**El Yazısını Dijitale Çevirme için Evrişimsel Sinir Ağı Uygulaması**

**Fırat Üniversitesi Teknoloji Fakültesi Yazılım Mühendisliği Bölümü**

210541056@firat.edu.tr

**Öz**

Bu çalışma, Evrişimsel Sinir Ağlarını (CNN) kullanarak kullanıcıların el yazısıyla yazdığı belgeleri ve yazıları dijital ortama aktarmayı amaçlamaktadır. Proje, el yazısı karakter tanıma problemine odaklanarak, 26 büyük harf, 26 küçük harf ve 10 rakam olmak üzere toplam 62 farklı sınıfı tanımak üzere bir CNN modeli geliştirmiştir. Çalışmada, Kaggle platformunda bulunan "Handwritten English Characters and Digits" veri seti kullanılmıştır. Geliştirilen model, %88.6 genel doğruluk oranı ile başarılı bir performans sergilemiştir. Elde edilen bu başarı, el yazısı tanıma alanında CNN'lerin etkinliğini bir kez daha göstermektedir. Ancak, modelin performansını daha da artırmak ve potansiyel sınırlılıklarını gidermek için ileriye dönük iyileştirme alanları da belirlenmiştir.

**1. Giriş**

Evrişimsel Sinir Ağı (Convolutional Neural Network, CNN), bilgisayarla görme (Computer Vision) projelerinde önemli bir yer tutan ve derin öğrenme alanında yaygın olarak kullanılan bir algoritmadır. CNN'ler, geleneksel yapay sinir ağlarına kıyasla çok daha etkin bir şekilde görüntüler üzerinde çalışabilmektedir. Bu mimari, girdi görüntülerindeki uzaysal ve zamansal bilgileri koruyarak anlamlı özelliklerin çıkarılmasına olanak tanır. CNN'lerin başarısının temelinde, evrişim katmanları, havuzlama (pooling) katmanları ve tam bağlantılı (fully connected) katmanlar gibi özel yapısal bileşenler yer almaktadır. Evrişim katmanları, görüntüdeki yerel özellikleri tespit ederken, havuzlama katmanları bu özelliklerin boyutunu azaltarak hesaplama verimliliğini artırır. Tam bağlantılı katmanlar ise bu özellikleri sınıflandırma veya tahmin etme işlemlerinde kullanır [1].

CNN'lerin başarısının arkasında, büyük veri setleri üzerinde derinlemesine öğrenme yapabilme kapasiteleri ve yüksek hesaplama gücü bulunmaktadır. GPU'ların (Graphics Processing Units) gelişimi ve derin öğrenme çatılarının (örneğin TensorFlow, PyTorch) yaygınlaşması, CNN'lerin daha karmaşık ve büyük modeller olarak tasarlanmasına olanak tanımış, böylece performanslarını daha da artırmıştır. Bunun yanı sıra, transfer öğrenme ve veri artırma gibi teknikler, sınırlı veri ile çalışırken bile CNN'lerin etkili olmasını sağlamaktadır [2].

CNN'lerin kullanım alanları oldukça geniştir. Görüntü tanıma ve nesne algılama dışında, yüz tanıma, el yazısı tanıma, tıbbi görüntü analizi, otonom araçlarda çevre algılama, video analizi ve artırılmış gerçeklik gibi birçok alanda etkin bir şekilde kullanılmaktadır.

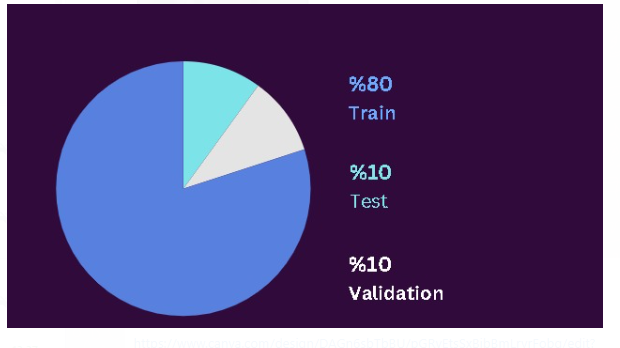
Bu çalışmada, CNN'lerin temel mimarisi arkasındaki matematik ile detaylı bir şekilde incelenmeye çalışılmış ve kodlama tarafında Tensorflow PyTorch gibi kütüphaneler kullanılmadan görüntü sınıflandırma problemlerinde nasıl uygulandığına dair bir örnek sunulmuştur. Ayrıca, bir görüntü işleme problemi için uygulanan CNN tabanlı yaklaşım adım adım incelenmiş, modelin eğitimi ve performans metriklerinin değerlendirmesi süreçleri açıklanmıştır.

**2. Materyal ve Metot**

**2.1. Veri Seti**

Bu çalışmada, el yazısı karakter tanıma görevi için "Handwritten English Characters and Digits" veri seti kullanılmıştır. Bu veri seti, Kaggle platformundan temin edilmiştir. Veri seti, toplam 62 farklı sınıfı içermektedir: 26 büyük harf (A-Z), 26 küçük harf (a-z) ve 10 rakam (0-9).

