Proje ana alanı: Yazılım

Proje tematik alanı: Yapay zeka

Proje adı: Ön çapraz bağ yırtıklarının tespitinde farklı derin öğrenme yöntemlerinin karşılaştırılması

Özet

Ön çapraz bağ (ÖÇB) yırtığı, günümüzde özellikle sporcularda ortaya çıkan ve uzun soluklu bir tedavisi olan bir sakatlıktır. Bu yırtıklar atlatması oldukça uzun sürebilecek düzeylere ulaşabilir ve yırtılma anından sonraki süreçte sakatlanan kişinin ön çapraz bağının yırtığının kısa sürede tespit edilmesi, gerekiyorsa ameliyat olması ve hızlıca tedavi sürecine başlaması önemlidir. Bu yırtığın tespiti için MRG (manyetik rezonans görüntüleme) makineleri kullanılır. Bu görüntüler çekildikten sonra doktorlar uzun süreler boyunca hastanın ne derece yırtığı olduğunu bulmaya çalışır ve yanlış tanıyı koyma ihtimali de ortaya çıkar. Bu sürecin olabildiğince kısalması için son senelerde oldukça gelişen derin öğrenme teknolojisi ile ESA (evrişimsel sinir ağları) kullanılarak tanılar otomatik tespit edilebilir. Modelin sınıflandıracağı veriler "Kaggle" sitesinden iki sınıfa ayrılmış Python 'pickle' objelerinden seçilmiştir. Bu veriler standart bir kalıba sokulduktan sonra çeşitli konvolüsyon ve küçültme işlemleri uygulanmıştır. Verilerin sınfılandırılması sonucu doğruluk oranları ve hata payları hesaplanmıştır. Doğruluk oranı eğitimde 0.969'a test setinde ise 0.858'e kadar ulaşmıştır. Ayrıca test setindeki anormallik tespitindeki doğruluk oranı %94 olarak hesaplanmıştır. Bu oranlar derin öğrenme ve tıp alanındaki gelişmeler için umut vericidir.

Anahtar Kelimeler: ön çapraz bağ, derin öğrenme, doğruluk oranı, evrişimsel sinir ağları

Amaç

Ön çapraz bağ, dizin merkezinde ve dıştan içe doğru, arkadan öne doğru uzanan ön çapraz bağ, dizin dönme hareketini kısıtlayan en önemli bağdır. Uyluk ve kaval kemiklerini bağlayan ligamenttir. Hafif yırtıklı ÖÇB'lerin kendi kendine iyileşmesi mümkünken, tam yırtık olanlarda ameliyat gerekmektedir (1). Bu sakatlığa maruz kalan hastaların yırtığının hangi kategoriye ait olduğunun görülmesi, ameliyat yapma kararını etkilediği için oldukça önemlidir. Derin öğrenme yardımıyla son yıllarda MR görüntülemeyle ilgili çeşitli çalışmalar yapılmıştır. Bu çalışmayla sağlıklı, yırtık ve kopuk ÖÇB'ler derin öğrenme modeliyle sınıflandırılarak, doktorların koyduğu tanılara yardımcı olunmaya çalışılacaktır.

- Derin öğrenme algoritması kullarak ÖÇB yırtıklarının tespitinde doğruluk oranlarını araştırmak.
- Son olarak MRG'lerde Farklı epoch ve kernel boyutlarında modelin doğruluk oranının nasıl değiştiği de gözlemlenecektir.
- Eğitilen modelde en yüksek bir doğruluk oranı elde etmek. Yüksek doğruluk oranları özellikle eğitim sürecinde olan radyologların, veya radyoloji dışı uzmanlığı olan doktorların tanı doğru koymasına imkan vermek amaçlanmıştır.

Giriş

Ön çapraz bağ (ÖÇB) yırtığı, ortopedik alandaki ciddi sakatlanmalardan biridir. Ergenlik veya gençlik çağındaki sporcu sakatlanmaları arasında önemli bir yeri bulunmaktadır. Ayrıca erkekler, kadınlara oranla daha fazla bu sakatlığa maruz kalmaktadır (2). Bu yaralanmaların zamanında ve doğru tespiti, ardından alınacak tedbirler ve rehabilitasyon sürecinin uzamaması açısından oldukça önemlidir. Bu bağlamda, medikal görüntüleme teknolojilerindeki ilerlemeler, sakatlığın teşhis dönemine yeni bir boyut kazandırmıştır. MRG (manyetik rezonans görüntüleme) bu projede dizleri görüntülemek için kullandığımız görüntüleme tekniğidir. Manyetik Rezonans Görüntüleme (MRG), vücutta ayrıntılı resimler oluşturmak için güçlü mıknatıslar, radyo dalgaları ve bilgisayarın kullanıldığı bir testtir (3).

ÖÇB yırtıklarında kas-iskelet radyologları için tanı yüksek doğruluk oranları ile yapılabilmekte iken uzman olmayan veya eğitim alan radyologlar veya ortopedi, fizik tedavi gibi radyoloji dışı uzmanlık alanları olanlarda tanı daha zor olabilmektedir. Özellikle kısmi yırtıklarda doğruluk oranları tam kat yırtıklara göre uzman radyologlar için tanı güçlüğü yaşanabilmektedir. Ayrıca özellikle kısmi yırtıklarda ortopedistlerin kullandığı tanısal muayene teknikleri bile bazen yanıltıcı olabilmektedir. Özellikle bu gibi durumlarda kullanılmak üzere, son yıllarda başka birçok alanda da kullanılmaya başlayan derin öğrenme teknolojisi ile ESA (evrişimsel) kullanılarak tanıya yardımcı olamak mümkündür.

Derin öğrenme, yapay sinir ağlarının büyük miktarda veriden öğrendiği makine öğrenmesinin bir alt kümesidir (4). Yapay zeka, insanın düşünmesini simüle etmeyi çalışarak bilgiyi işleyebilen insan kontrolünü gerektirmeyen yeni ve daha zeki bir makine oluşturmaya çalışır (5). Derin öğrenme modeli çok katmanlı sinir ağlarından oluşur. Bu modeler genelde görüntü işlerken verileri anlamlandırmak için kullanılır. Özellikle tıbbi görüntüler ayrıntılı özellikler içerir, tanısal yoğunluk, kenar ve şekil ayrımları dahil anlam buradan çıkarılabilir (6).

