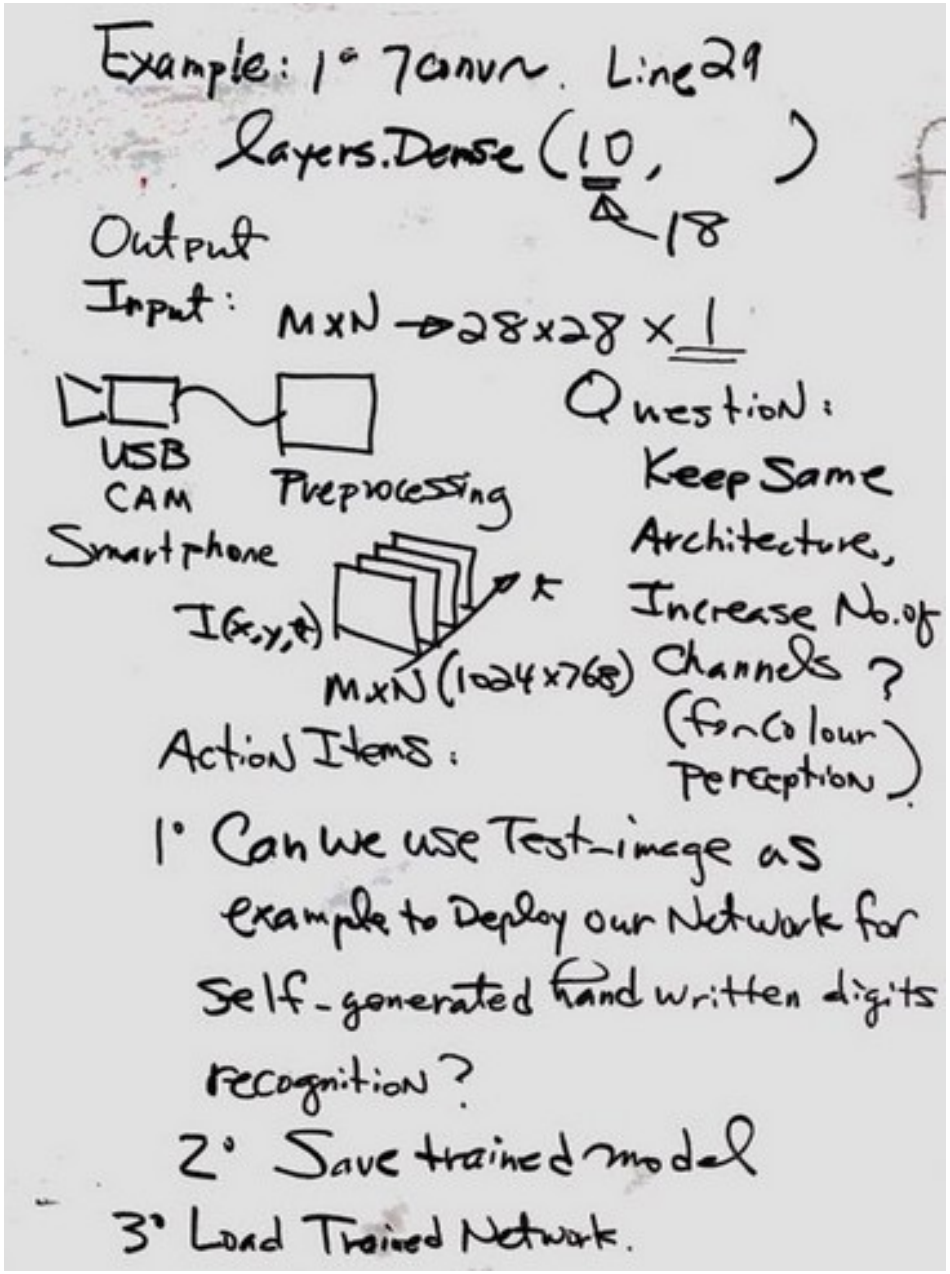
Image size: 28x28 = 784

10-29-2018 Deploy MNIST with Test Image (2)