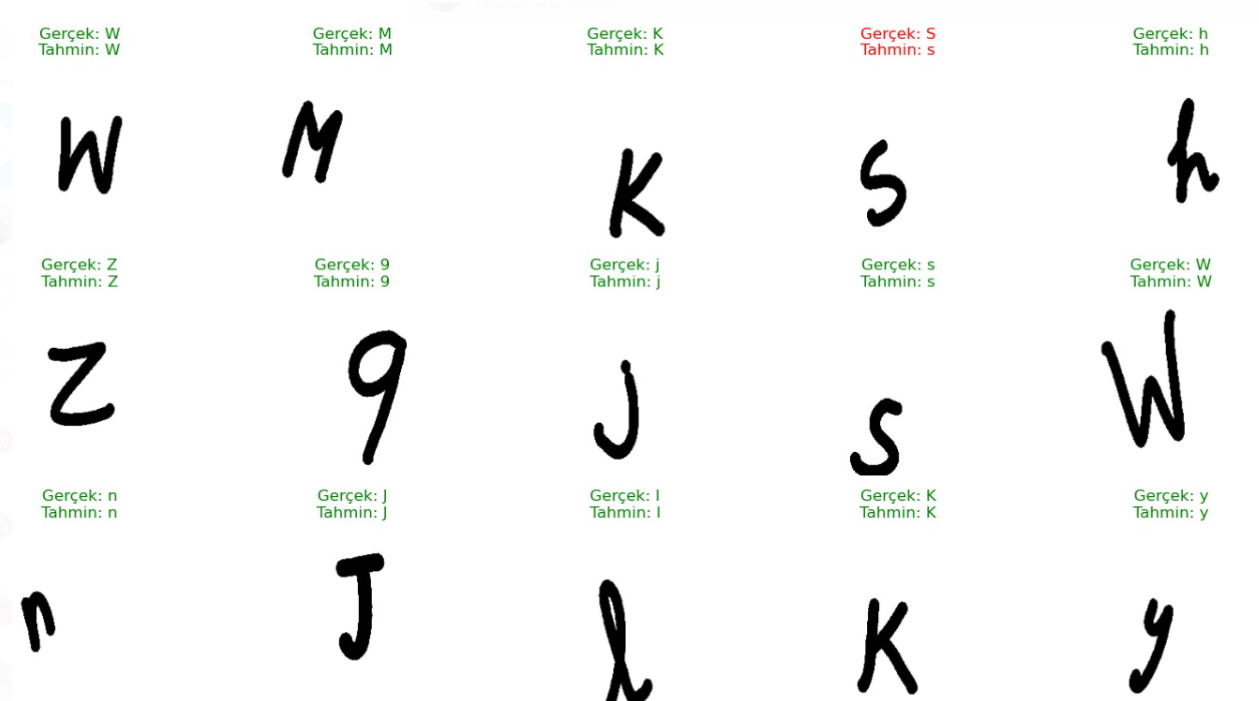
Çalışmada kullanılan veri seti, eğitim, doğrulama ve test olmak üzere üç ana bölüme ayrılmıştır. Veri setinin %80'i eğitim (%80 train) için, %10'u doğrulama (%10 validation) için ve kalan %10'u ise test (%10 test) için kullanılmıştır. Bu ayrım, modelin genellenebilirlik yeteneğini değerlendirmek ve aşırı öğrenmeyi (overfitting) önlemek amacıyla yapılmıştır.



**Şekil 1. HandWrite Dataset Veri Setinden Rastgele Elde Edilen 13639 görüntünün eğitim, doğrulama ve test şeklinde ayrılması**

|  |  |  |
| --- | --- | --- |
| **Etiket** | **Kategori** | **Görüntü Sayısı** |
| 0 | Rakam | 218 |
| 1 | Rakam | 221 |
| … | Rakam | 225 |
| 9 | Rakam | 198 |
| 10 | Büyük Harf | 224 |
| … | Büyük Harf | 217 |
| 35 | Büyük Harf | 235 |
| 36 | Küçük Harf | 248 |
| … | Küçük Harf | 189 |
| 62 | Küçük Harf | 197 |

**Tablo 1. Rakamların, büyük harflerin ve küçük harflerin 13640 görüntülü veri setinin kategorileri, sınıf etiketleri ve yaklaşık olarak görüntü sayıları.**



**Şekil 2.** **Handwritten english characters and digits veri setinden rastgele seçilen örnek görseller.**

**2.2. CNN**

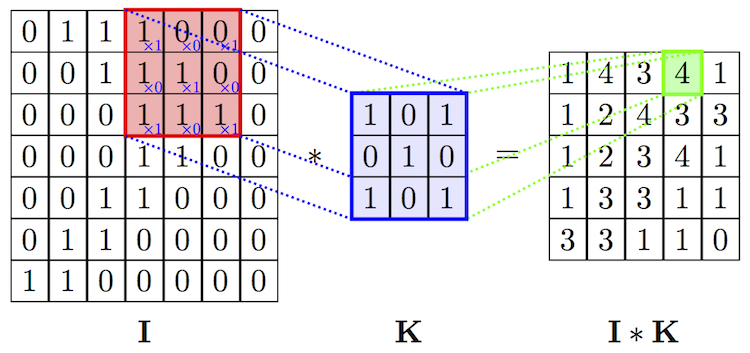
Bu bölümde, sıfırdan oluşturulan CNN mimarisinde kullanılan temel kavramlar detaylandırılmıştır.

