Teknoloji Fakültesi

MARMARA ÜNİVERSİTESİ

TEKNOLOJİ FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

BİTİRME PROJESİ

Akciğer X-ray Görüntülerinden Makine Öğrenmesi Yöntemleri İle Hastalık Analizi

PROJE YAZARI

Muhammed Kalabaşı

DANIŞMAN

Prof. Dr. Serhat ÖZEKES

İSTANBUL, 2024

MARMARA ÜNİVERSİTESİ

TEKNOLOJİ FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Marmara Üniversitesi Teknoloji Fakültesi Bilgisayar Mühendisliği Öğrencisi Muhammed Kalabaşı'nın "Akciğer X-ray Görüntülerinden Makine Öğrenmesi Yöntemleri İle Hastalık" başlıklı bitirme projesi çalışması, 03/06/2024 tarihinde sunulmuş ve jüri üyeleri tarafından başarılı bulunmuştur.

Jüri Üyeleri

Prof. Dr. Serhat Özekeş (Danışman)

Marmara Üniversitesi Teknoloji Fakültesi Bilgisayar Mühendisliği (İMZA) .....

Doç. Dr. Eyüp Emre Ülkü (Üye)

Marmara Üniversitesi Teknoloji Fakültesi Bilgisayar Mühendisliği (İMZA) .....

Dr. Öğr. Timur İnan (Üye)

Marmara Üniversitesi Teknoloji Fakültesi Bilgisayar Mühendisliği (İMZA) .....

İÇİNDEKİLER

Sayfa

SEMBOLLER/SYMBOLS

.pkl : Python programlama dilinde "pickle" modülü ile oluşturulan dosyalar.

KISALTMALAR/ABBREVIATIONS

REST : Representational State Transfer

API : Application Programming Interface

LSTM : Long Short Term Memory

RNN : Recurrent Neural Networks

GRU : Gated Recurrent Unit

RGB : Red-Green-Blue

LXMERT : Learning Cross-Modality Encoder Representations from Transformers

BLEU : Bilingual Evaluation Understudy

ROUGE-L : Recall-Oriented Understudy for Gisting Evaluation

METEOR : Metric for Evaluation of Translation with Explicit ORdering

LR : Logistic Regression

CNN : Convolutional Neural Networks

PCA : Principal Component Analysis

GUN : Generative Adversarial Network

CXR : Chest X-Ray

COVID-19 : Coronavirus Disease

ML : Machine Learning

ResNet : Residual neural network

DDİ : Doğal Dil İşleme

BG : Bilgisayarlı Görü

MRI : Magnetic Resonance Imaging

BPTT: Backpropagation Trough Time

SGD: Stochastic Gradient Descent

ReLU: Rectified Linear Unit

PIL: Python Imaging Library

ASGI: Asynchronous Server Gateway Interface

GUI: Graphical User Interface

DOM: Document Object Model

PNG: Portable Network Graphics

CORS: Cross-Origin Resource Sharing

Val_loss: Validation Loss

LR: Learning Rate

ŞEKİL LİSTESİ

Sayfa

TABLO LİSTESİ

Sayfa

ÖZET

Günümüzde makine öğrenmesi yöntemleri, geniş bir yelpazede kullanım alanı bularak hayatımızın birçok alanında devrim yaratmaktadır. Sağlık sektöründe hastalıkların erken teşhisi ve kişiselleştirilmiş tedavi planlarının oluşturulmasında, finans alanında dolandırıcılık tespiti ve risk yönetiminde, perakende sektöründe müşteri davranışlarının analiz edilerek satış stratejilerinin optimize edilmesinde etkin bir şekilde kullanılmaktadır. Ayrıca, otonom araçlardan akıllı şehir uygulamalarına kadar birçok ileri teknoloji projesinde makine öğrenmesi kritik bir rol oynamaktadır. Bu yöntemler, büyük veri kümelerinden anlamlı bilgiler çıkararak karar alma süreçlerini hızlandırmakta ve doğruluğunu artırmaktadır. Böylece, makine öğrenmesi, hem işletmelerin verimliliğini artırmakta hem de bireylerin yaşam kalitesini yükseltmektedir.

