SAKARYA ÜNİVERSİTESİ BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

DERİN ÖĞRENME VE EVRİŞİMLİ SİNİR AĞLARI

ÖDEV - 2

(CNN MODEL TRAINING)

NOT

Bu dosyadaki kişisel bilgiler github yüklemesi için silinmiştir.

Enes Çavuş

Mayıs, 2021

İçindekiler

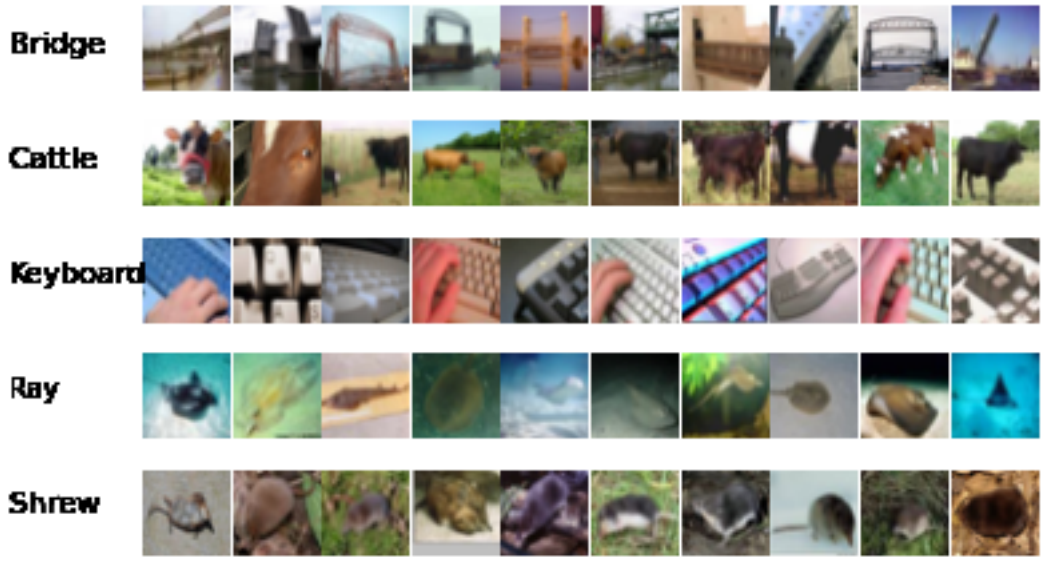
Giriş

Proje boyunca birçok kez farklı hiperparametreler deneyerek modeller eğittim. Genellikle overfitting durumu ile karşılaştım. Modeller 150 epoch ve üzerindeki eğitimlerde aşırı uydurmaya başlıyordu. O yüzden epoch sayımı bu değerin altında tuttum.

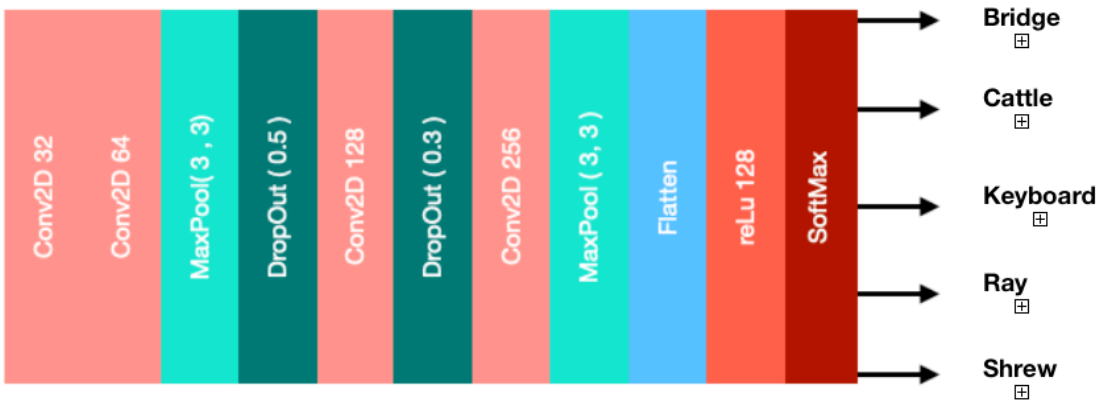
Learning Rate'i azaltarak, ağı sadeleştirerek ve DropOut katmanlarını ekleyerek modelin daha başarılı olması için denemeler yaptım. Diğer yorumlamalar ve analizler, sonraki başlıklarda incelenecektir.