**2.2.1. Convolution (Evrişim) Nedir?**

Evrişim, dijital görüntü işleme ve derin öğrenme alanlarında sıklıkla kullanılan temel bir işlemdir. Bu işlem, bir görüntü üzerinde belirli bir boyutta filtre (veya çekirdek, kernel) kaydırılarak, görüntünün farklı özelliklerini çıkarmayı amaçlar [4]. Evrişim işlemi, görüntüdeki kenarlar, dokular, renkler ve diğer önemli ayrıntıları tespit etmek için kullanılır. Evrişim işlemlerinin adımları sırasıyla şunlardır:

1. *Filtre Tanımlama*: Filtre, genellikle küçük boyutlu (3x3 veya 5x5) bir matristir. Her filtre, belirli bir özelliği vurgulamak için tasarlanmıştır. Örneğin, kenar algılama için farklı, bulanıklaştırma için farklı filtreler kullanılır.
2. *Filtrenin Görüntü Üzerinde Kaydırılması*: Filtre, giriş görüntüsünün sol üst köşesine yerleştirilir. Filtrenin her bir elemanı, görüntünün karşılık gelen bölgesiyle çarpılır. Çarpım sonuçları toplanarak yeni bir piksel değeri elde edilir. Filtre, görüntü boyunca belirli bir adım (stride) ile kaydırılır ve bu işlem tüm görüntü üzerinde tekrarlanır. Padding, görüntünün kenarlarına eklenen ekstra piksel sayısıdır ve genellikle görüntünün boyutunu kontrol etmek veya kenar bilgilerini korumak amacıyla kullanılır. Bu çalışmada, stride değeri 1 ve padding değeri 0 olarak belirlenmiştir.
3. *Sonuç*: Filtre uygulandıktan sonra, her bir konumda elde edilen değerler yeni bir görüntü oluşturur. Bu yeni görüntü, orijinal görüntünün belirli özelliklerini vurgular veya değiştirir.

Evrişim işlemine ait örnek bir görsel Şekil 3’te gösterilmektedir. Burada I imgeyi ve K ise filtreyi temsil etmektedir.



**Şekil 3. Evrişim işlemi örneği [1].**

**2.2.2. İleri Besleme (Forward Pass) Nedir?**

İleri besleme**,** yapay sinir ağlarında giriş verisinin ağ boyunca ileriye doğru taşınması ve her bir katmanda çıktı değerlerinin hesaplanması sürecidir. Bu süreç, ağın tahmin yapabilmesi için gereklidir ve genellikle eğitim (training) ve çıkarım (inference) aşamalarının temelini oluşturur. İleri besleme, ağın ağırlıklarını güncellemek için gerekli olan hatayı hesaplamadan önce gerçekleşir. İleri besleme adımları sırasıyla şöyledir:

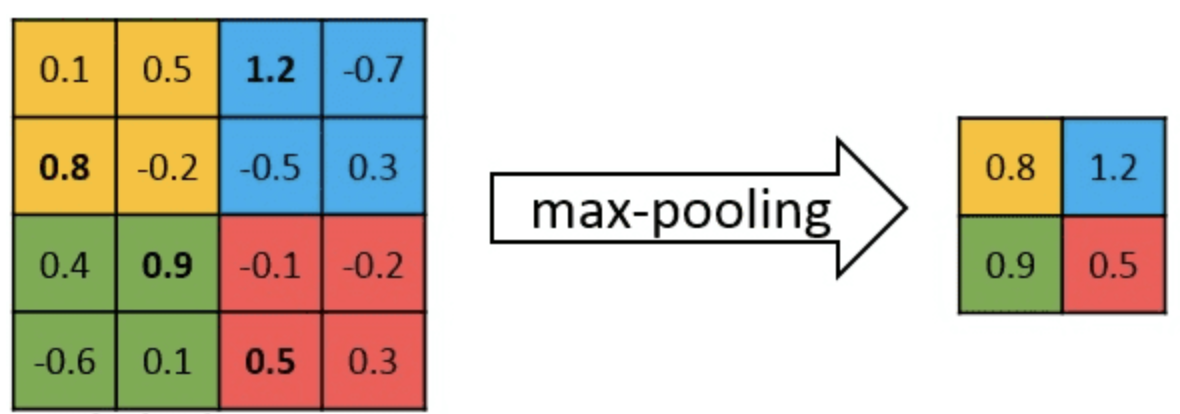
1. *Girdi Verisinin Alınması:* Ağın ilk katmanına (genellikle giriş katmanı) sunulan ham veri setidir. Bu, bir görüntü, metin veya herhangi bir sayısal veri olabilir. Örneğin, bir görüntü sınıflandırma ağında, giriş verisi genellikle piksel değerlerinden oluşan bir matristir.
2. *Evrişim Katmanları:* Girdi verisi, bir veya birden fazla evrişim katmanından geçirilir. Her evrişim katmanı, belirli sayıda filtre kullanarak giriş verisinin özelliklerini çıkarır. Filtreler, görüntüdeki kenarlar, dokular, renkler gibi düşük seviyeli özellikleri tespit eder. Matematiksel olarak, her filtre ile giriş verisi arasında evrişim işlemi gerçekleştirilir. Denklem 1’de gösterilmektedir. Burada “\*” konvolüsyon işlemini ve Bias filtreye eklenen sabit bir değerdir.

(1)

1. *Aktivasyon Fonksiyonları:* Her evrişim katmanının ardından, doğrusal olmayan bir aktivasyon fonksiyonu uygulanır. Aktivasyon fonksiyonları, ağa doğrusal olmayanlık kazandırarak daha karmaşık ilişkileri öğrenmesini sağlar. Yaygın kullanılan aktivasyon fonksiyonları arasında ReLU (Rectified Linear Unit) bulunur [5]. Denklem 2’de ReLU’ya ait eşitlik bulunmaktadır.

