Assignment6.2B Sutow Brett

July 15, 2021

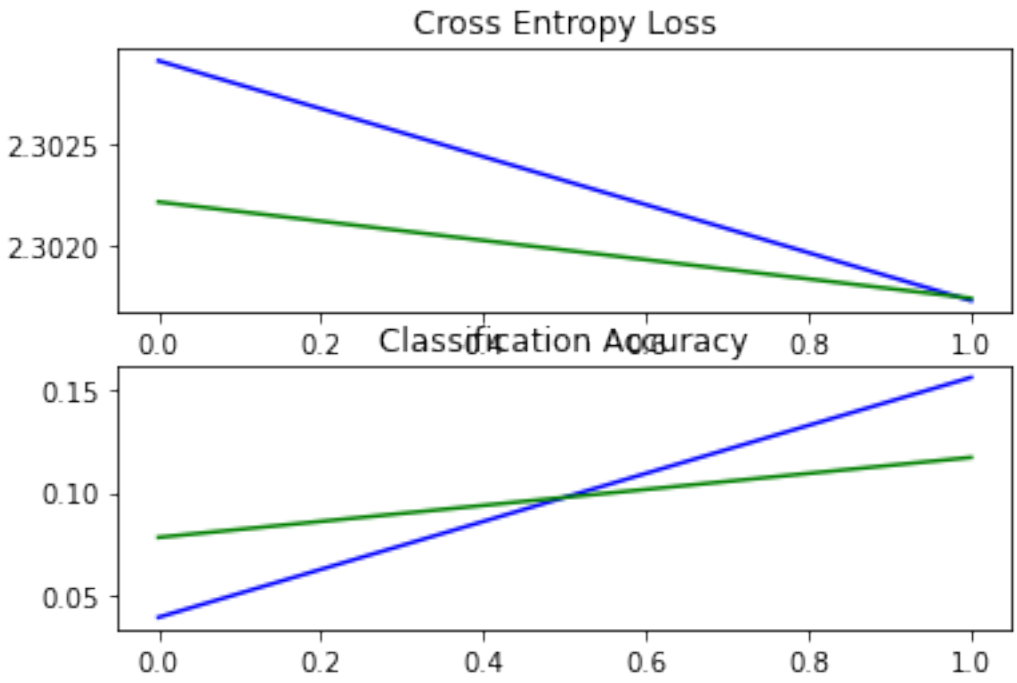
[1]:
```
#Assignment 6.2B#
from pathlib import Path
import numpy as np
import sys
from contextlib import redirect_stdout
from matplotlib import pyplot
import itertools
import matplotlib.pyplot as plt

from keras.datasets import cifar10
from keras.utils import to_categorical
from keras.models import Sequential
from keras.layers import Conv2D
from keras.layers import MaxPooling2D
from keras.layers import Dense
from keras.layers import Dropout
from keras.layers import Flatten
from keras.optimizers import SGD

from keras.preprocessing.image import load_img
from keras.preprocessing.image import img_to_array
from keras.models import load_model

from keras.preprocessing.image import ImageDataGenerator
import tensorflow as tf


def load_dataset():
    (trainX, trainY), (testX, testY) = cifar10.load_data()
    trainY = to_categorical(trainY)
    testY = to_categorical(testY)
    return trainX, trainY, testX, testY

def prep_pixels(train, test):
    train_norm = train.astype('float32')
    test_norm = test.astype('float32')
    train_norm = train_norm / 255.0
```

```
    test_norm = test_norm / 255.0
    return train_norm, test_norm


def summarize_diagnostics(history):
    plt.subplot(211)
    plt.title('Cross Entropy Loss')
    plt.plot(history.history['loss'], color='blue', label='train')
    plt.plot(history.history['val_loss'], color='green', label='test')
    plt.subplot(212)
    plt.title('Classification Accuracy')
    plt.plot(history.history['accuracy'], color='blue', label='train')
    plt.plot(history.history['val_accuracy'], color='green', label='test')

    results_dir = Path('dsc650/dsc650/assignments/assignment06').
↪joinpath('results')
    plt.show()




def define_model():
    model = Sequential()
    model.add(Conv2D(32, (3, 3), activation='relu',␣
↪kernel_initializer='he_uniform', padding='same', input_shape=(32, 32, 3)))
    model.add(Conv2D(32, (3, 3), activation='relu',␣
↪kernel_initializer='he_uniform', padding='same'))
    model.add(MaxPooling2D((2, 2)))
    model.add(Conv2D(64, (3, 3), activation='relu',␣
↪kernel_initializer='he_uniform', padding='same'))
    model.add(Conv2D(64, (3, 3), activation='relu',␣
↪kernel_initializer='he_uniform', padding='same'))
    model.add(MaxPooling2D((2, 2)))
    model.add(Conv2D(128, (3, 3), activation='relu',␣
↪kernel_initializer='he_uniform', padding='same'))
    model.add(Conv2D(128, (3, 3), activation='relu',␣
↪kernel_initializer='he_uniform', padding='same'))
    model.add(MaxPooling2D((2, 2)))
    model.add(Flatten())
    model.add(Dense(128, activation='relu', kernel_initializer='he_uniform'))
    model.add(Dropout(0.2))
    model.add(Dense(10, activation='softmax'))
    opt = SGD(lr=0.001, momentum=0.9)