NOTE: Eğitimleri Google Colab GPU servisini kullanarak gerçekleştirdim. Tüm kodlarımı ve grafikleri parçalar halinde derledim/çalıştırdım, bu yüzden rapor sonuna eklenen kod bir bütün olarak veya tek seferde çalıştırılaması durumunda hata verecektir.

1 - Eğitimde Kullanılan Sınıflara Ait Örnek Görüntüler



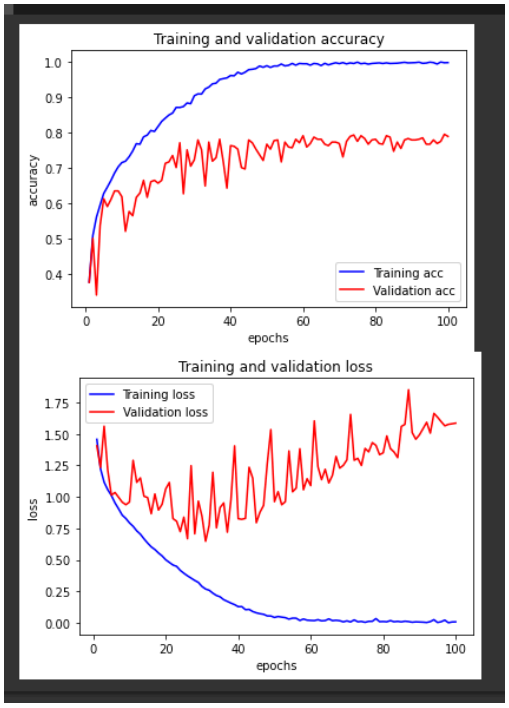
2 - Model Tasarımı - Blok Şeması

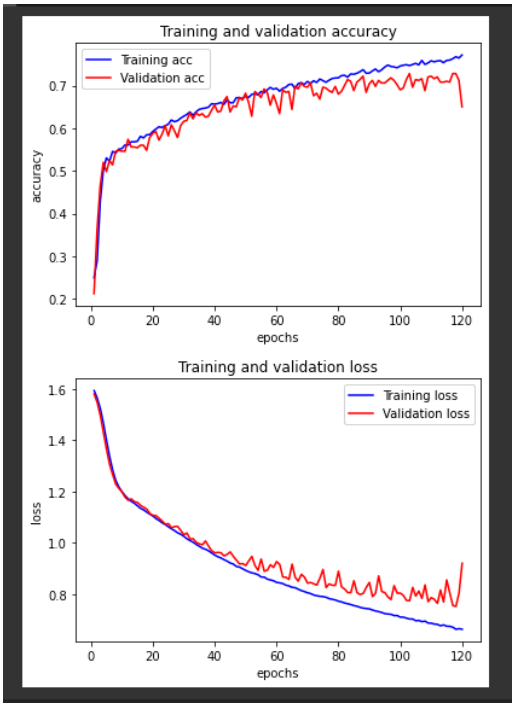


Modelde 4 adet ConvNet, 2 Adet Dropout, 2 Adet MaxPooling, 1 Flatten, 1 ReLu Dense ve son olarak 5 çıkışlı Softmax katmanlarını kullandım. Havuzlama katmanlarında (1, 1) , (2 , 2), (3, 3) gibi değerler denedim fakat görsellerin boyutları zaten çok küçük olduklarından birbirine yakın sonuçlar elde ettiğim modeller eğitebildim.

Dropout kullanımı overfitting önlenmesinde oldukça etkili oldu. Dropout öncesi eğitimlerimde çok düşük epoch değerlerinde bile aşırı uydurma başlarken Dropout işlemleri ile bu durum daha iyi bir hale geldi. Modelleri daha uzun süre eğitebilir hale getirdim.

3 - Model Eğtiimi - Accuracy - Loss Grafikleri Değerlendirme





Soldaki görsel, hiperparametreler üzerinde değişiklik yapmadan önce gerçekleştirdiğim ilk eğitimlerimden birine ait. Eğitim veri setinin kayıp fonksiyonunun değerinin en düşük seviyeler inmesi ve doğruluk değerinin de en yüksek başarım oranlarına sahip olmasına rağmen validation setindeki değerlerin başarısız olmaya başladığını ve olması gerekenin tam zıttı yönde ilerlediğini görmekteyiz. Buradan anlaşıldığı üzere, çok düşük epoch adımlarında bile model overfitting durumuna geçmektedir.

Sağdaki görselde ise hiperparametreler üzerinde geliştirmeler yapıldıktan sonraki son eğitim verilerine ait grafikler bulunmaktadır. Görüldüğü üzere overfitting için iyileştirmeler yapılmış ve daha başarılı bir eğitim gerçekleştirilmiştir.

5 - Eğitim ve Analiz Kodları

```python
# author: Enes Çavuş - b161210001
# subject: Derin Öğrenme Dersi 2. Ödev
# date: May 5 2021
# NOTE: Kodlar Jupyter Notebook üzerinden parça parça derlendi çalıştırıldı.
# Kodlarin tumu tek seferde çalıştırıldıginda hata verebilir.!

from keras.datasets import cifar100
from keras.models import load_model
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
from sklearn.metrics import classification_report, confusion_matrix
import seaborn as sn
import pandas  as pd
import keras
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D,Activation, Dense,Flatten
from keras import layers
from keras.utils import to_categorical
from keras import optimizers
import os
import random

# veri seti yukle
(x_train, y_train), (x_test, y_test) = cifar100.load_data()

print(x_train.shape) # egitim fotoları
print(y_train.shape) # egitim labellari
print(x_test.shape) # test fotolari
print(y_test.shape) # test labellari
# 12=bridge 19=cattle 39=keyboard 67=ray 74=shrew

# örnek gorsel incelemesi
for img in range(1):
    plt.imshow(x_train[img,:,:,:], interpolation='nearest')
    print(y_train[img])
    plt.show()

# odevde bana ait olan siniflari kullanmak uzere yeni dizilere aktariyorum
new_x_train = []
new_y_train = []
new_x_test = []
new_y_test = []
```

```python
for i in range(x_train.shape[0]):
    if (int(y_train[i]) == 12) or (int(y_train[i]) == 19) or (int(y_train[i]) == 39) or (int(y_train[i]) == 67) or (int(y_train[i]) == 74):
        new_x_train.append(x_train[i])
        new_y_train.append(y_train[i])

for i in range(x_test.shape[0]):
    if (int(y_test[i]) == 12) or (int(y_test[i]) == 19) or (int(y_test[i]) == 39) or (int(y_test[i]) == 67) or (int(y_test[i]) == 74):
        new_x_test.append(x_test[i])
        new_y_test.append(y_test[i])

new_x_train = np.array(new_x_train)
print(new_x_train.shape)
new_y_train = np.array(new_y_train)
print(new_y_train.shape)
new_x_test = np.array(new_x_test)
print(new_x_test.shape)
new_y_test = np.array(new_y_test)
print(new_y_test.shape)

# Eğitim ve test veri setleri - yeniden esleme - eski adlandirma ile devam edilecek
(x_train, y_train), (x_test, y_test) = (new_x_train, new_y_train),(new_x_test, new_y_test)

for i in range(0,16):
    plt.subplot(4,4,i+1)
    plt.imshow(x_train[i])

print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)

# normalizasyon adimlarinda sinif numaralarini sirasiyla 0-1-2-3-4 seklinde belirlenecek
for i in range(y_train.shape[0]):
    if y_train[i] == 12:
        y_train[i] = 0
    elif y_train[i] == 19:
        y_train[i] = 1
    elif y_train[i] == 39:
        y_train[i] = 2
    elif y_train[i] == 67:
        y_train[i] = 3
    elif y_train[i] == 74:
        y_train[i] = 4
```

```python
for i in range(y_test.shape[0]):
    if y_test[i] == 12:
        y_test[i] = 0
    elif y_test[i] == 19:
        y_test[i] = 1
    elif y_test[i] == 39:
        y_test[i] = 2
    elif y_test[i] == 67:
        y_test[i] = 3
    elif y_test[i] == 74:
        y_test[i] = 4


print(np.unique(y_train))
print(np.unique(y_test))

# gorselleri ve siniflari modelin anlayacagi sekle getirmek amacli encoding adimlari
x_train=x_train.astype('float32')/255.0
x_test=x_test.astype('float32')/255

y_train=to_categorical(y_train,5)
y_test=to_categorical(y_test,5)

######################################## MODEL TANIMLAMA ########################################
model=Sequential()
model.add(layers.Conv2D(32,
                        (3,3),
                        activation='relu',
                        padding='same',
                        input_shape= (32, 32, 3)))
model.add(layers.Conv2D(64,
                        (3,3),
                        padding='same',
                        activation='relu'))
model.add(layers.MaxPool2D())
model.add(layers.Conv2D(128,
                        (3,3),
                        padding='same',
                        activation='relu'))
model.add(layers.MaxPool2D())
model.add(layers.Conv2D(256,
                        (3,3),
                        padding='same',
                        activation='relu'))
model.add(layers.MaxPool2D())
```

```
model.add(layers.MaxPool2D())
model.add(layers.Flatten())
model.add(layers.Dense(128,activation='relu'))
model.add(layers.Dense(5,activation='softmax'))
model.summary()
#modeli derle
from keras import optimizers
model.compile(loss='categorical_crossentropy',
              optimizer=optimizers.RMSprop(lr=.00001),
              metrics=['acc'])

##################################### MODEL EGITIMI ###################################

history=model.fit(x_train,
                  y_train,
                  epochs=120,
                  validation_data=(x_test,y_test))

##################################### MODEL Dogruluk ve Kayip degerlerinin gorsellestirilmesi - plot ###################

accuracy = history.history['acc']
valid_acc = history.history['val_acc']
loss = history.history['loss']
valid_loss = history.history['val_loss']
epochs = range(1,121)
# ACCURACY
plt.title('Accuracy Values')
plt.plot(epochs, accuracy, label='Training ')
plt.plot(epochs, valid_acc, label='Validation')
plt.xlabel('epochs')
plt.ylabel('accuracy')
plt.legend()
plt.figure()
# LOSS
plt.title('Loss Values')
plt.plot(epochs, loss, label='Training')
plt.plot(epochs, valid_loss, label='Validation')
plt.xlabel('epochs')
plt.ylabel('loss')
plt.legend()
plt.show()

##################################### MODEL  ###################################
classNames = ["bridge", "cattle", "keyboard", "ray","shrew"]
plt.figure(figsize=(10,10))
```

```
##################################### MODEL  ###################################
classNames = ["bridge", "cattle", "keyboard", "ray","shrew"]
plt.figure(figsize=(10,10))
for i in range(5):
    index = sirali_list[i]
    image = x_test[index]
    img = image.astype('float32')
    img /= 255
    im = np.zeros(32*32*3).reshape((1,32,32,3))
    im[0] = img
    data = im
    ret = model.predict(data, batch_size=1)
    plt.subplot(1, 5, i+1)
    plt.imshow(image)

    plt.axis('off')

    plt.title(classNames[np.argmax(ret)])
plt.show()

########### CONFUSION MATRIX CODES #############
predicted = model.predict(x_test)
pred = np.argmax(predicted, axis=1)
matrix = confusion_matrix(np.argmax(y_test,axis=1),pred)

asDataFrame = pd.DataFrame(matrix, range(5),range(5))
sn.heatmap(asDataFrame, annot=True,annot_kws={"size": 10}, fmt=".1f")
plt.figure(figsize = (12,10))
plt.show()
```