T.C. KÜTAHYA DUMLUPINAR ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ BİLGİSAYAR MÜHENDİSLİĞİ

YÜKSEK DÜZEY PROGRAMLAMA PROJE ÖDEVİ

DIGIT RECOGNIZER

Şerife Sümeyye OCAK
202013171046

DOÇ.DR HASAN TEMURTAŞ

Digit Recognition with CNN (Convolutional Neural Networks)

Projenin Amacı

Bu projede amaç, el yazısı ile yazılmış rakamları tanımak ve doğru şekilde sınıflandırmaktır. Proje, el yazısı rakamları tanıma alanında yaygın olarak kullanılan MNIST veri setini kullanarak, derin öğrenme yöntemlerinden biri olan Konvolüsyonel Sinir Ağları (CNN) ile bu problemi çözmeyi hedeflemektedir.

Kullanılan Veri

Proje için kullanılan veri seti **MNIST (Modified National Institute of Standards and Technology)** veri setidir. MNIST, el yazısı ile yazılmış 70.000 adet 28x28 boyutlarında gri tonlama görüntü içerir. Bu veri seti, 10 farklı rakam (0-9) içeren etiketli görüntülerden oluşur.

- **Eğitim Verisi**: 42.000 etiketli görüntü
- **Test Verisi**: 28.000 etiketli görüntü

Kullanılan Modeller

Bu projede, el yazısı rakamlarını tanımak amacıyla **Konvolüsyonel Sinir Ağı (CNN)** kullanılmıştır. CNN, görüntü işleme ve tanıma görevlerinde oldukça etkili bir derin öğrenme modelidir. CNN'in temel bileşenleri şunlardır:

1. **Convolution (Konvolüsyon) Katmanları**: Görüntü üzerinde filtreler uygulayarak önemli özellikleri çıkarır.
2. **Pooling (Havuzlama) Katmanları**: Veriyi özetleyerek boyutlarını küçültür.
3. **Flatten Katmanı**: Çıkartılan özelliklerin tek boyutlu bir vektöre dönüştürülmesini sağlar.
4. **Dense (Tam Bağlantılı) Katmanları**: Sınıflandırma için sonuca ulaşılmasını sağlar.

Projenin CNN yapısı, dört konvolüsyonel katman, her birinden sonra max-pooling katmanı ve dropout (aşırı öğrenmeyi engellemek için) kullanarak tasarlanmıştır. Son olarak, tam bağlantılı bir katman ile çıktılar elde edilmiştir.

Kullandığınız Kütüphaneler

Proje kapsamında aşağıdaki kütüphaneler kullanılmıştır:

- **TensorFlow** ve **Keras**: Derin öğrenme modelinin oluşturulması ve eğitilmesi için kullanılmıştır.
- **NumPy**: Veri işleme ve matematiksel hesaplamalar için kullanılmıştır.
- **Matplotlib ve Seaborn**: Veri görselleştirme ve analiz için kullanılmıştır.

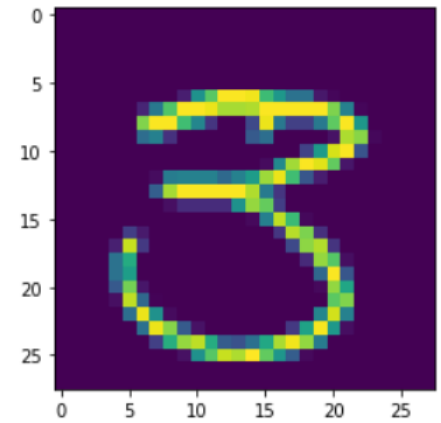
Gerekli Kütüphanelerin Yüklenmesi
Projede kullanılan başlıca Python kütüphaneleri aşağıda sıralanmıştır:

```
[ ]  import numpy as np # linear algebra
     import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
     import matplotlib.pyplot as plt
     import seaborn as sns
     import tensorflow as tf
     from tensorflow import keras
     from tensorflow.keras.preprocessing.image import ImageDataGenerator


     import os
     for dirname, _, filenames in os.walk('/kaggle/input'):
         for filename in filenames:
             print(os.path.join(dirname, filename))

     import warnings
     warnings.simplefilter(action='ignore', category=Warning)
```

Veri Hazırlığı: Eğitim ve test verileri uygun şekilde yüklendi ve normalize edildi.

```
[8]  plt.imshow(X[7].reshape([28,28]))
                                                                    Python
```

```
...  <matplotlib.image.AxesImage at 0x7f0bc7b29c90>
```



Model Tanımlaması: Konvolüsyonel Sinir Ağı modeli Keras ile tanımlandı.
Model, aşağıdaki gibi Keras ile tanımlanmıştır:

```
[ ]  model = tf.keras.models.Sequential([
        tf.keras.layers.Conv2D(64, (3,3), padding = 'same', activation='relu', input_shape=(28, 28, 1)),
        tf.keras.layers.MaxPooling2D(2, 2),
        tf.keras.layers.Dropout(0.25),
        tf.keras.layers.Conv2D(64, (3,3), padding = 'same', activation='relu'),
        tf.keras.layers.MaxPooling2D(2,2),
        tf.keras.layers.Dropout(0.25),
        tf.keras.layers.Conv2D(128, (3,3), padding = 'same', activation='relu'),
        tf.keras.layers.MaxPooling2D(2,2),
        tf.keras.layers.Dropout(0.25),
        tf.keras.layers.Conv2D(128, (3,3), padding = 'same', activation='relu'),
        tf.keras.layers.MaxPooling2D(2,2),
        tf.keras.layers.Dropout(0.25),
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(256, activation='relu'),
        tf.keras.layers.Dropout(0.5),
        tf.keras.layers.Dense(10, activation='softmax')
    ])
```

