SAINT MARY'S UNIVERSITY SINCE 1802

One University. One World. Yours.

Master of Science in Computing and Data Analytics

Deep Learning Assignment

Data & Text Mining

MCDA 5580

Submitted by:

Name	**A#**
Vamsi Manda	A00433234
Gyaneshwar Rao Nampally	A00433014
Allen Mathew	A00432526
Meghashyam	A00432392

Table of Contents

Objective and Requirement

The main objective of this assignment is to predict the images of the fruit dataset and test the accuracy. The dataset had two parts: one is training and the other is validation set. It consists of 60 different types of fruits which are grouped into respective folders.

Process of Model preparation

Partitioning Test and Train data

The model was trained considering 28,736 images of training dataset and validated against 9,673 images of testing dataset.

Method Used

The model has been created using CNN
2D convolution layers are used for images.

Architecture of the Layer

Sequential model is selected with the below-configured layers.
32 filters with (3,3) kernel size.
32 filters with (3,3) kernel size.
Pooling 2D with (2,2) kernel size.
64 filters with (3,3) kernel size.
64 filters with (3,3) kernel size.
Pooling 2D with (2,2) kernel size.
Flatten will convert the 2D matrix to a single vector.
Dense layer with 64 filters and RELU activation function.
Dense layer with 60 classes and softmax activation function. (the actual classification of 60 categories).

Rmspros is configured with a learning rate of 0.0001
Model is compiled with the setting for loss as catagorical_crossentropy.
Model is trained with 2 epochs.

