KÜTAHYA DUMLUPINAR ÜNİVERSİTESİ

Yüksek Düzey Programlama Ödev Raporu

Digit Recognizer Veri Seti İle Model Geliştirme

Doç. Dr. HASAN TEMURTAŞ

Kubilay Dicle 202013172019

MNIST El Yazısı Rakam Tanıma Projesi

Projenin Amacı

Bu projenin amacı, MNIST el yazısı rakamlar veri seti kullanılarak bir yapay sinir ağı (ANN) modeli tasarlayıp eğitmek ve rakamları sınıflandırmaktır. Bu model, 0-9 arasındaki rakamları analiz ederek doğru sınıflandırmayı öğrenir. Projede, TensorFlow ve Keras kütüphaneleri kullanılmıştır.

Modelin temel hedefleri:

- MNIST veri setini işlemek ve eğitmek.
- Model doğruluğunu test etmek.
- Eğitilen modeli bir dosyaya kaydederek gelecekte kullanmak.

MNIST Veriseti ve İşleme

MNIST, el yazısı rakamları içeren standart bir veri setidir. Her bir görüntü:

- 28x28 boyutunda gri tonlamalıdır.
- Değeri 0-255 arasında değişen pikseller içerir.

Veri ön işleme adımları:

1. **Düzleştirme**: 28x28 matris boyutundaki görüntüler, 784 birimlik tek boyutlu bir vektöre dönüştürülmüştür.
2. **Normalize etme:** Piksel değerleri [0, 1] aralığına ölçeklendirilmiştir. Bu işlem, modelin daha hızlı ve stabil öğrenmesini sağlar.

Sinir Ağı Modelinin Tasarımı

Model, Keras'ın Sequential yapısını kullanarak oluşturulmuştur. Bu yapı, katmanların sıralı bir şekilde eklenmesine olanak tanır.

Modelde kullanılan katmanlar ve işlevleri:

1. **Giriş Katmanı:**
    - Görsellerden gelen 784 giriş birimi (28x28 piksel) burada işlenir.
    - Aktivasyon fonksiyonu olarak ReLU (Rectified Linear Unit) kullanılmıştır.
2. **İlk Gizli Katman:**

    - 392 nöron içerir. Bu katman, giriş verisindeki ilişkileri öğrenir.
    - Dropout (0.2): Aşırı öğrenmeyi önlemek için bazı nöronlar rastgele devre dışı bırakılır.
3. **İkinci Gizli Katman:**
    - 196 nöron içerir ve yine ReLU aktivasyon fonksiyonu kullanır.
    - Dropout (0.2): Aynı şekilde, aşırı öğrenmeyi engeller.
4. **Çıkış Katmanı:**
    - 10 nöron içerir (her nöron 0-9 arasındaki bir rakamı temsil eder).
    - Aktivasyon fonksiyonu: Softmax. Bu fonksiyon, modelin her rakama ait olasılıklarını hesaplar.

Optimizasyon ve Kayıp Hesaplama:

- **Adam Optimizatörü:** Modelin ağırlıklarını en verimli şekilde günceller.
- **Sparse Categorical Crossentropy:** Doğru sınıf ile tahmin arasındaki farkı hesaplar.
- Modelin başarı kriteri: Accuracy (doğruluk oranı).

Modelin Eğitimi

Model, eğitim verileriyle 10 epoch boyunca eğitilmiştir.

- Her epoch'ta model, eğitim verilerinden öğrenir ve test verisiyle doğrulama yapar.
- Eğitim sürecinde, kayıp ve doğruluk değerleri sürekli olarak raporlanır.

Eğitim Sürecinden Notlar:

- Daha fazla epoch ile model daha iyi öğrenebilir, ancak aşırı öğrenmeye dikkat edilmelidir.
- Dropout katmanları, modelin genel performansını artırmaya yardımcı olmuştur.

Tahmin ve Görselleştirme

Eğitilen model, test verileriyle test edilmiştir. Test sonuçlarında:

- Model, belirli bir doğruluk oranı (%90+ gibi) elde etmiştir.
- Rastgele bir test görüntüsü seçilip modele verilmiş, modelin tahmini ve gerçek etiket karşılaştırılmıştır.

Modelin Kaydedilmesi

Eğitilen model, gelecekte tekrar kullanılabilmesi için bir dosyaya kaydedilmiştir:

- Dosya adı: modelim.h5
- Bu dosya, daha sonra aynı modeli tekrar yüklemek ve yeni tahminlerde kullanmak için uygundur.