Modelin Derlenmesi: Model, **Adam optimizer** ve **categorical crossentropy loss** fonksiyonu ile derlendi.

```
[ ]  from keras.optimizers import Adam
     optimizer = Adam(lr=0.001, beta_1=0.9, beta_2=0.999)
     model.compile(optimizer = optimizer,
                   loss = 'categorical_crossentropy',
                   metrics = ['accuracy'])
```

Model Eğitimi:
Modelin eğitimi sırasında kullanılan hiperparametreler:

- **Batch Size**: 64
- **Epochs**: 30
- **Optimizer**: Adam (öğrenme oranı 0.001)

Modelin eğitim aşamasında, **ReduceLROnPlateau** ile öğrenme oranı düşürülmüştür:

```
[ ]  from keras.callbacks import ReduceLROnPlateau
     learning_rate_reduction = ReduceLROnPlateau(monitor='val_acc',
                                                 patience=3,
                                                 verbose=1,
                                                 factor=0.6,
                                                 min_lr=0.00001)
```

```
[ ]  history = model.fit(
                train_datagen.flow(x_train,y_train,batch_size = batch_size),
                validation_data = (x_test,y_test),
                batch_size = batch_size,
                steps_per_epoch = x_train.shape[0]//batch_size,
                epochs = epochs,
                verbose = 1,
                callbacks=[learning_rate_reduction]
                )
```

Modelin Derlenmesi ve Eğitim

```
model.compile(optimizer=optimizer,
        loss='categorical_crossentropy',
        metrics=['accuracy'])
```

Eğitimde kullanılan **learning rate reduction** (öğrenme oranı azaltma) stratejisi de, modelin doğruluğu doğrulama veri seti üzerinde iyileşmezse, öğrenme oranını azaltarak eğitim sürecini iyileştirmeyi hedeflemiştir.

Model Performansı

Modelin doğruluğu, eğitim sürecinin sonunda elde edilen sonuçlarla belirlenmiştir. Modelin genel başarısı, **accuracy** metriği ile değerlendirilmiştir. Eğitilen model, test verisi üzerinde %98 başarı oranına ulaşmıştır.

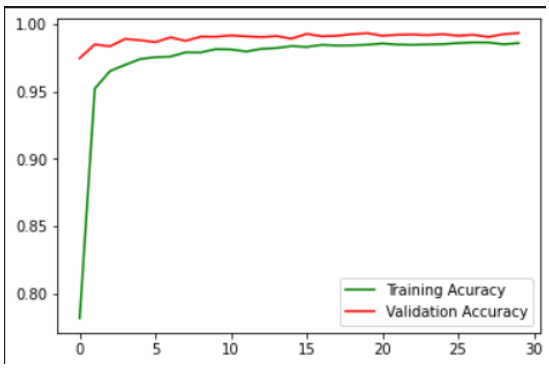
Modelin doğruluk sonuçları aşağıdaki gibi gösterilebilir:

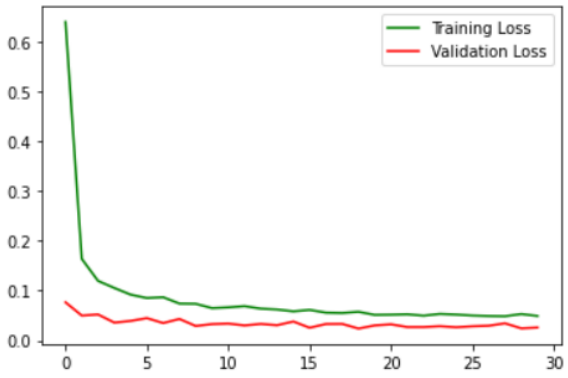
```
[ ]  model.evaluate(x_test,y_test)
```

Sonuçlar ve Çıktılar

Modelin eğitilmesi tamamlandıktan sonra, aşağıdaki çıktılar elde edilmiştir:

1. **Eğitim doğruluğu (Accuracy)**: Model, eğitim verileri üzerinde yaklaşık %99 doğruluk sağlamıştır.
2. **Test doğruluğu (Accuracy)**: Test verileri üzerinde model, %98 doğruluk oranı ile başarılı olmuştur.
3. **Confusion Matrix**: Test sonuçlarını detaylandırarak, hangi rakamların doğru veya yanlış sınıflandırıldığını görmek için kullanılmıştır.





Sonuç

Bu proje, el yazısı rakamlarının doğru bir şekilde sınıflandırılmasını sağlayan bir Konvolüsyonel Sinir Ağı (CNN) modelinin geliştirilmesi sürecini sunmuştur. Model, MNIST veri setinde yüksek başarı oranı elde etmiştir.