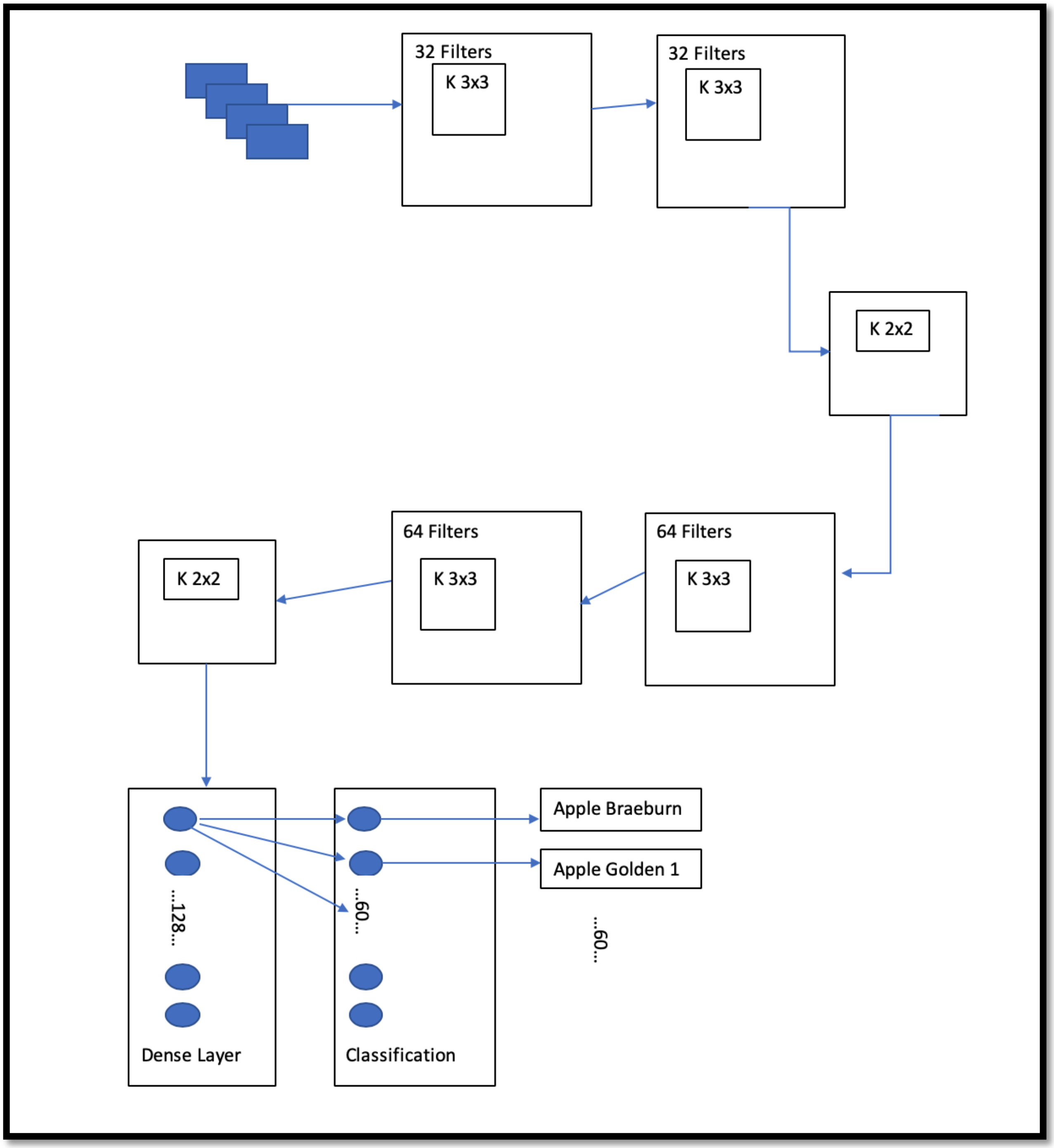

Figure 1 Architecture

Upon that two additional 2D convolution layers with 64 filters and kernel size 3 are added. Only 2 epochs were considered for the model as adding a greater number of epochs was consuming a lot of time as we didn't use the Tensor flow GPU. Finally, the accuracy of the model was observed to be 90%.

Code

```
# -*- coding: utf-8 -*-
import matplotlib.pyplot as plt
import numpy as np
from keras import Sequential
from keras import backend as K, utils
#from keras.callbacks import EarlyStopping
#from keras.datasets import cifar10
#from keras.layers import Dropout
from keras.layers import Dense, Conv2D, MaxPooling2D, Flatten
from keras.optimizers import rmsprop
import glob
import os
import cv2

def get_image(label, x, y):
    for i in range(0, len(y)):
        if np.argmax(y[i]) == label:
            return x[i]
    return None


def image_subset(index, x, y):
    xs = []
    ys = []
    for i in range(len(x)):
        if y[i] < index:
            xs.append(x[i])
            ys.append(y[i])
    return np.array(xs), np.array(ys)


def get_images(path):
    images = []
    labels = []
    names = []
    i = 0
    for fruit_dir_path in glob.glob(path):
        fruit_label = fruit_dir_path.split("/")[-1]
        for image_path in glob.glob(os.path.join(fruit_dir_path, "*.jpg")):
            image = cv2.imread(image_path, cv2.IMREAD_COLOR)

            image = cv2.resize(image, (45, 45))
            image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)

            images.append(image)
            names.append(fruit_label)
            labels.append(i)
        i += 1

    images = np.array(images)
    labels = np.array(labels)
    return images, labels, i

batch_size = 32
num_classes = 60
epochs = 2

x_train, y_train, _ = get_images("/Users/vamsimanda/Downloads/fruits-
360/Training/*")
x_test, y_test, _ = get_images("/Users/vamsimanda/Downloads/fruits-
360/Validation/*")

x_train, y_train = image_subset(num_classes, x_train, y_train)
x_test, y_test = image_subset(num_classes, x_test, y_test)

x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train /= 255
x_test /= 255
y_train = utils.to_categorical(y_train, num_classes)
y_test = utils.to_categorical(y_test, num_classes)

model = Sequential()

model.add(Conv2D(32, (3, 3), activation='relu',
                padding='same',
                input_shape=(45, 45, 3)))
model.add(Conv2D(32, (3, 3),
                activation='relu'))
model.add(MaxPooling2D((2, 2)))

model.add(Conv2D(64, (3, 3),
                padding='same',
                activation='relu'))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))

model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(num_classes, activation='softmax'))

opt = rmsprop(lr=0.0001, decay=1e-6)
model.compile(optimizer=opt,
              loss='categorical_crossentropy',
              metrics=['accuracy'])
model.fit(x_train, y_train,
          batch_size=batch_size,
          epochs=epochs,
          validation_data=(x_test, y_test),
          shuffle=True,
          verbose=2)

score = model.evaluate(x_test, y_test)
print('Test loss: {0}'.format(score[0]))
print('Test accuracy: {0}'.format(score[1]))

predictions = model.predict(x_test, verbose=0)
model.save("/Users/vamsimanda/Downloads/fruits-360/deepLearningModel.h5")
K.clear_session()
```