Bu çalışmada 3850 hastadan alınan bulgular ve bu hastaların 7000 adet X-ray görseli kullanılmıştır. Veri setindeki görseller ve bu görsellere ait bulgular Indiana Üniversitesi'nin Chest X-Rays veri setinden alınmıştır. Bu hastaların 3200 tanesinin 2 adet X-ray görseli bulunmaktadır. Bunun dışında tek X-ray görseli bulunan hastalar ve 3 adet X-ray görseli bulunan hastalar da bulunmaktadır. Görsellere ait bulgulardaki cümleler yardımcı fiillerin kaldırılması gibi ön işlemlere tabi tutulmuştur. Bu görseller ve bulgular Python'da Pandas kütüphanesi kullanılarak bir DataFrame'de toplanmıştır.

Bu görsellerin özellik vektörlerinin çıkarılması için önceden eğitilmiş CheXNet model ağırlıklarını ve DenseNet121 modeli kullanılmıştır. Model 14 farklı hastalığı tespit edebilecek ve 224x224 piksel boyutundaki 3 kanallı renkli görüntülerle yani RGB görüntülerle eğitilecek şekilde oluşturulmuştur. Bu yeni modelde DenseNet121 modelinin giriş ve çıkış katmanlarında(son 2 katman) CheXNet modelinin ağırlıkları kullanılmıştır. Çıkış katmanlarının çıktısını almak, modelin ara katmanlarında öğrenilen özellikleri kullanmamızı sağlar. Çıkarılan bu özellikler Pandas kütüphanesiyle oluşturulan DataFrame'e eklenmiştir ve bu DataFrame .pkl uzantılı dosya olarak kaydedilmiştir.

Çalışma sonunda bu uygulamalar ile test setinin dışından görseller ile başarıyla yorum üretilebilmiştir.

ÖZET

Today, machine learning methods are revolutionizing various aspects of our lives by being widely applied in numerous fields. In the healthcare sector, they are effectively used for early disease diagnosis and the creation of personalized treatment plans. In finance, they aid in fraud detection and risk management. In the retail industry, machine learning analyzes customer behavior to optimize sales strategies. Moreover, machine learning plays a critical role in many advanced technology projects, from autonomous vehicles to smart city applications. These methods accelerate decision-making processes and increase their accuracy by extracting meaningful insights from large datasets. Consequently, machine learning enhances both business efficiency and individual quality of life.

In this study, findings from 3,850 patients and 7,000 X-ray images from these patients were used. The images and their corresponding findings were obtained from Indiana University's Chest X-Rays dataset. Among these patients, 3,200 have two X-ray images each. Additionally, there are patients with a single X-ray image and some with three X-ray images. The sentences in the findings associated with the images underwent preprocessing, such as the removal of auxiliary verbs. These images and findings were compiled into a DataFrame using Python's Pandas library.

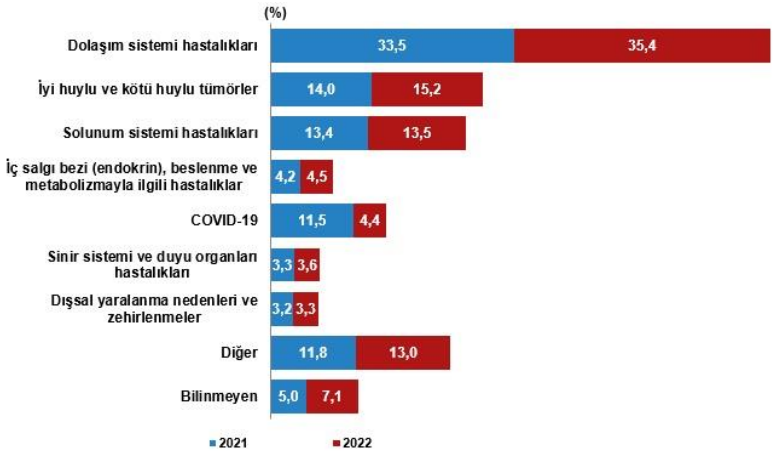
To extract feature vectors from these images, the pretrained CheXNet model weights and the DenseNet121 model were used. The model was designed to detect 14 different diseases and was trained with 224x224 pixel, three-channel color images, i.e., RGB images. In this new model, the weights from the CheXNet model were used in the input and output layers (the last two layers) of the DenseNet121 model. Obtaining outputs from the output layers allows us to use the features learned in the intermediate layers of the model. These extracted features were added to the DataFrame created with the Pandas library, and this DataFrame was saved as a .pkl file.

At the end of the study, these applications successfully generated interpretations using images from outside the test set.