(2)

1. *Havuzlama Katmanları:* Evrişim katmanlarından sonra, boyut azaltma ve hesaplama maliyetini düşürmek için havuzlama katmanları kullanılır. Maksimum Havuzlama (Max Pooling) gibi teknikler uygulanarak, özellik haritalarının boyutu küçültülür [5], [6]. Max Pooling, matris üzerinde belirli boyutlarda kaydırılarak işlemi gerçekleştirir ve her bölgeden maksimum değeri seçerek yeni bir, daha küçük boyutlu çıktı matrisi oluşturur. Max Pooling’ e ait örnek Şekil 4’te gösterilmektedir.



**Şekil 4. Maksimum havuzlama örneği.**

1. *Tam Bağlantı Katmanları:* Evrişim ve havuzlama katmanlarından sonra, genellikle bir veya daha fazla tam bağlantılı (dense) katman bulunur.Bu katmanlar, öğrenilen özellikleri kullanarak nihai çıktıyı üretir.Tam bağlantılı katmanlarda, her nöron bir önceki katmandaki tüm nöronlara bağlıdır. Matematiksel olarak Denklem 3’te gösterildiği gibi ağrılık matrisi ile giriş verisi çarpılıp sabit bir değer olan bias eklenir.

(3)

1. *Çıkış Katmanı:* Ağın son katmanıdır ve modelin tahminini üretir. Sınıflandırma görevlerinde, genellikle Softmax aktivasyon fonksiyonu kullanılır. Denklem 4’te Softmax fonksiyonuna ait eşitlik bulunmaktadır. Bu fonksiyon, her sınıfa ait olma olasılığı hesaplar.

(4)

İleri beslemede, eğitim sırasında modelin Model, verilen bir giriş verisi için tahmin üretir.Bu tahmin, eğitim sürecinde modelin ne kadar doğru olduğunu değerlendirmek için kullanılır. Ardından tahmin edilen değer ile gerçek değer arasındaki fark hesaplanır. Bu farka hata denir. Hata, modelin performansını ölçmek için kullanılan bir kayıp fonksiyonu (loss function) ile hesaplanır. İleri besleme sırasında hesaplanan değerler geri yayılım algoritması için gerekli olan ara değerlerin hesaplanmasını sağlar. Bu ara değerler, hatanın her bir ağırlığa nasıl dağıldığını belirlemek için kullanılır.

**2.2.3. Backward Pass (Geri Besleme) Nedir?**

**Geri besleme**, yapay sinir ağlarının eğitim sürecinde kullanılan temel bir diğer adımdır. Bu süreç, modelin tahmin ettiği çıktılar ile gerçek değerler arasındaki farkın (hata oranının) hesaplanmasını ve bu hatanın ağ boyunca geriye doğru yayılması ile her bir ağırlığın güncellenmesini içerir. Geri besleme, ağın öğrenme kapasitesini artırmak ve performansını optimize etmek için ağırlıkların doğru şekilde ayarlanmasını sağlar [7], [8]. Bu sayede model, veri setindeki örneklerden öğrenerek daha doğru tahminler yapabilme yeteneğini geliştirir. Üç temel adımdan oluşur.

1. *Hata Oranının (Loss) Hesaplanması:* **Loss**, modelin tahmin ettiği değerler ile gerçek değerler arasındaki farkı ölçen bir metriktir. Bu metrik, modelin ne kadar doğru olduğunu belirlemek için kullanılır ve eğitim sürecinin temelini oluşturur. Sınıflandırma problemlerinde yaygın olarak Cross-Entropy Loss kullanılır. Denklem 5’te Cross-Entropy Loss’un matematiksel gösterimi yapılmıştır.

(5)