Sonuç

Bu proje, yapay sinir ağları ile el yazısı rakam sınıflandırmasının temel bir uygulamasıdır. Eğitim verileriyle başarılı bir şekilde öğrenen model, test verileri üzerinde oldukça iyi bir doğruluk oranına ulaşmıştır.

Elde Edilen Başarılar:

- Model, veri işleme ve sınıflandırmada etkin çalışmıştır.
- Eğitim ve test verileri arasında uyum sağlanmıştır.

Görseller:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
Inputs (Dense)               (None, 784)               615440
_________________________________________________________________
Hidden1 (Dense)              (None, 392)               307720
_________________________________________________________________
Dropou1 (Dropout)            (None, 392)               0
_________________________________________________________________
Hidden2 (Dense)              (None, 196)               77028
_________________________________________________________________
Dropout2 (Dropout)           (None, 196)               0
_________________________________________________________________
Output (Dense)               (None, 10)                1970
=================================================================
Total params: 1,002,158
Trainable params: 1,002,158
Non-trainable params: 0
_________________________________________________________________
```

Modelin katman yapısı, her katmandaki parametre sayısı ve toplam öğrenilebilir parametreler

```
Train on 60000 samples, validate on 10000 samples
Epoch 1/10
60000/60000 [==============================] - 6s 105us/sample - loss: 0.0406 - accuracy: 0.9883 - val_loss: 0.0850 - val_accuracy: 0.9814
Epoch 2/10
60000/60000 [==============================] - 8s 125us/sample - loss: 0.0387 - accuracy: 0.9883 - val_loss: 0.0785 - val_accuracy: 0.9802
Epoch 3/10
60000/60000 [==============================] - 8s 133us/sample - loss: 0.0310 - accuracy: 0.9909 - val_loss: 0.1128 - val_accuracy: 0.9747
Epoch 4/10
60000/60000 [==============================] - 7s 120us/sample - loss: 0.0300 - accuracy: 0.9913 - val_loss: 0.0881 - val_accuracy: 0.9805
Epoch 5/10
60000/60000 [==============================] - 6s 108us/sample - loss: 0.0253 - accuracy: 0.9927 - val_loss: 0.0821 - val_accuracy: 0.9841
Epoch 6/10
60000/60000 [==============================] - 7s 110us/sample - loss: 0.0291 - accuracy: 0.9923 - val_loss: 0.0782 - val_accuracy: 0.9835
Epoch 7/10
60000/60000 [==============================] - 6s 105us/sample - loss: 0.0239 - accuracy: 0.9931 - val_loss: 0.0964 - val_accuracy: 0.9815
Epoch 8/10
60000/60000 [==============================] - 6s 107us/sample - loss: 0.0216 - accuracy: 0.9940 - val_loss: 0.1329 - val_accuracy: 0.9818
Epoch 9/10
60000/60000 [==============================] - 6s 107us/sample - loss: 0.0258 - accuracy: 0.9933 - val_loss: 0.1008 - val_accuracy: 0.9838
Epoch 10/10
60000/60000 [==============================] - 6s 107us/sample - loss: 0.0197 - accuracy: 0.9948 - val_loss: 0.1685 - val_accuracy: 0.9794
10000/10000 [==============================] - 1s 65us/sample - loss: 0.1685 - accuracy: 0.9794
Trained model, accuracy: 97.94
```

Modelin 10 epoch boyunca eğitim ve doğrulama süreci gösterilmiş, eğitim sonunda %97.94 doğruluk oranı elde edilmiştir

```
(train_images,train_labels),(test_images,test_labels)=datasets.mnist.load_data()

plot.imshow(train_images[0],cmap="Greys")
print("Label",train_labels[0])
```

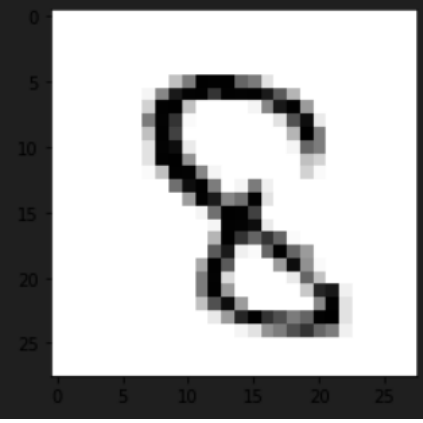
```
Label 5
```

MNIST veri setinden alınan bir örnek

```
image_index=1259
pred = model.predict(test_images[image_index].reshape(-1,shape_size))

plot.imshow(test_images[image_index].reshape(28, 28),cmap='Greys')
print("Label :",test_labels[image_index]," Predict :",pred.argmax())
```

```
Label : 8  Predict : 8
```



MNIST veri setinden alınan bir örnek