1. GİRİŞ

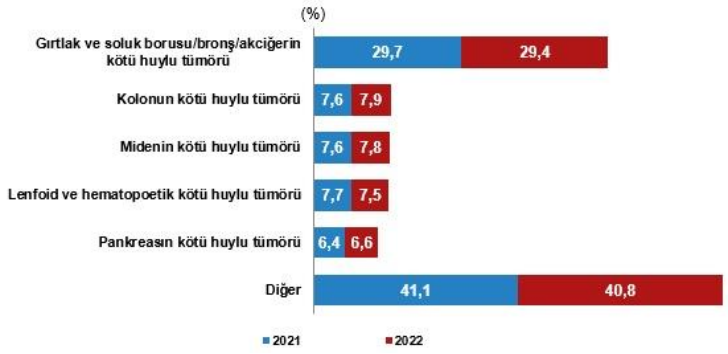
KOAH, dünya çapında üçüncü önde gelen ölüm nedenidir ve her yıl 3,2 milyon bireyi öldürmekte olup, kronik solunum hastalıklarından ölümlerin toplamının şaşırtıcı bir şekilde %81,7'sini oluşturmaktadır. Zatürre, yenidoğan döneminin dışında <5 yaşındaki çocuklar ve >65 yaşındaki yetişkinler arasında önde gelen bir ölüm nedenidir. Akciğer kanseri, hemen hemen tüm kanser türleri arasında en ölümcül olanıdır ve gelişmiş ülkelerde bile 5 yıllık sağkalım oranları sadece %10 ile %20 arasındadır. Astımın çocukluk çağının en yaygın kronik hastalığı olduğunu biliyoruz ve astım prevalansı son otuz yıldır artmaktadır. COVID-19 pandemisi olmasaydı, verem (TB) tek başına en büyük bulaşıcı ölüm nedeni olurdu; sadece 2019'da tahmini 500.000 yeni rifampisin dirençli veya çoklu ilaç dirençli TB vakası görülmüştür. Burada bahsedilen ilk beş durumun yanı sıra, rapor aynı zamanda uyku bozuklukları, pulmoner hipertansiyon, meslek hastalıkları ve hava kirliliği ile iklim değişikliğinin zararlı etkilerinin önemli küresel yükünü de vurgulamaktadır. [1]

Kronik obstrüktif akciğer hastalığı, astım, akciğer enfeksiyonları (zatürre, verem ve bronşit gibi), solunum yetmezliği, akciğer kanseri ve sertleşmesi gibi pek çok hastalığı içeren solunum sistemi hastalıkları; Türkiye'de tüm ölümlerin arasında üçüncü sırada gelmektedir. Ölümler nedenlerine göre incelendiğinde, 2022 yılında %35,4 ile dolaşım sistemi hastalıkları ilk sırada yer aldı. Bu ölüm nedenini %15,2 ile iyi huylu ve kötü huylu tümörler, %13,5 ile solunum sistemi hastalıkları izledi. Bahsedilen yılda Türkiye'deki ölüm sayısı 505 bine yakın kişi ölmüştür. Bu da demek oluyor ki solunum sistemi hastalıkları nedeniyle 2022 yılında Türkiye'de 70 bine yakın kişi yaşamını kaybetmiştir.



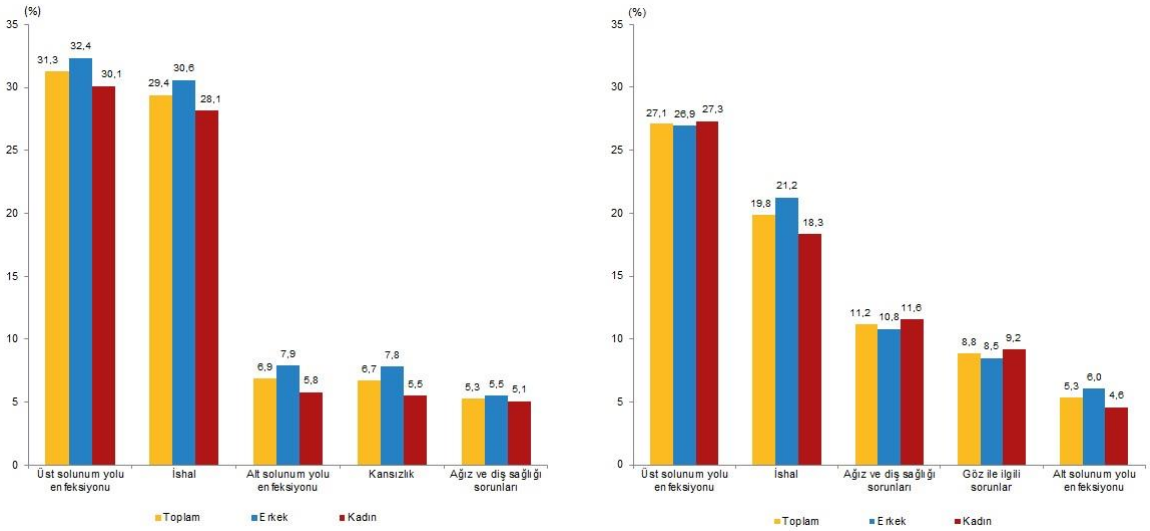
Şekil 1.1. Nedenlere göre ölüm oranı, 2021, 2022