Bu görüntüler veri setine dahil edilecek ve geliştireceğimiz derin öğrenme modelinin bu görüntülere tanı koymasının öğretilmesi amacıyla çalıştırılacaktır. Veriden gelen bilgiler girdi katmanından geçip oradan ara katmanlara, o da önceki katmandan gelen girdiler üzerinde basit işlemler gerçekleştirerek bir kendi içinde bir sonraki katmana iletir. Bu işlem sınıflandırma katmanına kadar devam eder (7). Sınıflandırma katmanında bir çıktı oluşturulur ve bu çıktı problemin ne olduğuna göre değişir. Sınıflandırma problemi, bir veri setindeki elemanın özelliklerine göre hangi sınıfa ait olduğunu belirleyen probleme denir. Örneğin, akciğer rahatsızlığı olan birinin çekilen MRG sonucu kanser olup olmadığı bir ikili sınıflandırma problemidir çünkü sınıfladırılabilecek sadece 2 sınıf vardır (8).

MRG görüntülerinin kullanılıp ön çapraz bağ veya menisküs yırtığı tespitini yapan projeler bulunmaktadır. Örneğin, Nicholas Bien, Rajpurkar Pranav ve diğer çalışma arkadaşlarıyla Johns Hopkins Üniversitesinde yapılan bir çalışmada, derin öğrenmeyle eğitilen ve ön çapraz bağ yırtığını tespit eden modelde %86.7'lik bir doğruluk elde edilmiştir (9). Fang Liu'nun çalışma arkadaşlarıyla gerçekleştirdiği projede ise, oluşturdukları derin öğrenme modeli ise çok daha başarılı bir şekilde %98'lik bir doğruluk oranı elde edilmiştir (10). Yukarıdaki gibi

ÖÇB veya menisküs gibi birçok diz sakatlığını tespit edebilen çalışmalar literatürde bulunmaktadır.

Bu çalışmada kernel boyutunun ve epoch sayılarının modelin başarı oranlarıyla doğru orantılı olup olmadığı ve başarı oranını nasıl etkilediğine bakılacaktır.

Yöntem

Veri Seti

Eğitilecek olan model her bir diz görüntüsünü tanınttığımız 3 farklı sınıfa ayıracaktır. Hırvatistan'daki Klinik Rijeka Merkez Hastanesi 2006 ve 2014 yılları arasında çekilen bazı diz MRG görüntüleri kullanılacak olan veri setidir. MRG görüntülerinden hazırlanmış olan yaklaşık 3GB'lık ve 917 vaka sayılı veri setindeki resimler bu modeli eğitebilmemiz için kullanılacaktır (11-12).

Veri Sınıflandırma

ÖÇB yırtıkları kısmi ver tam kat yırtık olarak sınıflandırılmaktadır. Tanının doğru konulabilmesi için de vakaların hangi sınıfa ait olduklarını belirlemek büyük önem taşır. ÖÇB kullanılan veri setinde 3 gruba ayrılmıştır:

1) Sağlıklı
2) Yarım Yırtık
3) Tamamen kopuk ÖÇB

1. Sağlıklı ÖÇB

Ayrıştıracağımız üç sınıftan birinde sağlıklı diz örnekleri bulunacaktır. Ön çapraz bağında herhangi bir anormallik yoktur (Şekil-1).

Şekil 1: *Sağlıklı ÖÇB görüntüsü*

2. Kısmi (parsiyal) Yırtık ÖÇB

İkinci derece yırtığa sahip vakalarda, ÖÇB'de bir kopukluk görülür ancak bu kopukluk bütün ligament için geçerli değildir. Sadece belirgin bir yırtık vardır.

Şekil-2: *Kısmi yırtıklı ÖÇB*

3. Kopuk (Tam kat) ÖÇB:

Liflerin tamamı kopmuştur ve tedavisi 9 aya kadar çıkabilir (13) (Şekil-3).

Şekil-3:*Kopuk ÖÇB yırtığı görüntüsü*

Verinin Hazırlanışı

Derin öğrenmeyle bir model geliştirilebilmesi ve eldeki veri setini üzerinde işlemlerin gerçekleştirilebilmesi için "pickle" formatındaki Python objelerinden fotoğraflarla ilgili veriler çıkartılıp, aynı büyüklükteki matrislere çevrilmesi gerekir. Bu şekilde modele gönderilecek olan her bir verinin aynı formatta olması sağlanır. Yapay zeka modellerinin eğitiminde bütün veri seti 3 bölüme ayrılır:

Birinci set ***'eğitim seti'*** olup genellikle eğitilen modellerde toplam veri setinin %70'ini oluşturur. İkinci set ***'validasyon seti'***dir ve bu sette de genellikle toplam verinin %20'si ayrılır. Son olarak ***'test seti'***nde ise bütün veri setinin %10'u yer alır. Bu projede eğitim, validasyon ve test setleri, sırası ile %80, %10 ve %10'luk oranlarla ayrıştırılacaktır (Şekil-4) (14).

```
X_train, X_temp, y_train, y_temp = train_test_split(img_list, Y, test_size=0.2, random_state=42)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.5, random_state=42)
```

Şekil-4: *Veri setlerinin oranlaması*

Verilere ait bilgileri almak ve sonrasında işlemek için bir csv tablosunda kayıtlı olan veriler bir değişkene kaydedilmiştir. Sonrasında bir döngüyle bu veri çerçevesi üzerinde gezilerek dosyanın ismini barındıran sütuna bakılmıştır. İsimler alınıp yeni bir sütun açılarak bu sütunlara dosyaların bilgisayar üzerinde hangi dizinde bulunduğu kaydedilmiştir (Şekil-5).

```
base_path = "C:/Users/borae/Documents/Coding Files/Anaconda Python/JUPYTER/TUBITAK PROJE/knee_mri_dataset"
images_found = []
df = pd.read_csv(f"{base_path}/metadata.csv")
df['path'] = "Image not found"


for index, MRI in tqdm(enumerate(df['volumeFilename'])):
    file_found_path = None  #dosyaların bulunduğu dizinin yazılması için yeni bir sütun('volumeFilename')

    #dosyaların bulunduğu klasörden her birinin dizinlerinin çıkartılması
    try:
        file_found_path = glob.glob(base_path + "/datas/" + MRI)[0]
        df['path'].iloc[index] = file_found_path


    except:

        df['path'].iloc[index] = "Image Not Here"
        pass
```