Output

```
Using TensorFlow backend.
WARNING: Logging before flag parsing goes to stderr.
W0808 13:09:29.472606 4800841152 deprecation_wrapper.py:119] From
/anaconda3/lib/python3.7/site-
packages/keras/backend/tensorflow_backend.py:74: The name
tf.get_default_graph is deprecated. Please use
tf.compat.v1.get_default_graph instead.

W0808 13:09:29.532650 4800841152 deprecation_wrapper.py:119] From
/anaconda3/lib/python3.7/site-
packages/keras/backend/tensorflow_backend.py:517: The name tf.placeholder
is deprecated. Please use tf.compat.v1.placeholder instead.

W0808 13:09:29.563200 4800841152 deprecation_wrapper.py:119] From
/anaconda3/lib/python3.7/site-
packages/keras/backend/tensorflow_backend.py:4138: The name
tf.random_uniform is deprecated. Please use tf.random.uniform instead.

W0808 13:09:29.623100 4800841152 deprecation_wrapper.py:119] From
/anaconda3/lib/python3.7/site-
packages/keras/backend/tensorflow_backend.py:3976: The name tf.nn.max_pool
is deprecated. Please use tf.nn.max_pool2d instead.

W0808 13:09:29.735142 4800841152 deprecation_wrapper.py:119] From
/anaconda3/lib/python3.7/site-packages/keras/optimizers.py:790: The name
tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer
instead.
```

```
W0808 13:09:29.748888 4800841152 deprecation_wrapper.py:119] From
/anaconda3/lib/python3.7/site-
packages/keras/backend/tensorflow_backend.py:3295: The name tf.log is
deprecated. Please use tf.math.log instead.

W0808 13:09:29.921550 4800841152 deprecation.py:323] From
/anaconda3/lib/python3.7/site-
packages/tensorflow/python/ops/math_grad.py:1250:
add_dispatch_support.<locals>.wrapper (from
tensorflow.python.ops.array_ops) is deprecated and will be removed in a
future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
W0808 13:09:30.044098 4800841152 deprecation_wrapper.py:119] From
/anaconda3/lib/python3.7/site-
packages/keras/backend/tensorflow_backend.py:986: The name tf.assign_add is
deprecated. Please use tf.compat.v1.assign_add instead.

2019-08-08 13:09:30.166999: I
tensorflow/core/platform/cpu_feature_guard.cc:142] Your CPU supports
instructions that this TensorFlow binary was not compiled to use: AVX2 FMA
Train on 28736 samples, validate on 9673 samples
Epoch 1/2
- 178s - loss: 1.5107 - acc: 0.6085 - val_loss: 0.7317 - val_acc: 0.7979
Epoch 2/2
- 189s - loss: 0.1903 - acc: 0.9430 - val_loss: 0.3472 - val_acc: 0.9042
9673/9673 [==============================] - 11s 1ms/step
Test loss: 0.34716642678206544
Test accuracy: 0.9041662359267248
```