Ayrıca aynı yılda yapılan araştırmada tümörden kaynaklı ölümlere en çok gırtlak ve soluk borusu/bronş/akciğer tümörü neden olmuştur. İyi ve kötü huylu tümörlerden kaynaklı ölümler alt ölüm nedenlerine göre incelendiğinde, ölenlerin %29,4'ünün gırtlak ve soluk borusu/bronş/akciğerin kötü huylu tümöründen öldüğü görülmüştür. [2]



Şekil 1.2. İyi ve kötü huylu tümörlerden kaynaklı ölümlerin alt ölüm nedenleri

Türkiye'de 2022 yılında yapılan araştırmaya göre üst solunum yolu enfeksiyonu 0-14 yaş grubundaki çocuklarda en fazla görülen hastalık olmuştur. Çocuklarda son 6 ay içinde görülen hastalık türleri incelendiğinde, 2022 yılında 0-6 yaş grubunda %31,3 ile en çok üst solunum yolu enfeksiyonu görülmüştür. 2022 yılında 7-14 yaş grubunda da %27,1 ile üst solunum yolu enfeksiyonu ilk sırada yer almıştır. [3]



Şekil 1.3. Çocukların başlıca hastalık/sağlık sorunlarının cinsiyete göre dağılımı, 2022

Solunum yolu ve akciğer hastalıkları teşhisi için mevcut en iyi yöntem, klinik bakımda ve epidemiyolojik çalışmalarda hayati bir rol oynayan göğüs röntgenleridir. Ancak, göğüs röntgenlerinde bu hastalıkları tespit etmek, uzman radyologların mevcudiyetine dayanan zor bir görevdir. Yüksek oranlarda çıkan bu hastalıkların tespiti ve tespit edilen hastalıklar için

ise destek almak mevcudiyeti şart görülmüştür. Bu çalışmada, bu gerekliliğe karşın bir çözüm olarak yapılmıştır ve göğüs röntgenlerinden otomatik olarak solunum yolu ve akciğer hastalıklarını tespit edebilen ve pratisyen radyologların seviyesinde bir model sunulmuştur. Bu sayede solunum yolu ve akciğer hastalıkları teşhisi için makine öğrenmesi tekniklerine başvurulabilir veya teşhis edilmiş hastalıklar için bir destek mekanizması görmüştür.

Bu model, 3850 hastadan alınan 7000 adet göğüs röntgeni görüntüsü içeren ve her biri zatürre dahil olmak üzere 14 farklı torasik hastalıkla ayrı ayrı etiketlenip bulguları eklenmiş olan, veri kümesinde eğitilmiştir.

Uygulamanın kullanım alanları ayrıntılı bir şekilde aşağıda sıralanmıştır:

Erken Tanı ve Önleyici Sağlık Hizmetleri: Proje erken tanı ve önleyici sağlık hizmetlerinde büyük bir potansiyele sahiptir. Otomatik tanı sistemleri, özellikle kırsal ve yetersiz hizmet alan bölgelerde, erken teşhis konulmasına yardımcı olabilir. Bu sayede, hastalıkların ilerlemesi önlenebilir ve tedavi süreçleri daha etkin bir şekilde yönetilebilir.