Şekil-5: *verilerin dizinlerinin bulunması ve kaydedilmesi*

Sonrasında sağlıklı olan ÖÇB vakalarını bir tabloda, yırtık olanları ise ayrı bir tabloda toplanır. Metadata tablosunda veriler 0, 1 ve 2 şeklinde sınıflandırılmıştır. Sağlıklı ÖÇB örneği 690, yırtık 172 ve kopuk ÖÇB'de ise 55 tane örnek bulunmaktadır. Kısacası üçlü sınıflandırma ile ÖÇB yırtıkları ayrıştırılıp farklı veri çerçevelerine konulacaktır. Sonundaysa sağlıklı, yırtık ve kopuk ÖÇB için oluşturulan veri çerçeveleri birleştirilecektir (Şekil-6).

```
#sağlıklı dizlerin olduğu bir dataframe
new_df0=df[df.aclDiagnosis==0]
#yırtık ÖÇB'ler new_df1 adlı veri çerçevesine
new_df1=df[df.aclDiagnosis==1]
#kopuk ÖÇB'ler new_df2 adlı veri çerçevesine
new_df2 = df[df.aclDiagnosis==2]
#üçünün birlikte olduğu yeni bir veri çerçevesi
frames = [new_df2,new_df1,new_df0]
new_df = pd.concat(frames)
```

Şekil-6: *Farklı etiketli verilerin kendilerine ait veri çerçevelerine eklenip sonra bir çerçevede toplanması*

Hazırda kaydedilen dizinlerdeki verileri işler. Veri pickle objesi formatında olsa da, Python'da 'pickle.load()' komutu kullanılarak verideki çekilmiş spesifik bir kesit alınır. Daha sonra bu kesit metadata önceden hazılanmış olan'roiX','roiY','roiWidth' yani genişlik ve 'roiHeight' yani yükseklik değişkenleri girilerek belirli bir kesit çıkarılır. Daha sonra 'image_array' adlı değişkenden ilgili bölgeyi alır. Bu şekilde ÖÇB yırtığının tespitinin en az işlem gücü gerektircek şekilde yapılanması modelin işleyeceği yerin sınırlandırılması ile modelin doğruluk oranının artması sağlanır. Çıkarılan bu spesifik kesit 90x90 çözünürlükte bir resim haline getirildi. Hazırlanan bu kesitler bir resim listesine eklendi. İşlenen resimler için yırtığın olup olmadığını belirten etiketler (0,1 veya 2) bir 'Y' adlı listede toplandı(Şekil-7).

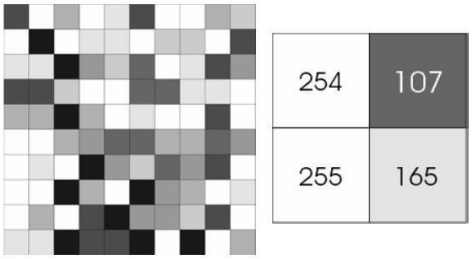
```
Y = []
for i in range(len(new_df)):

    #'pickle' dosyasının bulunması halinde verinin üzerinde bir kesit alma işlemi gerçekleştiriliyor.
    if df['path'].iloc[i] != "Image Not Here":
        try:

            with open(new_df['path'].iloc[i], 'rb') as file_handler:
                image_array = pickle.load(file_handler)
            img=image_array[new_df['roiZ'].iloc[i], :, :]
            x=new_df["roiX"].iloc[i]
            y=new_df["roiY"].iloc[i]
            w=new_df["roiWidth"].iloc[i]
            h=new_df["roiHeight"].iloc[i]
            image_array=img[y:y+h, x:x+w]
            #modelimize girecek her verinin aynı formatta olması lazım
            image8_array = resize(image_array, (90, 90))
            image_list.append(image8_array)
            Y.append(new_df["oclDiagnosis"].iloc[i])
        except:
            pass

img_list=np.asarray(image_list)


Y=np.asarray(Y)
```

Şekil-7: *Verilerin MR görüntülerinden ÖÇB'ye ait kesitin alınıp resim listesine konması ve gerçek etiketlerin Y adlı listeye doldurulması.*

Modelin Hazırlanışı ve Yapısı

- **Girdi Katmanı**

ESA (evrişimsel sinir ağı)'nın ilk katmanıdır. Modelimizin resim halindeki veriyi ham olarak aldığı katmandır, herhangi bir işlemden geçirilmemiştir. Resim çözünürlüğü ve boyutu, modelin daha isabetli tanılar koyması için önem taşır. Daha yüksek çözünürlükteki bir resim çok daha detaylı görüntü oluşturabildiği için modelin doğruluk oranının yüksek olmasında katkı sağlar. Ancak, resimin kalitesi ne kadar artarsa modeli eğiten makinenin oldukça büyük bir işlem gücü kapasitesini taşıyabiliyor olması gerekir. Elimizde bulunan veri setindeki görüntüler MRG görüntüsü olduğundan dolayı RGB değil 'grayscale' veya gri tonlu olarak adlandırılan, tek katmandan oluşan resimlerdir. Bu projedeki resimlerin her bytei 12 bitten oluşur. 1 bit ise 0 veya 1 değerlerini alır. 0 siyaha, 1 ise beyaz rengi temsil eder. 1 piksel 12 bitten oluştuğundan ve her piksel 0 veya 1 değerini alabildiğinden her piksel 2^{12} =4096 farklı değer alabilir ([0,4095]). 0'dan 4095'e giderken renk siyahtan beyaza doğru parlaklaşır (15-16)(Şekil-8)