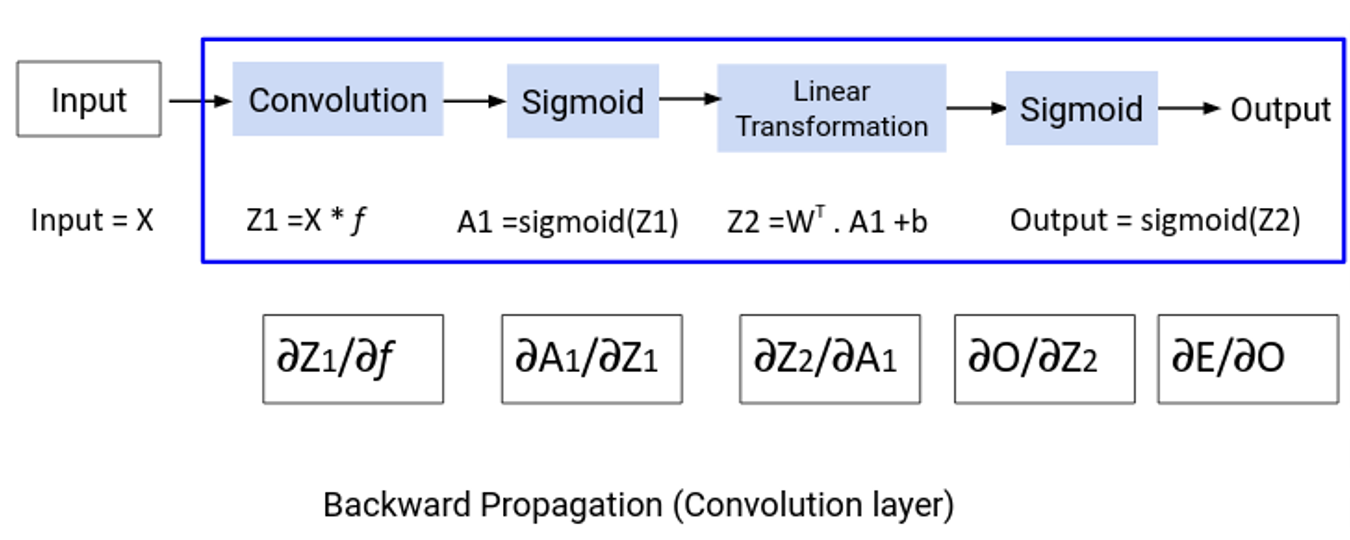
1. *Hata Oranının Zincir Kuralı ile Geri Yayılımı (Backpropagation):* **Backpropagation**, hata oranını ağ boyunca geriye doğru yayarak her bir ağırlığın hataya katkısını hesaplama sürecidir. Bu süreç, zincir kuralı (chain rule) kullanılarak gerçekleştirilir ve her bir ağırlığın gradyanı (türevi) hesaplanır. Backpropagation adımları şunlardır:
2. Çıkış Katmanından Başlama: Çıkış katmanındaki her bir nöron için, hata oranının ağırlıklarına göre türevi hesaplanır. Ardından çıkış katmanındaki her bir nöronun hata oranına olan katkısı belirlenir.
3. Gizli Katmanlara İlerleme: Hata, çıkış katmanından bir önceki gizli katmana doğru yayılır. Her bir gizli katmandaki ağırlıklar için gradyanlar hesaplanır. Bu, zincir kuralı kullanılarak yapılır. Denklem 6’da zincir kuralının matematiksel gösterimi bulunmaktadır. Burada “*L*” hata oranını, “*a”* aktivasyon fonksiyonunun çıktısını, “*z*” ise ağırlıklı toplamı temsil eder.

(6)

1. Gradientlerin Toplanması: Her bir ağırlık için hesaplanan gradyanlar toplanır ve ağırlıkların ne kadar ve hangi yönde güncelleneceği belirlenir. Gradyanlar, ağırlıkların hata oranını minimize edecek şekilde güncellenmesini sağlar.
2. *Ağırlıkların Güncellenmesi:* Hesaplanan gradyanlar kullanılarak, ağın ağırlıkları güncellenir. Bu güncelleme genellikle bir **optimizasyon algoritması** tarafından gerçekleştirilir. En yaygın kullanılan optimizasyon algoritması **Gradient Descent** ve onun çeşitli türevleridir. **Gradient Descent**, hata oranını minimize etmek için ağırlıkları gradyanların ters yönünde adım adım güncelleyen temel bir algoritmadır. Güncelleme kuralı Denklem 7’de gösterilmiştir. Burada *η* öğrenme oranını ve ∂w/∂L ise ağırlığın gradyanını ifade etmektedir.

(7)

Şekil 5’te örnek bir CNN ve tam bağlantı mimarisinin ileri ve geri beslemeye ait matematiksel olarak çalışmasını bir bütün olarak gösterilmektedir.



**Şekil 5. Örnek bir CNN ve tam bağlantı mimarisinin ileri ve geri beslemeye ait matematiksel adımları.**

**2.3. Performans Metrikleri**

Bir sınıflandırma probleminde modelin performansını değerlendirmek için sonuçların ne ölçüde doğru ya da yanlış olduğunu belirleyen DP (Doğru Pozitif), YP (Yanlış Pozitif), DN (Doğru Negatif) ve YN (Yanlış Negatif) gibi kavramların açıklanması gerekmektedir. Bu kavramlar, model performansının nicel olarak değerlendirilmesi için temel göstergeler olarak kullanılmaktadır. TP değeri modelin ilgili sınıfa ait veri örneklerini doğru tanımlama kapasitesini yansıtırken, FP değeri modelin sınıfa ait olmayan veri örneklerini yanlış sınıflandırma eğilimini gösterir. TN değeri modelin sınıfa ait olmayan veri örneklerini doğru bir şekilde ayırt etme yeteneğini yansıtırken, FN değeri modelin sınıfa ait olan veri örneklerini gözden kaçırma eğilimini ortaya koyar. Bu dört temel kavram, modelin performansını ölçmek için kullanılan performans metriklerinin hesaplanmasında temel taşlar olarak kullanılmaktadır [9], [10].

Bu çalışmada, modelin performansını değerlendirmek için genel doğruluk, kesinlik, duyarlılık ve F1-skoru gibi metrikler kullanılmıştır. Genel doğruluk modelin genel tahmin başarısını yansıtırken, hassasiyet ve duyarlılık özellikle pozitif sınıf için tahminlerin doğruluğunu ve kapsamını değerlendirmektedir. F1-skoru, duyarlılık ve hassasiyetin harmonik ortalamasını alarak modelin performansını dengeli bir şekilde ölçer. Bu metrikler, modelin performansını çok boyutlu bir perspektiften analiz etmeyi mümkün kılar. Bu performans ölçütleri Denklem 8-11'de verilen matematiksel ifadelerle tanımlanmaktadır.