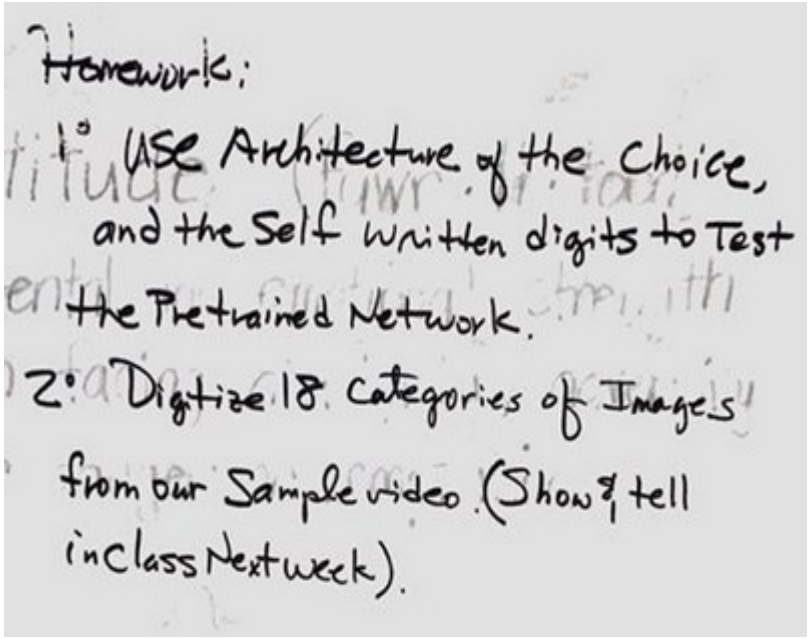
```
    """–---------------- store flatten image and generate----------------
    the correct_vals array
    correct_val for the first digit (9) would be
    [0,0,0,0,0,0,0,0,0,1]
       ----------------------------------------------------------------------"""
    images[i] = flatten
    correct_val = np.zeros((10))
    correct_val[no] = 1
    correct_vals[i] = correct_val
    i += 1
""" –-------the prediction will be an array with four values-----------
    which show the predicted number
    –----------------------------------------------------------------------"""
prediction = tf.argmax(y,1)

"""–-------run the prediction and the accuracy function----------------
using our generated arrays (images and correct_vals)
    –----------------------------------------------------------------------"""
print sess.run(prediction, feed_dict={x: images, y_: correct_vals})
print sess.run(accuracy, feed_dict={x: images, y_: correct_vals})
```

11-2-2018 Architecture Aspects on Deployment





11-2-2018 Architecture Aspects on Deployment



1
```
from keras import models
from keras import layers
from keras import optimizers
```

2
```
import matplotlib.pyplot as plt

plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.legend()
plt.figure()
plt.show()
```

3
```
model = models.Sequential()
model.add(layers.Dense(256, activation='relu', input_dim=4 * 4 * 512))
model.add(layers.Dropout(0.5))
model.add(layers.Dense(1, activation='sigmoid'))
```

4
```
model.compile(optimizer=optimizers.RMSprop(lr=2e-5),
              loss='binary_crossentropy',
              metrics=['acc'])
```

11-2-2018 Architecture Aspects on Deployment

1
```
from keras import models
from keras import layers
```

2
```
model = models.Sequential()
model.add(conv_base)
model.add(layers.Flatten())
model.add(layers.Dense(256, activation='relu'))
```

11-2-2018 Down Load Jupyter Notebooks in github

https://github.com/mmcky/nyu-econ-370/issues/10

Download an ipynb file using the RAW button in GitHub it displays the text in the browser. This is json code and contains all of the jupyter notebook cells. Copy this text into a file and name it with the extension ipynb and it will contain the notebook when using it with Jupyter.