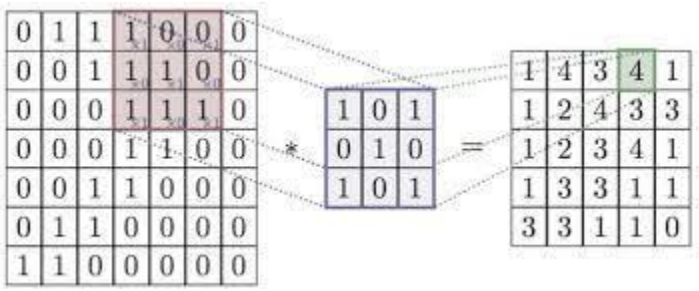


Şekil-8: *8 bitlik Gri ölçekli bir resmin sayılarla ifade edilişi (17)*

- **Konvolüsyon Katmanı**

Konvolüsyon, görüntü işlemede önemli bir rolü olan bir işlemdir. Konvolüsyon belirli sayılardan oluşan matrislerin görüntü üzerinde kaydırılma işlemidir (15). Görüntü üzerinde oynatılan matrislere filtre denir ve genellikle 2x2, 3x3 ve 5x5lik boyutlarda olurlar. Konvolüsyon işlemi görüntü işlemede pek çok amaçta kullanılabilir. Örneğin; kenar sınır piksellerinin belirlenmesi için konvolüsyon kullanılır. Renk geçişinin çok belirgin olduğu yerlere kenar sınır pikseli denir. Veya görüntüyü bulanıklaştırmaya da yarar. Filtrelerdeki matrislere 0-1 arası sayılar konduğunda filtrelerin uygulandığı değerlerdeki sayılar birbirine yaklaşırlar ve görüntünün bulanıklaşmasını sağlarlar. Filtreler görüntü üzerinde kaydırılırken filtrenin üzerindeki ve resimdeki pikseli temsil eden sayılar çarpılırlar ve çarpımların toplamı 'aktivasyon haritası' olarak adlandırılan yeni bir matrisin içine koyulur (Şekil-9).

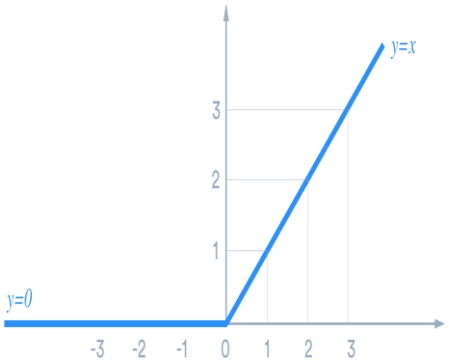
Bu projede filtre boyutları (2x2, 3x3 ve 5x5) 3 farklı şekilde uygulanmıştır ve her farklı filtreli model için doğruluk oranları karşılaştırılacaktır.

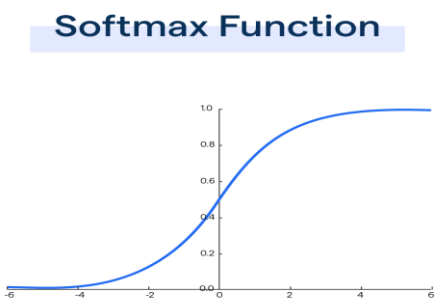


Şekil-9: *Konvolüsyon İşlemi ve yeni değerlerle oluşan aktivasyon haritası (18)*

- **Aktivasyon Katmanı**

Aktivasyon katmanında çeşitli aktivasyon fonksiyonları kullanılır. Aktivasyon fonksiyonları genelde doğrusal olmayan öğrenme problemlerini çözebilmek için kullanılır. Böylece daha karmaşık bir sinir ağı oluştururken modelin öğrenme yeteneğini geliştirirler. Yani problemlere daha esnek bir şekilde uyum sağlayabilir. Ancak modelimizde kullanacağımız 'ReLU' (Şekil-10) ve 'Softmax'(Şekil-11) adlı fonksiyonlardan ReLU, çoğu aktivasyon fonksiyonunun aksine modele bir lineerlik kazandırır. ReLU, görüntü işlemede yaygın olarak kullanılan bir aktivasyon fonksiyonudur. Lineer bir fonksiyon gibi gözükse de karmaşık ve derin ağları eğitebilir. Hesaplama açısından zaman kazandırır ve hesaplaması oldukça basittir. Hesaplamasının basit olması derin sinir ağının daha hızlı öğrenmesine yol açar. ReLU, softmax gibi bir fonksiyona göre daha az eğim kaybına sebep olur. Bu 'yok olan gradyan meselesi' olarak adlandırılan sinir ağlarının derinleştikçe ortaya çıkardığı problemin hafiflemesine yol açar. ReLU'da 'ölü nöron' denilen bir dezavantajı vardır. ReLU konvolüsyon uygulandıktan sonra ortaya çıkan değerlerden 0'dan küçük her değeri 0 olarak aldığından veya eğer bir sinir hücresi eğitim boyunca eksi bir ağırlığa sahip bir girişle çok küçük bir değeri çarptığında, çıkışı sıfır olabilir ve o sinir hücresi eğitim süresince bir şey öğrenmez (19).





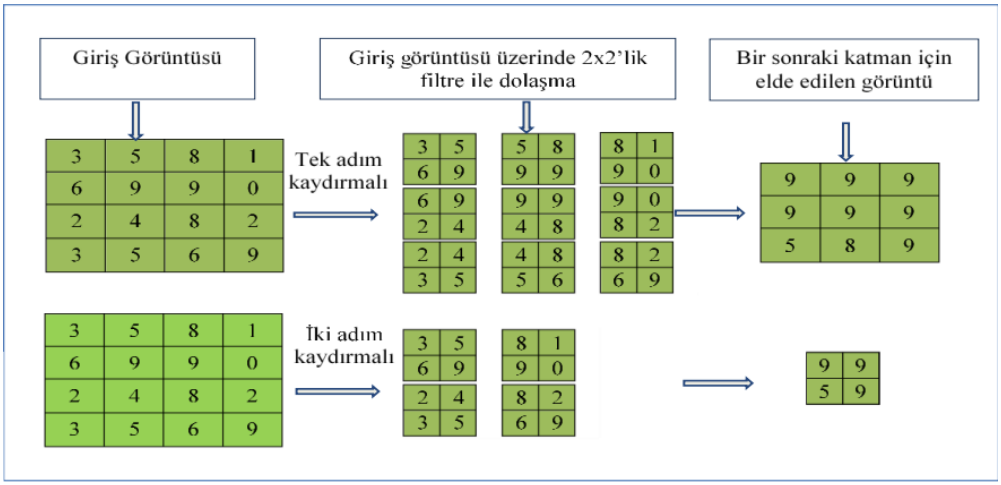
Şekil-10: *ReLU grafiği (20)*

Şekil-11: *Softmax grafiği (21)*

- **Havuzlama Katmanı**

Çoğunlukla aktivasyon katmanından sonra yerleştirilir. Amacı sonraki konvolüsyon işlemi için girdi matrisinin boyutunu küçültmektir (15). Matrisin boyutunu küçültmek bilgi kaybına yol açabilir. Ancak bir sonraki ağ katmanları için daha az işlem yükü olmasını sağlaması, modelin eğitilirken öğrendiği örnekleri ezberlemesini önleyerek(overfitting), farklı veri setlerinde de üzerine çalıştığımız veri setine benzer doğruluk oranlarının oluşmasına neden olur. Konvolüsyon katmanında da olduğu gibi bu katmanda da boyutun küçülmesi için çeşitli filtreler uygulanır. Bu filtreler genellikle ikiye ayrılır: Maksimum havuzlama filtre içinde bulunan değerler arasındaki en büyüğünü alır. Ortalama havuzlama ise filtre içindeki sayıların ortalamasını alarak yeni bir sayı oluşturup yeni matrise koyar (Şekil-12). Maksimum havuzlama daha iyi performans gösterdiği için genellikle tercih edilir. Bu katman bir ESA'da zorunlu değildir ancak yüksek çözünürlükteki görüntülerde ve bilgisayar donanımı yeterince güçlü olmayan kişiler işlem yükünü hafifletmek için genelde tercih eder.