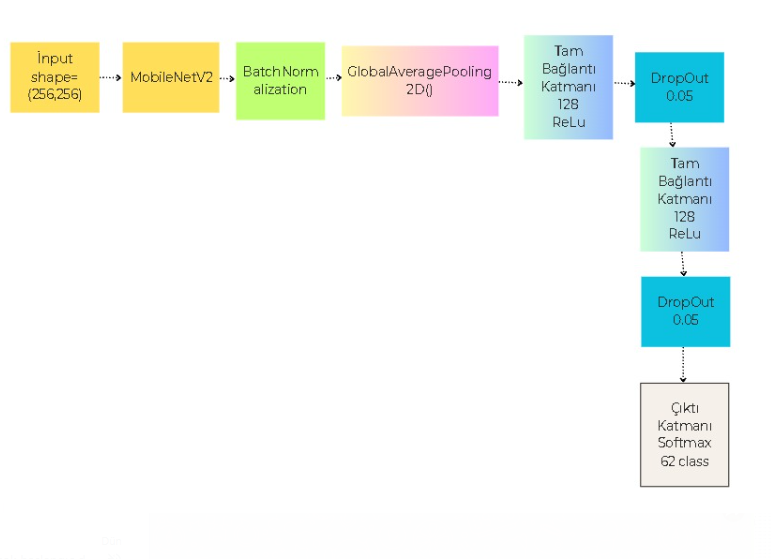
(8)

(9)

(10)

(11)

**3. Deneysel Bulgular**

Bu bölümde, "Handwritten English Characters and Digits" veri seti üzerinde el yazısı karakter sınıflandırması için sıfırdan oluşturulan ve önceden eğitilmiş MobileNetV2 modeli ile desteklenen CNN modelinin bulguları ayrıntılı olarak sunulmaktadır. Ayrıca, bu deneysel bulgulardan türetilen performans metrikleri ve modelin performansının kapsamlı bir analizi sağlanacaktır.

**Şekil 6. Çalışmada kullanılan mimari.**

**3.1. Deneysel Kurulumlar**

Bu çalışmada, görüntü veri kümesi eğitim için %80, doğrulama ve test için %10'ar oranında olmak üzere üç kategoriye ayrılmıştır. Eğitim alt kümesi, oluşturulan CNN modelini eğitmek için kullanılmıştır. Çalışmada kullanılan modelin mimarisi ve katmanları Şekil 10'da gösterilmektedir.  
  
Model, 256x256 piksel boyutunda 3 kanallı (RGB) giriş görüntülerini kabul etmektedir. Mimari, bir MobileNetV2 taban modeliyle başlamakta, ardından Batch Normalization ve GlobalAveragePooling2D katmanları gelmektedir. Bu katmanları takiben, iki adet Tam Bağlantı Katmanı (Dense) bulunmaktadır; her biri 128 nörona ve ReLU aktivasyon fonksiyonuna sahiptir. Bu katmanların arasına ve sonrasında %0.05 oranında Dropout katmanları eklenerek aşırı öğrenmenin önüne geçilmeye çalışılmıştır. Son katman ise 62 sınıfı temsil eden ve Softmax aktivasyon fonksiyonuna sahip bir Çıkış Katmanı'dır.

Çalışma, Google Colab ortamında yürütülmüştür. Çalışmanın gerektirdiği hesaplama gücünü karşılamak için ortamda Intel Xeon CPU (iki sanal CPU) ve 13 GB RAM kullanılmıştır ve tüm işlemler Python programlama diliyle gerçekleştirilmiştir.

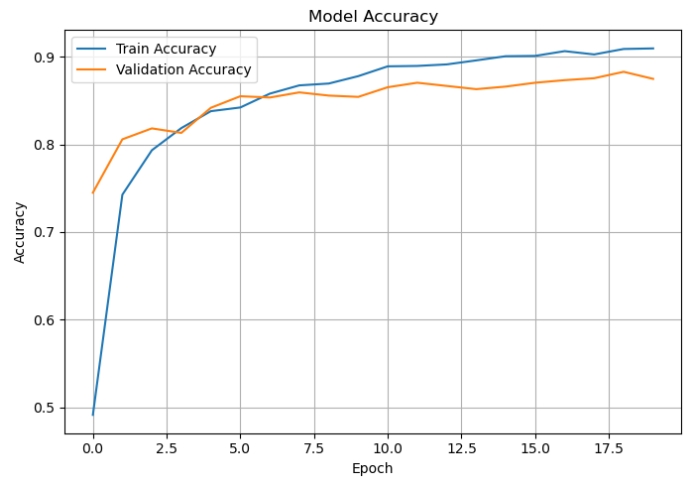
|  |  |
| --- | --- |
| **Hiperparametre** | **Değer** |
| Öğrenme Hızı | 0.001 |
| Kayıp Fonksiyonu | Binary-Cross-Entropy |
| Aktivasyon Fonksiyonu | ReLU, Sigmoid(Son Katman) |
| Tur Sayısı | 20 |

**Tablo 2. CNN modelinin hiperparametreleri.**

**3.2.Bulgular**

Bu bölümde CNN modelinin performansını inceleyeceğiz. Bu inceleme eğitim ve doğrulama kayıpları, eğitim ve doğrulama doğrulukları, sınıflandırma raporları ve karışıklık matrisleri üzerinde yapılacaktır. CNN modelinin doğruluk, duyarlılık, hassasiyet ve F1 puanı sonuçları toplu olarak incelenecektir.