Teleradyoloji ve Uzaktan Danışmanlık: Teleradyoloji, radyolojik görüntülerin uzaktan analiz edilmesini sağlayan bir teknolojidir. Proje, bu alanda da önemli bir rol oynayabilir. Otomatik olarak oluşturulan raporlar ve açıklamalar, radyologların uzaktan teşhis koymalarını kolaylaştırır ve daha hızlı ve doğru kararlar almalarına yardımcı olur. Bu durum, özellikle uzman hekim eksikliğinin yaşandığı bölgelerde büyük bir avantaj sağlar.

Eğitim ve Araştırma: Tıbbi eğitim ve araştırma alanlarında, akciğer görüntüleri kullanılarak geliştirilen bu önemli bir kaynak olabilir. Tıp öğrencileri ve uzmanlar, bu projeler sayesinde çeşitli akciğer hastalıklarının belirtilerini ve tanı yöntemlerini öğrenebilirler. Ayrıca, bu teknolojiler araştırmacılara geniş veri kümeleri üzerinde çalışarak yeni tanı ve tedavi yöntemleri geliştirme imkânı sunar.

Hastane ve Klinik Yönetimi:

Hastane ve kliniklerdeki iş yükünü azaltmak için kullanılabilir. Otomatik olarak oluşturulan raporlar ve tanılar, sağlık personelinin daha verimli çalışmasını sağlar ve hasta başına düşen inceleme süresini azaltır. Bu sayede, daha fazla hastaya hizmet sunulabilir ve genel sağlık hizmetleri kalitesi artar.

Hasta Takibi ve Tedavi Planlaması: Hastaların tedavi süreçlerinin takibi ve planlanmasında da kullanılabilir. Hastalıkların ilerleyişini izlemek ve tedaviye yanıtı değerlendirmek için düzenli olarak alınan röntgenler, bu sistemler sayesinde otomatik olarak

analiz edilebilir. Bu durum, doktorların hastaların durumunu daha yakından takip etmelerini ve gerektiğinde tedavi planlarında değişiklik yapmalarını sağlar.

2. LİTERATÜR TARAMASI

Literatür taramasında yerli ve yabancı literatür incelendiğinde akciğer röntgen görüntülerinden yararlanılarak yapılan birçok çalışmaya rastlanmıştır. Bu kısımda makine öğrenmesi ile solunum yolu ve akciğer hastalıkları teşhisi ile ilgili bazı literatür çalışmalarına yer verilmiştir.

Wijerathna, Raveen, Abeygunawardhana ve D. Ambegoda'nın yaptığı çalışmada iki yaklaşım kullanılmıştır. İlk olarak, Faster RCNN modeliyle birleştirilmiş modelde cümle oluşturucu olarak orijinal olarak soru cevaplamak için tasarlanmış LXMERT kullanmayı denenmiştir. İkinci olarak, özellik çıkarıcı ve cümle oluşturucu olarak sırasıyla CheXNet ve bellek odaklı bir transformatör kullanılmıştır. Model IU Chest X Ray veri seti kullanılarak eğitilmiş ve test edilmiştir. Model BLUE, ROUGE-L ve METEOR metrikleri kullanılarak değerlendirilmiştir.[4]

Selivanov, Rogov, Chesakov, Shelmanov, Fedulova ve Dylov'un yaptığı araştırmada otomatik klinik görüntü altyazı oluşturma için önerilen model, radyolojik taramaların analizini metinsel kayıtlardan yapılandırılmış hasta bilgileriyle bileştirilmiştir. Bu model, kapsamlı ve açıklayıcı radyoloji kayıtları oluşturmak için Show-Attend-Tell ve GPT-3 olmak üzere iki dil modelini kullanmıştır. Üretilen metinsel özet, bulunan patolojiler hakkında önemli bilgiler, bunların konumları ve her bir patolojiyi taramalarda yerelleştiren 2D ısı haritalarını içerir. Model, iki tıbbi veri kümesi olan Open-I ve MIMIC-CXR ile genel amaçlı MS-COCO üzerinde test edilmiştir ve doğal dil değerlendirme metrikleriyle ölçülmüştür.[5]

