**Sprawozdanie**

Sieci Neuronowe i Neurosterowniki

Projekt 2. Sieć Neuronowa

Radosław Bugiel, 235118

Armand Piecyk, 235293

Prowadzący: dr inż. Krzysztof Halawa

Termin: Środa 13:15 TP

1.Wstęp

Sieci neuronowe w założeniu mają przypominać strukturę ludzkiego mózgu. Ich elementarnymi składowymi są neurony. Jedną z ich cech jest to, że mogą być uczone przez co są niezwykle wartościowymi metodami analizy danych.

Celem tego projektu było stworzenie modelu takiej sieci, odpowiednie nauczenie go przez dostrojenie wag w neuronach.

Do zbudowania tej struktury użyty został język programowania Python wraz z biblioteką Keras.

2.Opis zadania sieci i danych

Zadaniem naszej sieci miała być identyfikacja obrazów i przypisanie ich do właściwej kategorii. Baza danych użyta do nauczenia i testowania sieci to Cifar-10. Zawiera ona 50.000 instancji treningowych i 10.000 instancji testowych, które są znormalizowane do rozmiarów 32x32 piksele.

Możemy wyróżnić kilka najważniejszych etapów:

**Pobieranie, obróbka i przygotowanie danych** - na wejście sieci poda-jemy tablice trójwymiarową, dane muszą być typu float32 i znormalizowane do zakresu [0;1]. Również wartości etykiet obrazków są zamienione na binarny wektor. Dane są dzielone na zbiór uczący i ten do testów.

**Stworzenie modelu sieci** - budowa modelu, składającego się z kilku warstw opisanych w dalszej części sprawozdania.

**Nauka sieci** - sieć jest trenowana za pomocą przygotowanych wcześniej danych.

**Wyniki** –na wejście sieci podawane są dane testowe, po czym wyświetlane są statystki dotyczące funkcji strat i precyzji w przewidywaniu modelu.

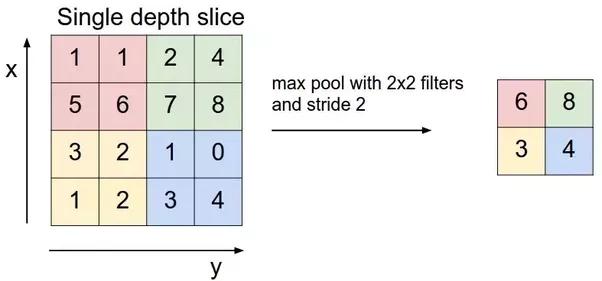
3.Poszczególne elementy sieci

3.1 Warstwa ELU

Użyta funkcja aktywacji z ang. Exponential Linear Unit. W przeciwieństwie do zwykłej funkcji aktywacji, ELU ma dodatkową zmienną alpha, która powinna być liczbą dodatnią. ELU jest bardzo podobne do RELU,różnica jest tylko gdy wejście jest ujemne(ELU potrafi zwrócić ujemne wyjście).

3.2 MAXPOLLING2

Ta operacja jest stosowana w celu wyodrębnienia z danego obszaru cechy, która dla nas ma największe znaczenie. Redukuje wymiarowość problemu oraz może być pomocna przy przetwarzaniu zrotowanych cyfr. Na wybranym przez nas obszarze obrazka o wymiarze 2x2 piksele wybierana jest wartość największa.



3.3 Flatten

Ta warstwa odpowiada za spłaszczenie swojego wejścia. Oznacza to, że redukujemy wymiar wejścia, które otrzymujemy z 1x4x4x128 do 1x2048. Celem tej zmiany jest podanie przygotowanych wcześniej danych do klasycznej sieci neuronowej.

1. Badanie sieci neuronowej

Mając utworzony wstępny program, można zbadać za jego pomocą wpływ poszczególnych parametrów modelu na jego działanie. Ze względu na dużą ilość epok i w związku z tym długi czas wykonywania programu (80 min) przeprowadziliśmy mniejszą ilość testów.

