İSTANBUL SABAHATTİN ZAİM ÜNİVERSİTESİ
MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ
MAKİNE ÖĞRENMESİNE GİRİŞ

MNİST EL YAZISI RAKAMLARININ TANIMLANMASI

030115011-İREM NUR KÜÇÜKENEZ

030115032 -ŞUHEDA AKTAŞ

030115211-MUKADDES DEMİRTAŞ

İstanbul Sabahattin Zaim Üniversitesi

Giriş

MNIST el yazısı ile rakam tanıma veri setidir. Bu veri setinin amacı, 0-9 arasındaki el yazısı rakamlarını doğru bir şekilde tanımlanma yapmasını sağlamaktır.
Bununla birlikte sonuç olarak doğruluk oranlarının verilmesi hedeflerimiz arasındadır.

Literatür Taraması

Daha önce CNN mimarisi ile yapılan çalışmalardan biri Orhan Gazi Yalçın tarafından yapılmış Image Classification in 10 Minutes with MNIST Dataset makalesinde açıklanmıştır. Bu çalışmada MNIST veri seti ve çok katmanlı bir mimari kullanılmıştır. Çalışma Python ortamında yazılmış olup Keras ve Tensorflow kütüphaneleri ile yapılmıştır.

Bu çalışmada data setin aktarılması ve sonrasında görüntüleri yeniden şekillendirme ve Normalleştirme işlemleri yapılmıştır. Sonrasında Konvolüsyonlu sinir ağının oluşturulması aşamasına geçilmiştir. Conv2D, MaxPooling, Flatten, Dropout katmanları kullanılmıştır.10 nörondan oluşmaktadır çünkü 10 adet sınıf mevcuttur. Bundan sonraki aşama ise modelin eğitimi ve test aşamasıdır. Eğitim ve test olmak üzere datasetin ikiye ayrılması ve bu değerler doğrultusunda eğitimin gerçekleştirimi sağlanmıştır. Bu çalışma sonucunda % 98,5 doğruluğa ulaşılmıştır.[1]

Veri Seti

MNIST veri kümesi, ABD Ulusal Standartlar ve Teknoloji Enstitüsü'nün (NIST) iki veri kümesinden oluşturulmuştur. Eğitim seti 250 farklı kişiden, yüzde 50 lise öğrencisi ve Nüfus Sayımı Bürosu'ndan yüzde 50 çalışandan oluşan el yazısı rakamlarından oluşmaktadır.

MNIST veri kümesi http://yann.lecun.com/exdb/mnist/ adresinde kamuya açıktır ve aşağıdaki dört bölümden oluşur: - Eğitim seti görüntüleri: train-images-idx3-ubyte.gz (9.9 MB, 47 MB sıkıştırılmış ve 60.000 numune) - Eğitim seti etiketleri: train-labels-idx1-ubyte.gz (29 KB, 60 KB sıkıştırılmış ve 60.000 etiket) - Test seti görüntüleri: t10k-images-idx3-ubyte.gz (1.6 MB, 7.8 MB, sıkıştırılmamış ve 10.000 örnek) - Test kümesi etiketleri: t10k-labels-idx1-ubyte.gz (5 KB, 10 KB sıkıştırılmış ve 10.000 etiket).

Ayrıca csv formatlarına erişim için Kaggle.com dan MNIST veri dosyaları indirilmiştir.[2]

Her biri 28×28 pikselden oluşan, gri tonlarında el yazısı rakamlar. 250 kişiden alınan toplam 60.000 numune mevcut. Bir veri satırının ilk sütununda resimde gösterilen rakam, kalan 784 sütunda ise piksellerin satır satır yan yana konmuş halde gri tonlarının sayı değerleri var.

Yöntem

KNN YÖNTEMİ- İrem Nur Küçükenez

kNN algoritması, tüm mevcut durumları depolayan ve bir benzerlik ölçüsüne (ör. Mesafe fonksiyonları) göre yeni durumları sınıflandıran bir algoritmadır.