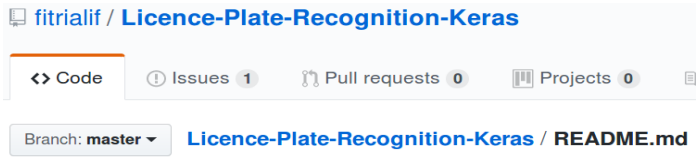
11-6-2018 Keras Supported CNN Architectures

Xception
InceptionV3
ResNet50
VGG16
VGG19
MobileNet

11-6-2018 Keras Supported MobileNet Architectures

11-6-2018 License Plate Recognition (1) Keras

https://github.com/fitrialif/Licence-Plate-Recognition-Keras/blob/master/README.md



128 x 64

Example: Architecture below, source code see the url

https://github.com/fitrialif/Licence-Plate-Recognition-Keras/blob/master/image_ocr.ipynb

Layer (type)	Output Shape	Param #	Connected to
the_input (InputLayer)	(None, 128, 64, 1)	0	
conv1 (Conv2D)	(None, 128, 64, 16)	160	the_input[0][0]
max1 (MaxPooling2D)	(None, 64, 32, 16)	0	conv1[0][0]
conv2 (Conv2D)	(None, 64, 32, 16)	2320	max1[0][0]
max2 (MaxPooling2D)	(None, 32, 16, 16)	0	conv2[0][0]
reshape (Reshape)	(None, 32, 256)	0	max2[0][0]
dense1 (Dense)	(None, 32, 32)	8224	reshape[0][0]
gru1 (GRU)	(None, 32, 512)	837120	dense1[0][0]
gru1_b (GRU)	(None, 32, 512)	837120	dense1[0][0]
add_1 (Add)	(None, 32, 512)	0	gru1[0][0] gru1_b[0][0]
gru2 (GRU)	(None, 32, 512)	1574400	add_1[0][0]
gru2_b (GRU)	(None, 32, 512)	1574400	add_1[0][0]
concatenate_1 (Concatenate)	(None, 32, 1024)	0	gru2[0][0] gru2_b[0][0]
dense2 (Dense)	(None, 32, 23)	23575	concatenate_1[0][0]
softmax (Activation)	(None, 32, 23)	0	dense2[0][0]

C1-M1
C2-M2
D1
GRU1
GRU1b
ADD
GRU2
GRU2b
D2

This example does not have data set, therefore it is not verifiable

Master Source of reference:

https://github.com/topics/plate-recognition

11-6-2018 Chinese License Plate Recognition Keras

https://github.com/zeusees/HyperLPR

pip 一键安装、更新的新的识别模型、倾斜车牌校正算法、定位算法 (2018.08.11)

增加PHP车牌识别工程@coleflowers (2018.06.20)

添加了 HyperLPR Lite 仅仅需 160 行代码即可实现车牌识别 (2018.3.12)

提供精确定位的车牌矩形框 (2018.3.12)

增加了端到端模型的 cpp 实现 (Linux)(2018.1.31)

感谢 sundyCoder Android 字符分割版本

增加字符分割训练代码和字符分割介绍(2018.1.)

更新了 Android 实现，大幅提高准确率和速度（骁龙 835 (720x1280) ~50ms)(2017.12.27)

添加端到端的序列识别模型识别率大幅度提升，使得无需分割字符即可识别，识别速度提高 20% (2017.11.17)

新增的端到端模型可以识别新能源车牌、教练车牌、白色警用车牌、武警车牌 (2017.11.17)

增加 cpp 版本，目前仅支持标准蓝牌（需要依赖 OpenCV 3.3) (2017.10.28)

特性

720p, 单核 Intel 2.2G CPU (MaBook Pro 2015) 平均识别时间低于 100ms

基于端到端的车牌识别无需进行字符分割，识别率高，卡口场景准确率在 95%-97% 左右；轻量，总代码量不超 1k 行

TODO

支持多种车牌以及双层

支持大角度车牌

deep learning based 实时检测模型

轻量级识别模型

模型资源说明

cascade.xml 检测模型 - 目前效果最好的 cascade 检测模型

cascade_lbp.xml 召回率效果较好，但其错检太多

char_chi_sim.h5 Keras 模型 - 可识别 34 类数字和大写英文字 使用 14W 样本训练

char_rec.h5 Keras 模型 - 可识别 34 类数字和大写英文字 使用 7W 样本训练

ocr_plate_all_w_rnn_2.h5 基于 CNN 的序列模型

ocr_plate_all_gru.h5 GRU 的序列模型从 OCR 模型修改，效果好但速度较慢，需 20ms。 plate_type.h5 用于车牌颜色判断的模型；model12.h5 左右边界回归模型