**Wyniki testów:**

**Zmiana liczby epok:**

Przy docelowej liczbie epok(125) wyniki wachają się w granicach 85-92% poprawnej klasyfikacji. Zmiana ilości epok na 80 powoduje mały spadek precyzji w granicach 1-3%. Zaś zwiększenie liczby epok do 170 nie powoduje żadnych widocznych zmian klasyfikacji. Wzrasta jedynie czas wykonywania się programu do 105 minut.

Następnie zmieniliśmy **ilość bloków VGG** z 3 na 2 co daje wyraźny spadek precyzji o ponad 3%.

**Zmiana learning rate:**

Przy pierwszych 75 epokach learning rate(lrate) wynosi 0.001, dla następnych 25 epok mamy lrate równe 0.0005, a ostatnie 25 epok wykonuje się z lrate na poziome 0.0003. Jeden test polegał na ujednoliceniu lrate i po testach, gdzie 125 epok wykonało się z lrate równym 0.001 nie było zmian, ani w precyzji, ani w czasie wykonywania.

**Zmiana batchsize:**

Zauważyliśmy, że najlepsze wyniki wychodzą dla batchsize w granicach 32, zmiana w którąkolwiek ze stron powoduje zmniejszenie precyzji.

**Wyniki:**

Dokładność: 87.24%

Strata: 0.55

Macierz:

[[925 10 6 3 1 0 4 3 32 16]

[ 4 948 0 0 0 0 2 0 5 41]

[ 61 1 768 16 28 21 80 12 6 7]

[ 37 8 19 679 40 62 93 23 19 19]

[ 15 1 14 15 856 6 64 23 5 1]

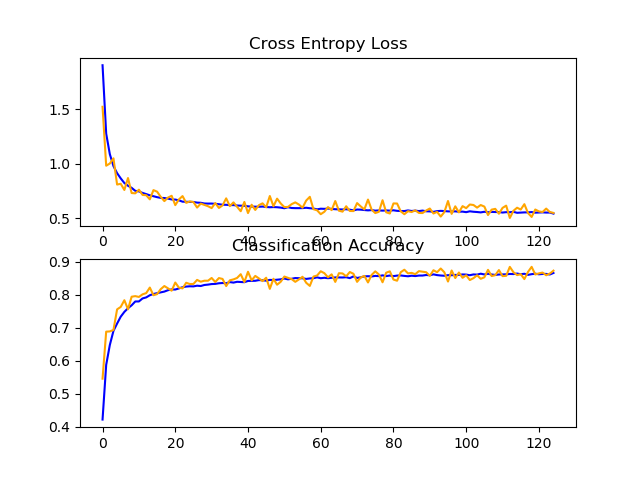
[ 16 2 17 75 24 779 45 25 1 16]

[ 9 4 2 7 5 0 968 1 3 1]

[ 14 1 3 8 21 15 10 924 1 3]

[ 40 9 1 0 1 0 2 0 929 18]

[ 19 23 0 0 0 0 1 2 8 947]]



1. Wnioski

Z testów wynika, że największy wpływ na precyzje ma liczba epok, ale tylko do pewnego momentu. Zwiększanie tej liczby powyżej pewnego poziomu nie ma sensu, gdyż powoduje jedynie zwiększenie czasu pracy programu.

Zmiana ilości bloków VGG także wyraźnie wpłynęła na poprawność klasyfikacji zdjęć.