Çalışmada bütün modellerde 2x2'lik max pooling katmanı uygulanmıştır.



Şekil-12: *Maksimum havuzlamanın 2x2'lik filtrelerle bir örneği (15)*

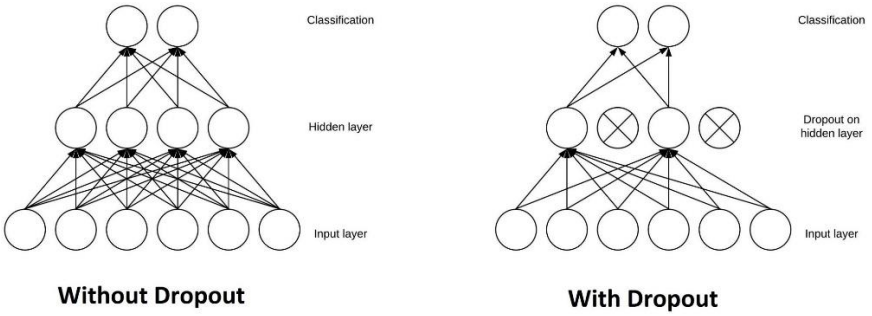
- **Tam Bağlantılı Katman**

Bu katman, konvolüsyon ve havuzlama katmanlarından geçip ortaya çıkan matrisi tek boyutlu bir vektöre dönüştürür. Bu katman ayrıca bu vektörü kullanarak oldukça geniş bir bağlantı ağı kurar ve bunu sınıflandırma ve regresyon gibi problemlerde çıktı üretimi için kullanır (15). Bu katmana girdi olarak giren vektör, nöronlar aracılığıyla "dense" olarak adlandırılan katmanlardan da geçer. Bu katmanlarda ise her bir bağlantının bri ağırlığı vardır. Ağırlık, bir sayıdır ve vektörden gelen sayılarla çarpılıp bir sonraki dense katmanına aktarılır.

- **Dropout Katmanı**

Dropout katmanı, girdilerin bazılarını rastgele bir şekilde 0'a ayarlar. Bu katmanın aslında modelin veri setini ezberlememesi için önemlidir (15). Veri setini ezberlettiğimiz bir model daha önce karşısına çıkmamış vakalarda gözüken doğruluk oranının oldukça altında kalabilir. Bazı verilerin 0 olması, daha genel ve ön planda olan özellikleri saptar ve test setinde yine eğitim setinin çok altında kalmaması sağlanır (Şekil-13)

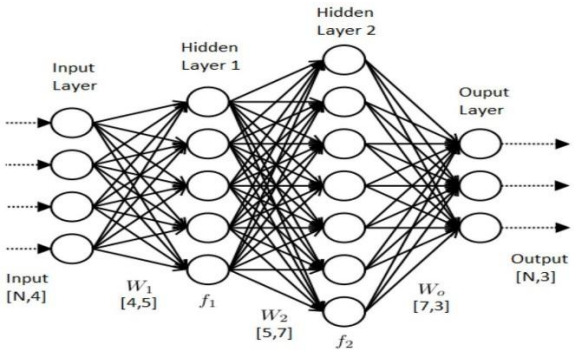
Bu projede Dropout olma olasılığı %30 olarak belirlenmiştir.



Şekil 13: *Dropout katmanının tam bağlantılı (dense) katmanına etkisi (22)*

- **Sınıflandırma Katmanı**

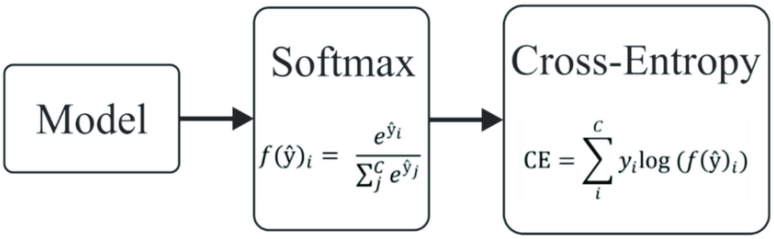
Verinin dense katmanlarından geçmesiyle artık MRG'nin tanısının konacağı katmandır. Vektördeki sayıların dense katmanındaki sayılarla çarpılmasıyla en son bir sayı elde edilir. Bu sayının nasıl sınıflandırılacağı, sınıflandırılması yapılacak olan nesne sayısına bağlıdır. Bizim projemizde 3 farklı sınıfa ayrılacağından bizde 3 olacaktır. Bu katmandaki nöron sayısı sınıf sayısına göre değişir (15). Modelin tahmin etmiş olduğu sınıf ise genellikle en yüksek değere sahip olur (Şekil-14).



Şekil-14: *Tam bağlı katman ve sınıflandırma katmanının bir örneği, matrisin vektöre dönüşümü ve vektördeki değerlerin ağırlıklarla çarpılması (23)*

- **Optimizasyon ve Hata Payı Fonksiyonu**

'Cross-Entropy' fonksiyonları, gerçek değerle modelin tahmin ettiği değer arasındaki farklı hesaplayan logaritmik fonksiyonlardır (24). Geliştirdiğimiz modeldeki hata payını bu fonksiyon hesaplar. Optimizasyon algoritmaları ise nöron ağındaki özellikleri (ağırlıklar veya öğrenme hızı) değiştirerek hata fonksiyonunun değerinin daha düşük çıkmasını amaçlar (25). Bu çalışmada bütün modellerde 'Adam', kayıp fonksiyonu için de birden fazla sınıfla sınıflandırma yapıldığı için 'CategoricalCrossEntropy' fonksiyonları kullanılacaktır(Şekil-15).