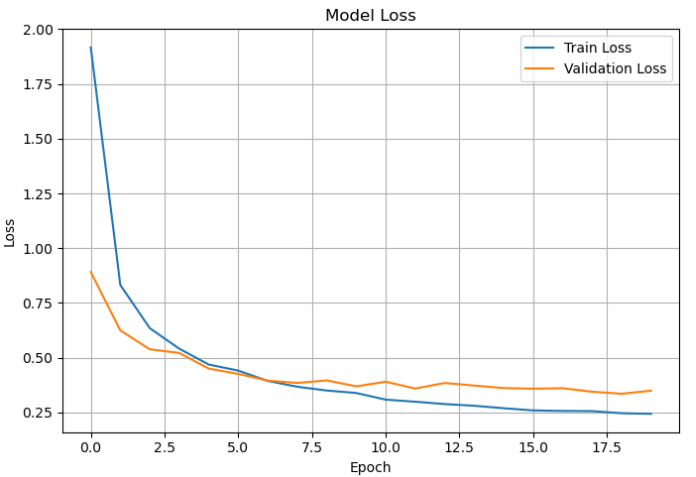
Şekil 7'deki doğruluk grafiği, modelin %80 ile %90 arasında bir doğruluk seviyesine yakınsadığını gösteriyor. Bu, yalnızca başarılı bir öğrenme sürecini değil, aynı zamanda hem eğitim hem de görülmeyen doğrulama veri kümelerinde tutarlı bir şekilde performans gösteren bir modeli de gösteriyor.



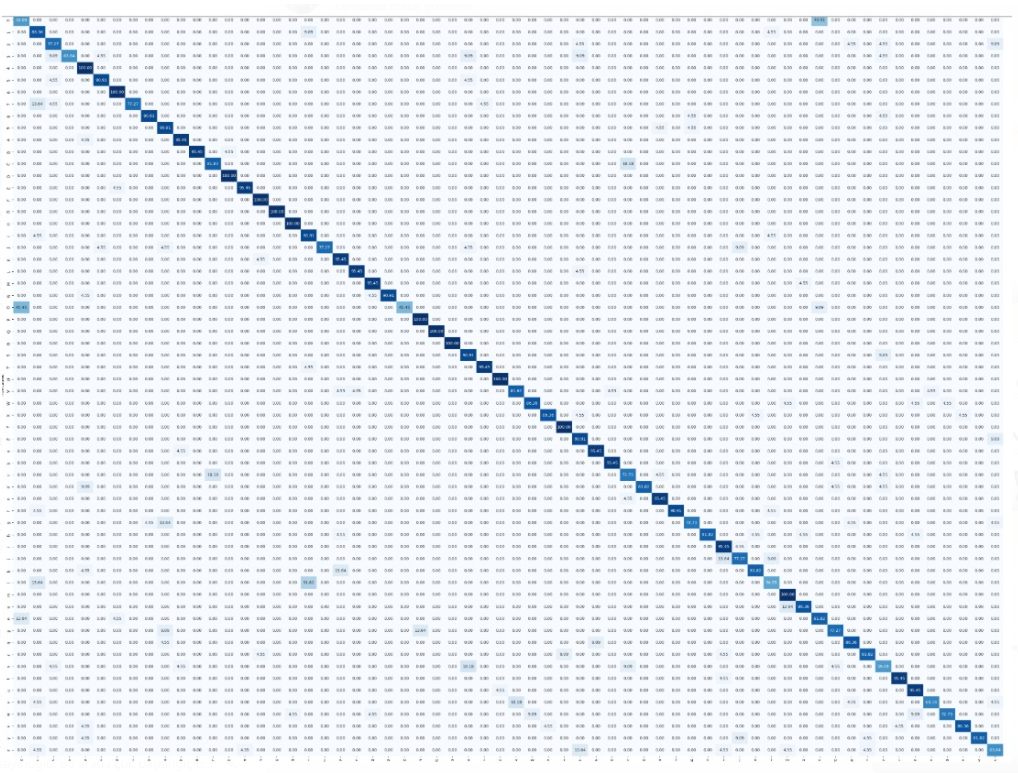
**Şekil 7. CNN modelinin eğitim ve doğrulama doğruluk grafiği.**

Şekil 8'de, kayıp grafiği doğruluk grafiğini tamamlar niteliktedir ve ilk dönemlerde önemli bir düşüşü gösterir, bu da eğitim ve doğrulama setlerindeki hatayı etkili bir şekilde en aza indirmeyi öğrenmiş bir modelin göstergesidir.

Şekil 9, modelin test veri kümesindeki sınıflandırma performansını tanımlayan karmaşıklık matrisini göstermektedir. Modelin genel doğruluğu, diyagonal hücrelerdeki toplam değerlerin (doğru sınıflandırmalar) tüm matris elemanlarının toplamına bölünmesiyle hesaplanabilir. Özellikle tişört, pantolon, spor ayakkabı, çanta ve bot gibi sınıfları için yüksek doğruluk oranları gözlemlenmiştir. Bu sınıflarda, model ilgili sınıf örneklerini başarıyla belirlemiş ve diğer sınıflarla karışıklık oranını düşük tutmuştur. Bu gösterim, sınıflandırma ortamında gezinirken modelin doğruluğunu vurgulamaktadır. Tablo 3'te karmaşıklık matrisinden türetilen performans metriklerinin değerleri bulunmaktadır.