Sınıflandırılacak veririn komşularıyla olan uzaklığına bakar ve en uygun etiket ile sınıflandırma işlemini yapar.

Hedefimiz, ham piksel yoğunlukları üzerine bir kNN sınıflandırıcı yetiştirmek ve ardından bilinmeyen rakamları sınıflandırmaktır.

Bu hedefe ulaşmak için, görüntü sınıflandırıcıları yetiştirmek için beş aşama üzerinden gidilmiştir.

Adım1-İlk veri setimizi yapılandırmak: Veri setimiz 0-9 sayılarını temsil eden rakamlardan oluşur. Bu görüntüler gri tonlamalı, 28 x 28 görüntüler ve rakamlar siyah bir arka plan üzerinde beyaz olarak görmektedir. Bu sayılar, ayrıca sınıflandırma işimizi biraz daha kolaylaştırmakta çünkü veri setimiz yoğun bir şekilde önceden işlenmiş, sıralanmış ve ortalanmış durumdadır.

Adım 2- Veri setini bölme: İlk set, k-NN sınıflandırıcımızı eğitmek için kullanılan eğitim setimizdir. Ayrıca, k nın en iyi değeri bulmak için bir doğrulama seti kullanacağız. Sonunda, test setini kullanarak sınıflandırıcımızı değerlendireceğiz.

Adım 3- Ayıklama özellikleri: Her basamağı temsil etmek ve karakterize etmek için özellikler çıkarmak yerine, görüntünün yalnızca gri tonlamalı piksel yoğunluğunu kullanırız.

Adım 4- Sınıflandırma modelimizin eğitimi: k-NN sınıflandırıcımız, eğitim setindeki görüntülerin ham piksel yoğunlukları konusunda eğitilecektir. Doğrulama setini kullanarak en iyi k değerini belirleyeceğiz.

Adım 5- Sınıflandırıcımızı değerlendirme: k'nin en iyi değerini bulduğumuzda, test setimizdeki k-NN sınıflandırıcımızı değerlendirebiliriz.

Naive-Bayes YÖNTEMİ- Mukaddes Demirtaş

Naive Bayes sınıflandırıcısının temeli Bayes teoremine dayanır. Bir öğrenme algoritmasıdır aynı zamanda dengesiz veri kümelerinde de çalışabilir. Algoritmanın çalışma şekli bir eleman için her durumun olasılığını hesaplar ve olasılık değeri en yüksek olana göre sınıflandırır. Az bir eğitim verisiyle çok başarılı işler çıkartabilir. Test kümesindeki bir değerin eğitim kümesinde gözlemlenemeyen bir değeri varsa olasılık değeri olarak 0 verir yani tahmin yapamaz

Naive-Bayes yöntemi için verilimizi csv formatlı dosyamızdan okumakla işe başlıyoruz. Sonrasında veri kümemizi test ve train şekinde bölüyoruz. Veriler 28X28 piksel veridir (yükseklik ve genişlik) ve her veri noktası 0 ile 255 arasındaki tamsayı olarak ağırlıklandırılır. Bu verileri tek bir 784 piksel veya veri dizisine dönüştüreceğiz. Görsel olarak, “piksel” önekini çıkarırsak, pikseller görüntüyü şu şekilde oluşturur:

Naive bayes sınıfımızı import ediliyor, Naive bayes türü olarak multinominalNaiveBayes seçilir.

MultinomialNB: Tahmin edeceğiniz veri veya kolon nominal ise (Int sayılar) ise kullanılması daha makuldur.