    model.compile(optimizer=opt, loss='categorical_crossentropy',␣
↪metrics=['accuracy'])
    return model
```

```
def load_image(filename):
    img = load_img(filename, target_size=(32, 32))
    img = img_to_array(img)
    img = img.reshape(1, 32, 32, 3)
    img = img.astype('float32')
    img = img / 255.0
    return img

def run_test_harness():
    classes = ('airplane', 'auomobile', 'bird', 'cat', 'deer', 'dog', 'frog',␣
 ↪'horse', 'ship', 'truck')
    trainX, trainY, testX, testY = load_dataset()

    trainX, testX = prep_pixels(trainX, testX)

    for i in range(9):
        plt.subplot(330 + 1 + i)
        x = trainX[i]
        x = np.reshape(x, (32, 32, 3))
        plt.imshow(x)


    results_dir = Path('dsc650/dsc650/assignments/assignment06').
 ↪joinpath('results')
    plt.show()

    model = define_model()


    train_datagen = ImageDataGenerator(rescale=1. / 255, shear_range=0.2,␣
 ↪zoom_range=0.2, horizontal_flip=True)
    test_datagen = ImageDataGenerator(rescale=1. / 255)

    train_generator = train_datagen.flow(trainX, trainY, batch_size=64)
    validation_generator = test_datagen.flow(trainX, trainY, batch_size=64)


    nb_train_samples = 5
    nb_validation_samples = 5
    epochs = 2
    batch_size = 2
    history = model.fit_generator(
        train_generator,
        steps_per_epoch=nb_train_samples // batch_size,
        epochs=epochs,
        validation_data=validation_generator,
```

```
        validation_steps=nb_validation_samples // batch_size)


    _, acc = model.evaluate(testX, testY, verbose=0)

    results_dir = Path('dsc650/dsc650/assignments/assignment06').
 ↪joinpath('results')

    summarize_diagnostics(history)




def run_example_prediction():
    classes = ('airplane', 'auomobile', 'bird', 'cat', 'deer', 'dog', 'frog',␣
 ↪'horse', 'ship', 'truck')
    results_dir = Path('dsc650/dsc650/assignments/assignment06').
 ↪joinpath('results')



    data_dir = Path('dsc650/dsc650/assignments/assignment06').joinpath('Data')


# entry point, run the test harness#

run_example_prediction()
run_test_harness()
```

```
/opt/conda/lib/python3.8/site-
packages/tensorflow/python/keras/engine/training.py:1844: UserWarning:
`Model.fit_generator` is deprecated and will be removed in a future version.
Please use `Model.fit`, which supports generators.
  warnings.warn('`Model.fit_generator` is deprecated and '

Epoch 1/2
2/2 [==============================] - 1s 624ms/step - loss: 2.3029 - accuracy:
0.0417 - val_loss: 2.3022 - val_accuracy: 0.0781
Epoch 2/2
2/2 [==============================] - 0s 161ms/step - loss: 2.3019 - accuracy:
0.1510 - val_loss: 2.3017 - val_accuracy: 0.1172
```


Cross Entropy Loss
2.3025
2.3020
0.0
0.2
0.4
0.6
0.8
1.0
Classification Accuracy
0.15
0.10
0.05
0.0
0.2
0.4
0.6
0.8
1.0

[ ]: