T.C

SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BSM 310 – YAPAY ZEKA

-CIFAR-100-

Grup üyeleri:
G191210303 Barış YILMAZ
G191210387 Cemal ASLAN

Sakarya
2020

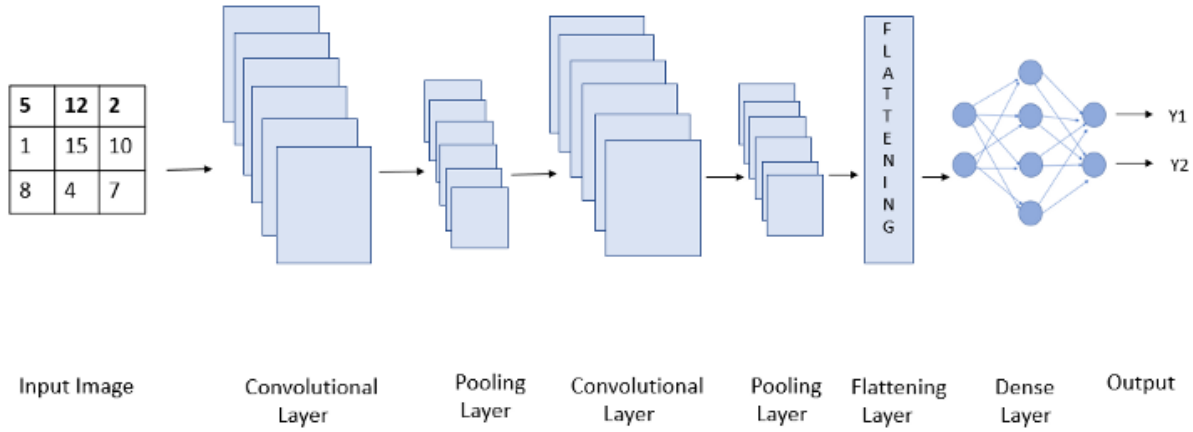
ÖZET

Bu çalışmada yapay zeka alanında görüntü tanıma veri kümesi olan CIFAR-100'den bahsedilmiştir. Geçmişten günümüze yapılan yapay zeka çalışmalarında kendi içerisinde farklı dallara ayrılan yapay zeka çalışmaları alanlarından son zamanlarda en çok ilgi duyulan ve çalışma yapılan alan ise bilgisayarlı görüntü işleme çalışmalarıdır. Bilgisayarların insanlar gibi görüntüleri algılayabilmesi, sınıflandırabilmesi ve yorulayabilmesi amacıyla geliştirilen özel derin öğrenme mimarileri bulunur. Bunlardan en çok kullanılan ve bu çalışmada da bahsedilecek olan mimari evrişimsel sinir ağlarıdır.

GİRİŞ

CIFAR-100 (Kanada Gelişmiş Araştırma Enstitüsü) veri kümesi, genel olarak makine öğrenmesi algoritmalarını ve bilgisayar vizyonunu öğretmek için kullanılan bir görüntü koleksiyonudur. Bu, makine öğrenimi araştırması için en yaygın kullanılan veri setlerinden biridir. 100 farklı sınıfta 60.000 (50000 eğitim, 10000 test) 32×32 renkli görüntü içerir. 100 farklı sınıfta uçaklar, arabalar, kuşlar, kediler, geyikler, köpekler, kurbağalar, atlar, gemiler, kamyonlar vb. birçok nesne bulunur. Fotoğraflardaki nesneleri tanımak için kullanılan bilgisayar algoritmaları genellikle örnek olarak öğrenir. CIFAR-100, bilgisayarınıza nesneleri nasıl tanıyacağınızı öğretmek için kullanabileceğiniz bir resim koleksiyonudur. CIFAR-100'deki görüntüler düşük bir çözünürlüğe (32×32) sahip olduğundan, bu veri seti araştırmacıların neyin işe yaradığını görmek için farklı algoritmaları hızla denemelerine izin verebilir.

METOT



Cifar-100 verisetinde görüntü tanıma için farklı metotlar bulunmaktadır ve farklı yollarla farklı doğruluk payı sonuçları alınabilmektedir. Bize verilen ödevin içeriğindeki metotlara eklemeler yaparak, bazı metotlarda düzenlemeler yaparak içeriğin sonucundaki doğruluk payını değişikliklerimizle ve data augmentation ile geliştirmeye çalıştık.

Orijinal içerikteki hyper parametrelere “kernel_initializer='he_uniform'” eklemesi yaptık ve “elu” olan activation’ı “relu” olarak güncelledik. Ek olarak her katmana Batch Normalization ekledik ve 0.25 sabit olan Dropout’u ise 0.2 ile başlayıp her katmanda 0.1 artacak şekilde güncelledik.

RMSProp, lr=0.0001 ve decay=1e-6 parametreli optimizer’ı ise Adam(lr=.001, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.0, amsgrad=False) olarak güncelledik. Loss ve metrics’i yapıya en uygun oldukları için değiştirmedik.

Data augmentation olarak ise rastgele yatay çevirme ve görüntünün genişlik, yüksekliğindeki %10’luk kaymaları şeklinde tasarladık.
ImageDataGenerator(width_shift_range=0.1, height_shift_range=0.1, horizontal_flip=True)

DENEYSEL SONUÇLAR

Öncelikle benzer model olan ResNet ile transfer learning uygulanmamış şekilde modelin orijinal içeriğin 10 epoch eğitimi;

```
Epoch 1/10
782/782 [==============================] - 90s 75ms/step - loss: 4.2060 - accuracy: 0.0658 - val_loss: 3.5740 - val_accuracy: 0.1708
Epoch 2/10
782/782 [==============================] - 58s 74ms/step - loss: 3.4440 - accuracy: 0.1890 - val_loss: 3.0487 - val_accuracy: 0.2641
Epoch 3/10
782/782 [==============================] - 58s 75ms/step - loss: 3.0585 - accuracy: 0.2596 - val_loss: 2.8131 - val_accuracy: 0.3036
Epoch 4/10
782/782 [==============================] - 58s 75ms/step - loss: 2.7486 - accuracy: 0.3205 - val_loss: 2.4826 - val_accuracy: 0.3741
Epoch 5/10
782/782 [==============================] - 58s 75ms/step - loss: 2.5626 - accuracy: 0.3581 - val_loss: 2.3368 - val_accuracy: 0.4074
Epoch 6/10
782/782 [==============================] - 58s 75ms/step - loss: 2.4096 - accuracy: 0.3918 - val_loss: 2.2819 - val_accuracy: 0.4187
Epoch 7/10
782/782 [==============================] - 58s 74ms/step - loss: 2.2669 - accuracy: 0.4220 - val_loss: 2.1676 - val_accuracy: 0.4507
Epoch 8/10
782/782 [==============================] - 58s 74ms/step - loss: 2.1321 - accuracy: 0.4490 - val_loss: 2.1230 - val_accuracy: 0.4584
Epoch 9/10
782/782 [==============================] - 58s 75ms/step - loss: 2.0067 - accuracy: 0.4791 - val_loss: 2.0564 - val_accuracy: 0.4731
Epoch 10/10
782/782 [==============================] - 58s 75ms/step - loss: 1.8884 - accuracy: 0.5040 - val_loss: 2.0109 - val_accuracy: 0.4833
```

Aslında val_accuracy'den görüleceği üzere 9 epoch yeterliymiş.
Yaptığımız değişikliklerle bizim 10 epoch eğitimimiz;

```
Epoch 1/10
781/781 [==============================] - 67s 42ms/step - loss: 4.5927 - accuracy: 0.0858
Epoch 2/10
781/781 [==============================] - 33s 42ms/step - loss: 3.3490 - accuracy: 0.2257
Epoch 3/10
781/781 [==============================] - 33s 42ms/step - loss: 2.6858 - accuracy: 0.3327
Epoch 4/10
781/781 [==============================] - 32s 41ms/step - loss: 2.3066 - accuracy: 0.3954
Epoch 5/10
781/781 [==============================] - 32s 41ms/step - loss: 2.0264 - accuracy: 0.4569
Epoch 6/10
781/781 [==============================] - 32s 41ms/step - loss: 1.8449 - accuracy: 0.4992
Epoch 7/10
781/781 [==============================] - 32s 41ms/step - loss: 1.7070 - accuracy: 0.5251
Epoch 8/10
781/781 [==============================] - 32s 41ms/step - loss: 1.5847 - accuracy: 0.5553
Epoch 9/10
781/781 [==============================] - 32s 41ms/step - loss: 1.4795 - accuracy: 0.5859
Epoch 10/10
781/781 [==============================] - 33s 42ms/step - loss: 1.3904 - accuracy: 0.6021
<tensorflow.python.keras.callbacks.History at 0x7f4601893b50>


test_loss, test_acc = model.evaluate(x_test,  y_test)
print(test_acc)

313/313 [==============================] - 3s 8ms/step - loss: 1.4591 - accuracy: 0.5927
0.5927000045776367
```