**Şekil 8. CNN modelinin eğitim ve doğrulama kayıp grafiği.**



**Şekil 9. CNN modelinin test veri seti üzerinde değerlendirilmesinden ortaya çıkan karmaşıklık matrisi.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Kategori** | **Kesinlik** | **Duyarlılık** | **F1-Skor** | **Genel Doğruluk** |
| Rakamlar(0-9) | 0.92 | 0.83 | 0.86 | % 88.6 |
| Büyük Harfler(A-Z) | 0.88 | 0.82 | 0.75 |
| Küçük Harfler(A-Z) | 0.89 | 0.73 | 0.66 |

**4. Tartışma ve Sonuçlar**

Bu çalışma, Evrişimsel Sinir Ağlarının (CNN) el yazısı karakter tanıma problemine uygulanabilirliğini ve etkinliğini detaylı bir şekilde incelemeyi amaçlamıştır. Geliştirilen CNN modeli, Kaggle platformunda bulunan "Handwritten English Characters and Digits" veri setini kullanarak, 26 büyük harf, 26 küçük harf ve 10 rakam olmak üzere toplam 62 farklı sınıfı dijital ortama aktarma yeteneğini sergilemiştir.

Yapılan deneyler sonucunda, model genel olarak **%88.6 doğruluk oranı** ile başarılı bir performans elde etmiştir. Bu başarı, CNN'lerin görüntü sınıflandırma ve özellikle el yazısı tanıma gibi karmaşık görevlerdeki güçlü potansiyelini bir kez daha ortaya koymaktadır. Elde edilen yüksek doğruluk oranı, modelin farklı el yazısı stillerini ve karakter varyasyonlarını başarılı bir şekilde öğrenme ve ayırt etme kapasitesini göstermektedir.

Ancak, her modelde olduğu gibi, bu çalışmada geliştirilen modelin de bazı sınırlılıkları bulunmaktadır. Bu tür karışıklıklar, veri setindeki bazı sınıfların benzerliği, veri miktarındaki dengesizlikler veya model mimarisinin bu detayları tam olarak öğrenmekte zorlanması gibi faktörlerden kaynaklanabilir.

Gelecekteki çalışmalarda, modelin performansını daha da artırmak ve tespit edilen sınırlılıkları gidermek için çeşitli iyileştirmeler önerilmektedir:

* **Veri Artırma (Data Augmentation):** El yazısı karakterlerindeki varyasyonları artırmak ve modelin genelleme yeteneğini geliştirmek için veri artırma teknikleri (döndürme, ölçekleme, kaydırma vb.) uygulanabilir.
* **Hiperparametre Optimizasyonu:** Öğrenme hızı, epoch sayısı, batch boyutu ve optimizasyon algoritması gibi hiperparametrelerin daha kapsamlı bir şekilde optimize edilmesi, modelin performansını potansiyel olarak artırabilir. Özellikle daha gelişmiş optimizasyon algoritmaları (Adamax, Nadam vb.) veya öğrenme hızı çizelgeleri (learning rate schedules) denenebilir.
* **Daha Derin veya Farklı Mimari Keşifleri:** MobileNetV2 tabanlı mimarinin yanı sıra, ResNet, VGG veya Inception gibi farklı önceden eğitilmiş CNN modelleri veya daha derin ve karmaşık mimariler, daha iyi özellik çıkarımı sağlayarak performansı artırabilir.
* **Hata Analizi ve Düzeltme:** Yanlış sınıflandırılan örneklerin detaylı bir analizi yapılarak, modelin hangi tür hataları daha sık yaptığı belirlenebilir ve bu hatalara özel çözümler geliştirilebilir. Örneğin, belirli karakterler için özel özellik çıkarıcılar eklenebilir.
* **Transfer Öğrenme Yaklaşımlarının Derinleştirilmesi:** Önceden eğitilmiş modellerin sadece taban katmanlarının değil, daha fazla katmanın eğitilebilir hale getirilmesi (fine-tuning) ile modelin veri setine daha iyi adapte olması sağlanabilir.

Sonuç olarak, bu çalışma, el yazısını dijitalleştirmek için CNN'lerin güçlü bir araç olduğunu göstermiştir. Elde edilen %88.6'lık doğruluk oranı, bu alandaki mevcut başarıları pekiştirmektedir. Gelecekteki araştırmalar, yukarıda belirtilen iyileştirme alanlarına odaklanarak modelin daha da sağlam ve doğru hale getirilmesine katkıda bulunacaktır.

**KAYNAKLAR**

[1] Kaggle. (t.y.). Handwritten English Characters and Digits. Erişim adresi: <https://www.kaggle.com/datasets/sujaymann/handwritten-english-characters-and-digits/data>

[2] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84-90. (Bu, derin öğrenme alanındaki temel çalışmalardan biridir.)

[3] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521, 436-444. (Genel derin öğrenme referansı)

[4] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint* arXiv:1704.04861. (MobileNetV2'nin temelini oluşturan MobileNet'in referansı)

[5] Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning*. (ReLU aktivasyon fonksiyonuna dair temel referans)

[6] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*. (VGG ağları ve derin CNN'ler hakkında genel bilgi için kullanılabilir.)

[7] Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, *12*(Jul), 2121-2159. (Adam optimizasyon algoritmasının temelini oluşturan adaptif öğrenme oranlarına dair referans)

[8] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *60*(2), 91-110. (

[9] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.

[10] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer* Vision and Pattern Recognition (pp. 770-778).

[11] Russell, S. J., & Norvig, P. (2010). Artificial Intelligence: A Modern Approach. Pearson Education.