Kod źródłowy:

import keras

import sys

from matplotlib import pyplot

from keras.utils import to\_categorical

from keras.models import Sequential

from keras.utils import np\_utils

from keras.preprocessing.image import ImageDataGenerator

from keras.layers import Dense, Activation, Flatten, Dropout, BatchNormalization

from keras.layers import Conv2D, MaxPooling2D

from keras.datasets import cifar10

from keras import regularizers

from keras.callbacks import LearningRateScheduler

import numpy as np

import timeit

def summarize\_diagnostics(history, name):

pyplot.subplot(211)

pyplot.title('Cross Entropy Loss')

pyplot.plot(history.history['loss'], color='blue', label='train')

pyplot.plot(history.history['val\_loss'], color='orange', label='test')

pyplot.subplot(212)

pyplot.title('Classification Accuracy')

pyplot.plot(history.history['acc'], color='blue', label='train')

pyplot.plot(history.history['val\_acc'], color='orange', label='test')

pyplot.savefig(name + '\_plot.png')

pyplot.close()

(x\_train, y\_train), (x\_test, y\_test) = cifar10.load\_data()

x\_train = x\_train.astype('float32')

x\_test = x\_test.astype('float32')

x\_train = x\_train/ 255.0

x\_test = x\_test/ 255.0

num\_classes = 10

y\_train = to\_categorical(y\_train,num\_classes)

y\_test = to\_categorical(y\_test,num\_classes)

weight\_decay = 1e-4

model = Sequential()

model.add(Conv2D(32, (3,3), padding='same', kernel\_regularizer=regularizers.l2(weight\_decay), input\_shape=x\_train.shape[1:]))

model.add(Activation('elu'))

model.add(BatchNormalization())

model.add(Conv2D(32, (3,3), padding='same', kernel\_regularizer=regularizers.l2(weight\_decay)))

model.add(Activation('elu'))

model.add(BatchNormalization())

model.add(MaxPooling2D(pool\_size=(2,2)))

model.add(Dropout(0.2))

model.add(Conv2D(64, (3,3), padding='same', kernel\_regularizer=regularizers.l2(weight\_decay)))

model.add(Activation('elu'))

model.add(BatchNormalization())

model.add(Conv2D(64, (3,3), padding='same', kernel\_regularizer=regularizers.l2(weight\_decay)))

model.add(Activation('elu'))

model.add(BatchNormalization())

model.add(MaxPooling2D(pool\_size=(2,2)))

model.add(Dropout(0.3))

model.add(Conv2D(128, (3,3), padding='same', kernel\_regularizer=regularizers.l2(weight\_decay)))

model.add(Activation('elu'))

model.add(BatchNormalization())

model.add(Conv2D(128, (3,3), padding='same', kernel\_regularizer=regularizers.l2(weight\_decay)))

model.add(Activation('elu'))

model.add(BatchNormalization())

model.add(MaxPooling2D(pool\_size=(2,2)))

model.add(Dropout(0.4))

model.add(Flatten())

model.add(Dense(num\_classes, activation='softmax'))

datagen = ImageDataGenerator(rotation\_range=15,width\_shift\_range=0.1,height\_shift\_range=0.1,horizontal\_flip=True,)

datagen.fit(x\_train)

start = timeit.default\_timer()

batch\_size = 32

opt\_rms = keras.optimizers.rmsprop(lr=0.001,decay=1e-6)

model.compile(loss='categorical\_crossentropy', optimizer=opt\_rms, metrics=['accuracy'])

history = model.fit\_generator(datagen.flow(x\_train, y\_train, batch\_size=batch\_size),\

steps\_per\_epoch=x\_train.shape[0] // batch\_size,epochs=125,\

verbose=1,validation\_data=(x\_test,y\_test))

scores = model.evaluate(x\_test, y\_test, batch\_size=128, verbose=1)

print('\nTest result: %.3f loss: %.3f' % (scores[1]\*100,scores[0]))

summarize\_diagnostics(history,'1')

from sklearn.metrics import confusion\_matrix

predict\_classes = model.predict\_classes(x\_test[1:10000,], batch\_size=32)

true\_classes = np.argmax(y\_test[1:10000],1)

print(confusion\_matrix(true\_classes, predict\_classes))

stop = timeit.default\_timer()

print('Time: ', stop - start)

input("Press enter to exit")