MuktinomialNB sınıfından bir nesne ürettip, makineyi eğitiyoruz. Sonrasında, tahminleri test verilerine uygularız. Modelimizin doğruluğunu değerlendiriyoruz.

```
000 001 002 003 ... 026 027
028 029 030 031 ... 054 055
056 057 058 059 ... 082 083
 |   |   |   | ...  |   |
728 729 730 731 ... 754 755
756 757 758 759 ... 782 783
```

Şekil 1 Piksellerin görüntüyü gösterme şekli

CNN YÖNTEMİ- Şuheda Aktaş

Convolutional Neural Network (CNN) Türkçesi Konvülasyonel (Evrişimsel) Sinir Ağı'dır. Neural Networks temelde insan gibi olayları öğrenebilmeyi hedefleyen bir modeldir. Temel fikir, düğümlerin 3 katmanlı bir yapı oluşturulması iddiasına dayanır. Her düğümün içerisinde paralel şekilde çalışan yüksek işlem kabiliyetine sahip işlemciler bulunmaktadır. Ancak temelde düğümlerin insan beynine benzer bir biçimde çalışması hedeflenir. 3 katmandan birisi İnput katmanı, Output katmanı ve Hidden(saklı) katmanından oluşur.

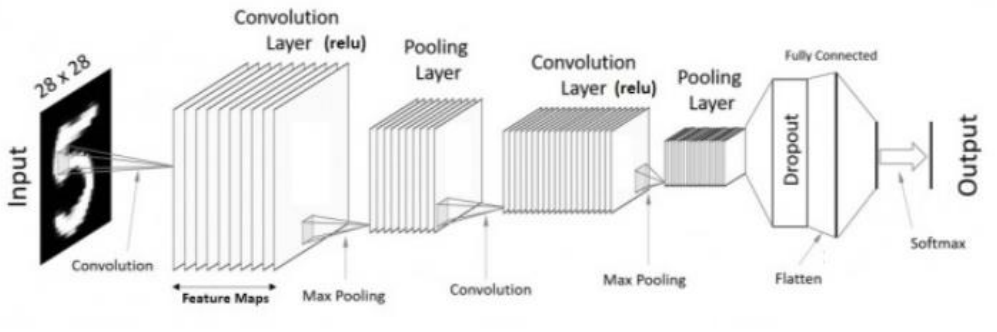
Pandas kütüphanesi yardımıyla indirdiğimiz eğitim ve test csv dataları projeye yüklenir.

Sonrasında görsellerimize normalization işlemi yapılır çünkü farklı renklerden kaynaklı belli başlı hatalar elde edilebilir bu hataların önüne geçmek amaçlı bu işlem yapılır. Sonraki adımımız ise reshape'tir. Bunun amacı kerasa üç boyutlu metriks verilmesidir iki boyutluda işlem yapılmamaktadır. Sınıflandırma için label encoding işlemi yapılır. Ve kategorik olarak ayrılması adına kerasın categorical fonksiyonu kullanılır. Datamızın eğitimden önceki işlemleri bunlardır. Bundan sonra modelimize ve eğitimize geçiyoruz.

Convolution Layer(relu); Filtreleme ile datasetteki özniteliklerin farklı özelliklerini ayırt edilmesi sağlanmıştır.

Pooling Layer; convolution layerın size'nın küçültülme işlemidir. Ayırt edici özelliklerin tamamı yerine bir kaçının tutulmasıdır.

Modelin son katmanını için burada aktivasyon fonksiyonu için Softmax fonksiyonunu kullanıyoruz. Softmax fonksiyonu sonucunda test girdisinin her bir sınıfa ait benzerliği için olasılık değeri üretilir. Yapay sinir ağı modelinin çıktı olarak verdiği skor değerler normalize ederek olasılık değerlerine dönüşmektedir; yani en az farka sahip, en çok benzerliğe sahip sınıfı bulmaktır.