Harry Li, Ph.D. 2018

11-6-2018 Chinese License Plate Recognition Keras

https://github.com/zeusees/HyperLPR

可识别和待支持的车牌的类型

- 单行蓝牌
- 单行黄牌
- 新能源车牌
- 白色警用车牌
- 使馆 / 港澳车牌
- 教练车牌
- 武警车牌
- 民航车牌
- 双层黄牌
- 双层武警
- 双层军牌
- 双层农用车牌
- 双层个性化车牌

An easy, flexible, and accurate plate recognition project for Chinese licenses in unconstrained situations.

11-6-2018 Chinese License Plate Recognition

https://github.com/liuruoze/EasyPR

EasyPR: 开源的中文车牌识别系统，其目标是成为一个简单、高效、准确的非限制场景 (unconstrained situation) 下的车牌识别库。相比于其他的车牌识别系统， EasyPR 有如下特点：

它基于 openCV 这个开源库。这意味着你可以获取全部源代码，并且移植到 opencv 支持的所有平台。

它能够识别中文。例如车牌为苏 EUK722 的图片，它可以准确地输出 std:string 类型的 " 苏 EUK722" 的结果。

它的识别率较高。图片清晰情况下，车牌检测与字符识别可以达到 80% 以上的精度。

本次更新版本是 1.6 正式版本，主要有以下几点更新：

修正了多项 readme 的文本提示 . 增加了 C# 调用 EasyPR 的一个项目的链接，感谢 @zhang-can 同学。

注意： 对于 Opencv3.2 或以上版本，如果碰到编译问题，例如“ ANN_MLP” 相关的错误，尝试将 config.h 中将 #define CV_VERSION_THREE_ZERO 改为 #define CV_VERSION_THREE_TWO 试试 . linux 系统推荐使用 Opencv3.2 以上版本。 3.2 以下的版本例如 3.0 和 3.1 在识别时可能会出现车牌识别结果为空的情况。稳妥起见，建议都升级到最新的 3.2 版本。 Windows 版本没有这个问题。

待做的工作

完成一个 CNN 框架

替换 ANN 为 CNN

增加新能源车的识别（待定）

增加两行车牌的识别（待定）

Harry Li, Ph.D. 2018

11-6-2018 Facial Recognition Keras

https://github.com/krasserm/face-recognition

11-9-2018 Python Imports for Augmentation To Create Train Data Set (1)

https://leemendelowitz.github.io/blog/how-does-python-find-packages.html

Python finds packages: Python imports by searching the directories listed in sys.path

```
>>> import sys
>>> print '\n'.join(sys.path)

/usr/lib/python2.7
/usr/lib/python2.7/plat-x86_64-linux-gnu
/usr/lib/python2.7/lib-tk
/usr/lib/python2.7/lib-old
/usr/lib/python2.7/lib-dynload
/usr/local/lib/python2.7/dist-packages
/usr/lib/python2.7/dist-packages
/usr/lib/python2.7/dist-packages/PILcompat
/usr/lib/python2.7/dist-packages/gtk-2.0
/usr/lib/python2.7/dist-packages/ubuntu-sso-client
>>>
```

sys.path.append() adds the path for the code to access.

11-9-2018 Python Common Path Name Manipulations

https://docs.python.org/3/library/os.path.html

posixpath for UNIX-style paths
ntpath for Windows paths
macpath for old-style MacOS paths
os2emxpath for OS/2 EMX paths

Example: check if in the right directory

```
>>> print os.path.isdir(data_dir)
True
>>>
```

Example: to join the directory path

```
>>> path = os.path.join('CTIOne_harry_Training_Data', 'level1','level2')
>>> print path
CTIOne_harry_Training_Data/level1/level2
>>>
```

Example: List what is in the directory. (Just like ls command)

```
>>> print os.listdir(data_dir)
['test.txt']
```