Benzer model olan ResNet ile transfer learning uygulanmış 40 epoch eğitim;

```
Epoch 1/40
781/781 [==============================] - 30s 34ms/step - loss: 5.0259 - accuracy: 0.0252
Epoch 2/40
781/781 [==============================] - 26s 33ms/step - loss: 4.4013 - accuracy: 0.0522
Epoch 3/40
781/781 [==============================] - 26s 33ms/step - loss: 4.3133 - accuracy: 0.0647
Epoch 4/40
781/781 [==============================] - 26s 34ms/step - loss: 4.2605 - accuracy: 0.0726
Epoch 5/40
781/781 [==============================] - 26s 33ms/step - loss: 4.2263 - accuracy: 0.0741
Epoch 6/40
781/781 [==============================] - 26s 33ms/step - loss: 4.1755 - accuracy: 0.0819
Epoch 7/40
781/781 [==============================] - 26s 33ms/step - loss: 4.1826 - accuracy: 0.0801
Epoch 8/40
781/781 [==============================] - 26s 34ms/step - loss: 4.1488 - accuracy: 0.0849
Epoch 9/40
781/781 [==============================] - 26s 33ms/step - loss: 4.1192 - accuracy: 0.0886
Epoch 10/40
781/781 [==============================] - 26s 34ms/step - loss: 4.1127 - accuracy: 0.0912
Epoch 11/40
781/781 [==============================] - 26s 34ms/step - loss: 4.0960 - accuracy: 0.0925
Epoch 12/40
781/781 [==============================] - 26s 33ms/step - loss: 4.0804 - accuracy: 0.0953
Epoch 13/40
781/781 [==============================] - 26s 33ms/step - loss: 4.0775 - accuracy: 0.0948
Epoch 14/40
781/781 [==============================] - 26s 33ms/step - loss: 4.0604 - accuracy: 0.0983
Epoch 15/40
781/781 [==============================] - 26s 34ms/step - loss: 4.0491 - accuracy: 0.0997
Epoch 16/40
781/781 [==============================] - 27s 34ms/step - loss: 4.0325 - accuracy: 0.1043
Epoch 17/40
781/781 [==============================] - 27s 34ms/step - loss: 4.0213 - accuracy: 0.1063
Epoch 18/40
781/781 [==============================] - 26s 33ms/step - loss: 4.0121 - accuracy: 0.1060
Epoch 19/40
781/781 [==============================] - 26s 33ms/step - loss: 4.0120 - accuracy: 0.1052
Epoch 20/40
781/781 [==============================] - 27s 34ms/step - loss: 4.0071 - accuracy: 0.1063
```

```
Epoch 21/40
781/781 [==============================] - 26s 34ms/step - loss: 3.9890 - accuracy: 0.1103
Epoch 22/40
781/781 [==============================] - 26s 34ms/step - loss: 3.9848 - accuracy: 0.1108
Epoch 23/40
781/781 [==============================] - 26s 34ms/step - loss: 3.9702 - accuracy: 0.1143
Epoch 24/40
781/781 [==============================] - 28s 35ms/step - loss: 3.9665 - accuracy: 0.1113
Epoch 25/40
781/781 [==============================] - 28s 36ms/step - loss: 3.9704 - accuracy: 0.1095
Epoch 26/40
781/781 [==============================] - 28s 36ms/step - loss: 3.9603 - accuracy: 0.1141
Epoch 27/40
781/781 [==============================] - 28s 36ms/step - loss: 3.9474 - accuracy: 0.1166
Epoch 28/40
781/781 [==============================] - 28s 36ms/step - loss: 3.9347 - accuracy: 0.1195
Epoch 29/40
781/781 [==============================] - 28s 35ms/step - loss: 3.9341 - accuracy: 0.1187
Epoch 30/40
781/781 [==============================] - 28s 36ms/step - loss: 3.9344 - accuracy: 0.1163
Epoch 31/40
781/781 [==============================] - 28s 36ms/step - loss: 3.9262 - accuracy: 0.1211
Epoch 32/40
781/781 [==============================] - 28s 36ms/step - loss: 3.9172 - accuracy: 0.1255
Epoch 33/40
781/781 [==============================] - 28s 35ms/step - loss: 3.9169 - accuracy: 0.1211
Epoch 34/40
781/781 [==============================] - 27s 35ms/step - loss: 3.9169 - accuracy: 0.1231
Epoch 35/40
781/781 [==============================] - 27s 34ms/step - loss: 3.8989 - accuracy: 0.1231
Epoch 36/40
781/781 [==============================] - 26s 34ms/step - loss: 3.8993 - accuracy: 0.1272
Epoch 37/40
781/781 [==============================] - 27s 34ms/step - loss: 3.8921 - accuracy: 0.1255
Epoch 38/40
781/781 [==============================] - 27s 34ms/step - loss: 3.8925 - accuracy: 0.1243
Epoch 39/40
781/781 [==============================] - 27s 34ms/step - loss: 3.8874 - accuracy: 0.1263
Epoch 40/40
781/781 [==============================] - 27s 34ms/step - loss: 3.8899 - accuracy: 0.1262
<tensorflow.python.keras.callbacks.History at 0x7f45aee05490>

restest_loss, restest_acc = tl_model.evaluate(x_test,  y_test)
print(restest_acc)

313/313 [==============================] - 5s 13ms/step - loss: 3.8827 - accuracy: 0.1381
0.13809999823570251
```

Transfer learning ile deneyebildiğimiz 40 epoch eğitimde en fazla %12.5'lara kadar çıktı.

DEĞERLENDİRME

Çok fazla değişiklik ve eklemeler deneyerek uzun uğraşlar sonucu doğruluk payını iyileştirebilecek şekilde olacak değişik ve eklemelerle büyüttüğümüz veri setinde orijinal içeriğin eğitiminde çıkan %50 civarındaki doğruluk payını %10 iyileştirerek kendi içeriğimizde %60'lara çıkarabilmiş olduk.

KAYNAKÇA

https://paperswithcode.com/
https://www.kaggle.com/
https://machinelearningmastery.com/