Şekil-15*: Hata payı (cross-entropy) fonksiyonunun hesaplanışı (26)*

Konfüzyon Matrisi

Konfüzyon matrisi modelin yaptığı tahmin ve verinin gerçek değeri arasında karşılaştırma yapılabilmesi için kullanılabilecek bir araçtır. Yatay eksende gerçek değerler, dikey eksende ise modelin tahminleri yerleştirilerek hangi verinin ne kadarlık miktarını doğru veya yanlış bulduğu görselleştirilerek öğrenilebilir.

İterasyon

Bir modelin eğitim setinin üzerinden kaç kere geçtiğini ve ağırlıkları güncelleme amacı olan terime 'epoch' denir (27). Fazla epoch modelin doğruluk oranını artırabilirken, aynı zamanda eğitim verisinin ezberlenme riskini barındırır. Bu yüzden de test ve eğitim setleri arasındaki doğruluk oranı farkı artabilir. Bu çalışmada ise 60 ve 90 Epoch'lu modeller eğitilmiştir.

Çalışmadaki ESA Katmanları

1: 20 tane 5x5'lik konvolüsyon katmanında filtre.

2: "Relu" aktivasyon fonksiyonu.

3: MaxPooling2D(pool_size(2,2),strides=(2,2))

Bu döngü 3 kere tekrarlanıp veriler indirgendikten sonra kalan matrise:

4: Flatten(), matrisi vektöre çevirir. Tam bağlantılı katmanda yer alır. Tam bağlantılı katmanda her nöron bir önceki katmandaki nöronların her birinden girdi alır ve bir sonraki katmandaki tüm nöronlara iletir. Flatten() komutu sonucu oluşan vekördeki her bir değer ise tam bağlantılı katmanın ilk katmanındaki nöronlara girdi olarak girer.

5: Dense(64) ilk katmanında 64 nöron bulunduran bir tam bağlantılı katman

6: Activation("relu"), relu aktivasyon fonksiyonunu kullanan bir dense katmanı

7: Dropout(0.3) 3/10 olasılıkla nöronlardaki bir girdiyi 0 olarak ayarlar.

8: Dense(classes), son katmanda 3 çıkış olacak o da sınıflandırmak isteyeceğimiz nöron sayısında olacak ki bu durumda 3 tane oluyor.

9: Activation("softmax") bu katman için kullanılacak aktivasyon fonksiyonu.

İş Zaman Çizelgesi

	AYLAR									
İşin Tanımı	**Nisan**	**Mayıs**	**Haziran**	**Temmuz**	**Ağustos**	**Eylül**	**Ekim**	**Kasım**	**Aralık**	**Ocak**
Literatür Taraması				X	X	X	X	X		
Arazi Çalışması						X	X	X		
Verilerin Toplanması ve Analizi								X	X	X
Proje Raporu Yazımı								X	X	X

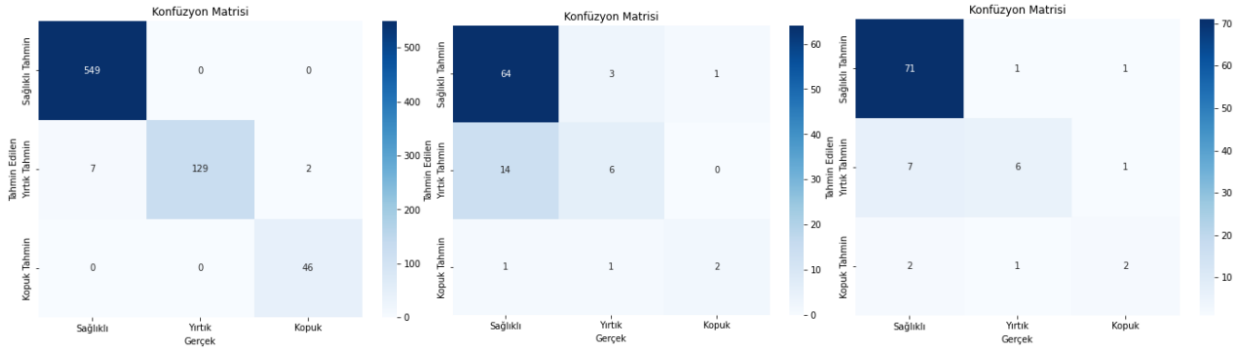
Bulgular

Farklı kernel boyutları (2x2, 3x3, 5x5) ve epoch sayıları (60 ve 90) olmak üzere toplam 6 model için, tüm veri seti, sağlam (0), kısmı yırtıklar (1) ve tam kat yırtıklar (2) için doğruluk ve hata payları hesaplanmıştır (Tablo-4). Buna göre en iyi modelin test setinde‘5x5 filtre, 90 Epoch’ %85.8 doğruluk oranı ve %1.5 hata payı olduğu ortaya çıkmıştır. Bu modele ait eğitim seti, validasyon ve test olguları için konfüzyon matrisleri, doğruluk ve hata payı grafikleri aşağıda verilmiştir (Tablo 1-3, Grafik 1-2). Bu modelde anormallik doğruluk oranı %94 olarak tespit edilmiştir.

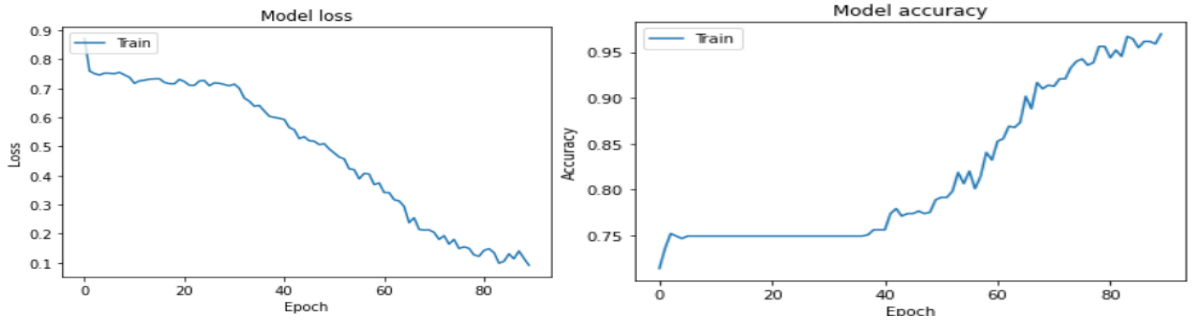
$$anormallik(\%) = \frac{doğru\ tahmin\ edilen\ (sağlıklı + kısmi\ yırtık + kopuk)}{doğru\ tahmin\ edilen\ (sağlıklı + kısmi\ yırtık + kopuk) + yanlış\ tahmin\ edilen\ gerçek\ kopuk + kopuk\ tahmin\ edilen\ gerçek\ yırtık\ veya\ sağlıklı}$$

5x5’lik Filtreler ve 90 Epoch

Konfüzyon Matrisleri (92’şer vaka validasyon ve test setleri, 733 vaka eğitim seti)



***Tablo-1**: Eğitim* ***Tablo-2:** Validasyon* ***Tablo-3:** Test*



***Grafik-1:** Eğitim seti hata payı grafiği* ***Grafik-2:** Eğitim seti doğruluk oranı grafiği*

Tablo-4: Uygulanan altı modelde, tüm data seti, sağlam (0), kısmı yırtıklar (1) ve tam kat yırtıklar (2) için doğruluk ve hata payları. *HP: hata payı.*

	Model	**Tümü (% doğruluk-HP)**			**(0) (% doğruluk)**			**(1) (% doğruluk)**			**(2) (% doğruluk)**		
		Eğitim	Validasyon	Test	Eğitim	Validasyon	Test	Eğitim	Validasyon	Test	Eğitim	Validasyon	Test
1	2x2, 60 Epoch	*83-0.36*	*72-0.76*	*81.5-0.51*	*90.3*	*76.5*	*88.6*	*74.5*	*25*	*41.6*	*75.9*	*33.3*	*0*
2	2x2, 90 Epoch	*87.9-0.26*	*75-1.17*	*79.3-0.60*	*97.3*	*80*	*90.4*	*82.2*	*55.5*	*37.5*	*80.5*	*0*	*33.3*
3	3x3, 60 Epoch	*88.6-0.28*	*72.8-0.77*	*79.3-0.66*	*93.6*	*77.5*	*88.4*	*97.8*	*36.3*	*33.3*	*81.4*	*33.3*	*20*
4	3x3, 90 Epoch	*96.4-0.09*	*74.0-1.54*	*84.8-1.26*	*99.2*	*81.5*	*89.6*	*98.5*	*35.7*	*57.1*	*100*	*50*	*100*
5	5x5, 60 Epoch	*95.2-0.11*	*76.0-1.62*	*83.6-1.32*	*97.5*	*80*	*89.7*	*99.2*	*40*	*35.7*	*97.7*	*100*	*66.6*
6	5x5, 90 Epoch	*96.9-0.09*	*78.2-2.25*	*85.8-1.5*	*98.7*	*81.0*	*89.8*	*81.0*	*60*	*75*	*95.8*	*66.6*	*50*

Sonuç ve Tartışma

Çalışmamızda ÖÇB yırtığı tespiti için farklı derin öğrenme modelleriyle çeşitli doğruluk oranları elde edilmiştir ve bu modellerden "5x5'lik filtre ve 90 Epoch" diğer modellere göre daha başarılı olduğu saptanmıştır. Bu model ile test setinde sağlıklı (0), kısmi yırtıklı (1) ve kopuk (2) ÖÇB'ler için doğruluk oranları sırasıyla %88, %75, %50 olarak bulunmuştur. ÖÇB'de anormallik (kısmi yırtık ve kopuk olanların tespitinde) için ise doğruluk oranı %94 olarak hesaplanmıştır. Kernel boyutundaki ve epoch sayısındaki artış ile beklenildiği gibi modelin başarı oranında da artış sağladığı görülmüştür.

Literatürde Bien ve arkadaşlarının yaptığı çalışmada (9) 1370 olgunun oluşturduğu veri seti iç validasyonla değerlendirilmiş, ÖÇB ve menisküs anomalileri için derin öğrenme modeli geliştirilmiştir. Çalışmada 3 ayrı MRG plandan veri işlenmiştir. Bu çalışmada ÖÇB'de anormallik tespitinde %86.7'ye ulaşılmışken, bizim en iyi modelimizde anormallik %94 olarak hesaplanmıştır.

Liu ve arkadaşlarının yaptığı çalışmada ise, sadece tam kopuk ÖÇB yırtıklarına bakılmış olup 5 radyolog tarafından değerlendirilen veri setinde derin öğrenme modelinin %98'lik bir doğruluk oranı elde ettiğini bildirmişlerdir (10). Yırtık vaka sayısının daha fazla bulunması ve önceden eğitilmiş model kullanmış olmaları doğruluk oranındaki farklılığı açıklayabilir.

Bu çalışmanın sınırlılıklarından biri kullanılan veride vakalara artroskopik veya cerrahi korelasyon yapılıp yapılmadığı ile ilgili bilgi belirtilmemiştir. Bunun yanında veri setimizde ÖÇB'nin durumunu belirten etiket homojen değil orantısız bir şekilde dağılmıştır. Sağlıklı ÖÇB vakaları bütün setin yaklaşık %75'ini oluştururken, yırtık ÖÇB vaka oranı %18 ve kopuk ÖÇB oranı ise yaklaşık %6'sını oluşturmaktadır. Bu durum, modelin sağlıklı ÖÇB vakalarını iyi öğrenmesine sebep olsa bile, yırtık-kopuk ayrımını çok keskin yapamaz hale getirmektedir. Örneğin 5 numaralı modelde sağlıklı, yırtık ve kopuk verilerin doğruluk oranları sırasıyla %88.4, %33.3 ve %20 olarak bulunmuştur. Sağlıklı ÖÇB'nin test setindeki başarı oranı çok daha fazlayken, yırtık ve kopuk doğruluk oranları için veri azlığından dolayı iyi doğruluk oranları elde edilememiştir. Daha sağlıklı bir derin öğrenme modelinin geliştirilebilmesi için hem daha fazla hem de homojen dağılımlı veriye gerek duyulmaktadır.

Geliştirdiğimiz model, literatürdeki örneklerle karşılaştırıldığında yüksek doğruluk oranları ile ön çapraz bağ yırtıklarının derin öğrenme ile tespitinde umut veren sonuçlar ortaya koymuştur.

Öneriler

Modelin üzerinden geçtiği verinin büyüklüğü artılarak daha güvenilir bir model ve daha yüksek doğruluk oranları elde edilebilir. Ancak daha büyük veri setinin işlenmesi için daha güçlü bilgisayarlar gerektirdiği göz önüne alınmalıdır.

Modelin gördüğü veri çeşitliliği artırılarak daha detaylı eğitilebilir ve ÖÇB yırtığı büyüklüğü konusunda daha detaylı bir analiz çıkartılabilir ve yine veri miktarını da artırarak veri

ezberlemesinin önüne geçilerek doğruluk oranı artırılabilir. ÖÇB yırtıkları 3 değil 4 seviyede de incelenebilir.

Daha güçlü bilgisayarlar kullanılarak daha kapsamlı metodlar uygulanabilir. Grafik kartları paralel hesaplama yaparak bilgisayarını verileri daha hızlı işlemesinde ve konvolüsyon katmanındaki ağırlıkları daha hızlı değiştirmesinde yardımcı olur. Örneğin Google Colab, kullanılarak bize sunulan ücretsiz GPU erişiminden faydalanarak modelin daha hızlı ve performanslı eğitilmesi sağlanabilir.

Literatürdeki bazı çalışmalarda, veri setini 3 farklı planda dahil edildiği görülmüştür (aksiyal, sagittal ve koronal). Her kesit için farklı bir derin öğrenme modeli hazırlanmış ardından lojistik regresyonla (3 kesitteki tahminleri sentezlemek) bir global model oluşturulup son tahmin oluşturulmuştur (28) ve bu şekilde en yüksek %95 değerinde bir doğruluk oranı elde edilmiştir. Farklı planlar veri setine eklenerek doğruluk oranlarının yükseltilmesi hedeflenebilir.

Modelin işlediği veriye ve modelin kendisine bağlı olarak da farklı optimizasyon algoritmaları denenerek bu yeni sonuçlar da incelenebilir.

Önceden eğitilmiş bir model yardımıyla da modelin doğruluk oranı artırılabilir. Önceden eğitilmiş modeller önceden bir veri seti üzerinde eğitilmiş olduğu için deneyim sahibi olurlar ve hangi parametrelerin daha iyi sonuç vereceği hakkında fikir sahibi olurlar. Bu onların daha çabuk optimize edilmesini sağlayacak olup önceden eğitilmemiş modellere göre daha yüksek doğruluk oranlarına ulaşabilir. Bu modeller ayrıca önceden çok daha büyük veri setlerinde eğitildiği için, başka bir veri setinde başarılı olabilmeleri için daha küçük veri setleriyle de işlem gerçekleştirebilirler (29).

Kaynakça

1. https://www.acibadem.com.tr/hayat/on-capraz-bag-yaralanmalari-hakkinda-her-sey
2. Chang PD, Wong TT, Rasiej MJ. Deep learning for detection of complete anterior cruciate ligament tear. J Digit Imaging. (2019) 32(6):980–6. doi: 10.1007/ s10278-019-00193-4
3. https://www.medicalpark.com.tr/emar/hg-2048
4. Lassau N, Estienne T, de Vomecourt P, Azoulay M, Cagnol J, Garcia G, et al. Five simultaneous artificial intelligence data challenges on ultrasound, CT, and MRI. Diagn Interv Imaging. (2019) 100(4):199–209. doi: 10.1016/j.diii.2019.02.001
5. J. Liu *et al*., "Applications of deep learning to MRI images: A survey," in *Big Data Mining and Analytics*, vol. 1, no. 1, pp. 1-18, March 2018, doi: 10.26599/BDMA.2018.9020001.
6. Pedoia V, Norman B, Mehany SN, Bucknor MD, Link TM, Majumdar S. 3D convolutional neural networks for detection and severity staging of meniscus and PFJ cartilage morphological degenerative changes in osteoarthritis and anterior cruciate ligament subjects. J Magn Reson Imaging. (2019) 49(2):400–10. doi: 10. 1002/jmri.26246
7. https://www.veribilimiokulu.com/yapay-sinir-agiartificial-neural-network-nedir/
8. https://www.freecodecamp.org/news/binary-classification-made-simple-with-tensorflow/#:~:text=Binary%20classification%20is%20a%20fundamental,fraud%20detection%2C%20and%20many%20more.
9. Bien N, Rajpurkar P, Ball RL, Irvin J, Park A, Jones E, et al. Deep-learningassisted diagnosis for knee magnetic resonance imaging: development and retrospective validation of MRNet. PLoS Med. (2018) 15(11):e1002699. doi: 10.1371/ journal.pmed.1002699
10. Liu F, Guan B, Zhou Z, Samsonov A, Rosas H, Lian K, et al. Fully automated diagnosis of anterior cruciate ligament tears on knee MR images by using deep learning. Radiology: Artificial Intelligence. (2019) 1(3):180091. doi: 10.1148/ryai. 2019180091
11. https://github.com/SohaibAnwaar/Knee-Mri-Research/blob/main/Classify-Segments.ipynb
12. http://www.riteh.uniri.hr/~istajduh/projects/kneeMRI/
13. https://www.medicana.com.tr/saglik-rehberi-detay/10473/on-capraz-bag-yaralanmalari-ve-ameliyati-oncesi-sonrasi
14. https://medium.com/@evertongomede/the-significance-of-train-validation-test-split-in-machine-learning-91ee9f5b98f3
15. https://dergipark.org.tr/tr/download/article-file/380999
16. https://alison.com/course/1326/resource/file/resource_200-1513762488755404360.pdf
17. https://www.researchgate.net/figure/Figure-216-A-8-bit-grayscale-image-pixel-value-ranges-between-0-black_fig14_328828460
18. https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/

19. https://dergipark.org.tr/tr/download/article-file/738321
20. https://www.researchgate.net/figure/ReLU-function-graph_fig2_346250677
21. https://botpenguin.com/glossary/softmax-function
22. https://www.baeldung.com/cs/ml-relu-dropout-layers
23. https://www.datasciencecentral.com/the-artificial-neural-networks-handbook-part-1/
24. https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e#:~:text=Also%20called%20logarithmic%20loss%2C%20log,from%20the%20actual%20expected%20value.
25. https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6
26. https://www.researchgate.net/figure/Softmax-and-cross-entropy-loss-function_fig6_357633642
27. https://stanford.edu/~shervine/l/tr/teaching/cs-230/cheatsheet-deep-learning-tips-and-tricks
28. D. Azcona, K. McGuinness and A. F. Smeaton, "A Comparative Study of Existing and New Deep Learning Methods for Detecting Knee Injuries using the MRNet Dataset," 2020
29. https://www.baeldung.com/cs/neural-network-pre-training#:~:text=A%20model%20that%20has%20a,building%20a%20model%20from%20scratch.

Ekler

model.pdf