Konvolüsyon İşlem Adımları

Feauture dedektörü = çekirdek = filtre
Feauture dedektörü kenarları veya dışbükey şekiller gibi özellikleri algılar.
Feauture haritası = conv (giriş görüntüsü, özellik dedektörü). Matrislerin eleman çarpımı.
Feauture haritası = convolved özelliği
Epoch = giriş görüntüsünde gezinme .
Görüntünün boyutunu azaltıyoruz.
Çoklu özellik haritaları oluşturuyoruz, çoklu özellik dedektörleri(filtreler) kullanıyoruz.
Relu aktivasyon fonksiyonunun kullanılması.
Relu; Yapay sinir ağlarına doğrusal olmayan gerçek dünya özelliklerini tanıtmak için aktivasyon fonksiyonuna ihtiyaç duyulur. Temel olarak basit bir yapay sinir ağında x girdiler, w ağırlıklar olarak tanımlanır ve ağın çıkışına aktarılan değere f(x) yani aktivasyon fonksiyonu uygulanır. Bizde modelimizde Relu aktivasyon fonksiyonunu uyguladık. Relu(Rectified Linear Units) fonksiyonu, sıfırın altındaki değerler sıfır, sıfırın üzerindeki değerlere ise kendi değerini atayan bir fonksiyondur.
CNN Mimarimiz→
conv => max pool => dropout => conv => max pool => dropout => fully connected (2 layer)

Deneysel Sonuçlar

1.İrem Nur Küçükenez – Knn Yöntemi

k=1, accuracy=99.26%
k=3, accuracy=99.26%
k=5, accuracy=99.26%
k=7, accuracy=99.26%
k=9, accuracy=99.26%
k=11, accuracy=99.26%
k=13, accuracy=99.26%
k=15, accuracy=99.26%
k=17, accuracy=98.52%
k=19, accuracy=98.52%
k=21, accuracy=97.78%
k=23, accuracy=97.04%
k=25, accuracy=97.78%
k=27, accuracy=97.04%
k=29, accuracy=97.

2.Mukaddes Demirtaş -Naive Bayes

Doğruluk = 0.83283283283283283

3.Şuheda Aktaş -CNN

Epoch değeri=10

```
Epoch 10/10

151/151 [==============================] - 13s 84ms/step - loss: 1.2685 - acc:
0.5674 - val_loss: 0.2416 - val_acc: 0.9383
```

Epoch değeri=20

```
Epoch 20/20

151/151 [==============================] - 22s 149ms/step - loss: 1.1564 - acc:
0.6041 - val_loss: 0.1756 - val_acc: 0.9543
```

Epoch değeri=30

```
Epoch 30/30

151/151 [==============================] - 22s 145ms/step - loss: 1.0912 - acc:
0.6282 - val_loss: 0.1848 - val_acc: 0.9510
```

Epoch değeri=40

```
Epoch 40/40

151/151 [==============================] - 15s 101ms/step - loss: 1.0418 - acc: 0.6439 - val_loss: 0.1522 - val_acc: 0.9543
```

Sonuç ve Değerlendirme

Ham piksel yoğunlukları arasındaki Öklid mesafesini hesaplayarak %99'luk çok yüksek bir hassasiyet elde ettik.

Naive-Bayes algoritması için sonuç; MNIST veri setinde multinomial NaiveBayes kullanılmıştır.Doğruluk olarak %83' lük gibi bir değer elde ettik.

CNN mimarisi MNIST veri setinde %95'lik bir başarı elde edilmiştir. En iyi sonucu alabilmek adına katmanlı mimari tercih edilmiştir. Epoch değerlerinin artımıyla başarı oranın değişimi gözlenmiştir.

Literatür taraması ve kendi içimizde yaptığımız yöntemler bazında karşılaştırırsak MNIST veri seti için knn yönteminin kullanılması daha uygundur en yüksek başarımı knn yöntemiyle elde etmekteyiz.

Kaynaklar

[1]Yalçın, O. G. Image Classification in 10 Minutes with MNIST Dataset. *2018*.

[2]Ergun, E. . https://www.kaggle.com/efeergun96/simple-deep-mnist/data.