Bu makalenin araştırmacıları olan Rasheed, Hameed, Djeddi, Jamil ve Al-Turjman, X-ray görüntülerinden yüksek doğrulukla otomatik korona virüs teşhisi için makine öğrenme yöntemlerinin potansiyelini araştırmıştır. İki en yaygın kullanılan sınıflandırıcı seçmişlerdir: LR ve CNN. Ayrıca, öğrenme sürecini hızlandırmak ve yüksek ayırt edici özellikleri seçerek sınıflandırma doğruluğunu artırmak için PCA tabanlı bir boyut azaltma yaklaşımı da araştırılmıştır. Derin öğrenme tabanlı yöntemler, geleneksel yaklaşımlara kıyasla büyük miktarda eğitim örneği gerektirir, ancak COVID-19 X-ray görüntüleri için yeterli miktarda etiketlenmiş eğitim örneği mevcut değildi. Bu nedenle, eğitim örneklerini artırmak ve aşırı öğrenme problemini azaltmak için GAN kullanılarak veri artırma tekniği uygulanmıştır. Bu

çalışma için çevrimiçi mevcut veri setini kullanarak toplamda 500 X-ray görüntüsü elde etmek için GAN kullanılmıştır.[6] Aynı şekilde Castiglioni, Ippolito, Interlenghi, Monti, Salvatore, Schiaffino, Polidori, Gandola, Messa ve Sardanelli tarafından yapılan araştırmada da CXR ile ML kullanılarak COVID-19 hastalığı teşhis edilmiştir ve bu alanda yapılan çalışmaların potansiyelini ortaya koymuşlardır.[7]

Dong, Pan, Zhang ve Xu'nun araştırmasında ise CNN kullanılarak göğüs röntgenleri için bilgisayar destekli bir teşhis sistemi tasarlama olasılığı araştırılmıştır. Doğal dil teşhis raporlarına sahip 16.000 göğüs röntgeninden oluşan gerçek dünya veri seti kullanılarak, herhangi bir önceden alan bilgisi olmadan görüntülerden çok sınıflı bir sınıflandırma modeli eğitilmiştir.[8]

Baltruschat, Nickisch, Grass, Knopp ve Saalbach'in yaptığı araştırmada transfer learning ile ve fine-tuning olmadan sıfırdan özel bir X-ray ağı eğitimi seçeneklerini değerlendirmişlerdir. X-ray verilerinin yüksek uzaysal çözünürlüğünden yararlanmak için genişletilmiş bir ResNet-50 mimarisini ve sınıflandırma sürecine non-image data (hastanın yaşı, cinsiyeti) dahil eden bir ağı da dahil etmişlerdir. Bir deneyde, farklı ResNet derinliklerini (örneğin ResNet-38 ve ResNet-101) de araştırmışlardır.[9]

3. MATERYAL VE YÖNTEM

3.1. Kullanılan Materyaller

3.1.1 Doğal Dil İşleme

Diller bilindiği üzere ikiye ayrılır; makine dili(machine language) ve insanlar tarafından kullanılan doğal dildir(natural language). Bilgisayarların insanların dillerini anlaması, onlarla iletişime geçmeleri için doğal dil işleme bilimi kullanmaları gerekmektedir. Kısacası, bilgisayarların doğal dilleri işleme sürecidir. Bunu insanbilgisayar etkileşimi (human computer interaction-HCI) (Preece, 1994.) biliminin altında da görülebilir veya hesaplamalı dil bilimi(compurtational linguistics-CL) olarak da görülebilir. Burada dilin hesaplanması işlem görmesi durumudur.

Veri bilimiyle doğal dil işleme arasındaki ilişki düşünüldüğünde, aslında bu konuyu metin işleme(Text Processing) ve metin madenciliğinin(text mining) (Seker, Metin Madenciliği(Text Mining), 2015) altında düşünmek gerekir. Çünkü veri biliminin genelde ilgilendiği kısım buralardır. Bunun sebebi, eğer veri kaynakları metin şeklindeyse bunların

ilk önce analiz edilmesi ve işlenebilir hale getirilmesinde doğal dil işleme önemli bir rol oynamaktadır.

Doğal dil işleme girdilere göre ikiye ayrılır. Yazılı metinlere (text processing) göre ve ses (speech processing) üzerine yapılanlardır. Doğal dilin genellikle iki kaynağı bulunmaktadır. Birinin konuşması durumunda ona sesli yanıt verilmesi veya yazılı bir metnin analiz edilmesi olarak düşünülebilir. Doğal dil işleme çalışmaları şu şekilde olmaktadır; ses üzerinden çok fazla çalışma yapılamamakla birlikte daha çok sesli anlatımların yazılı hale getirilip ve daha sonra yazılı hale getirilen metinler işleme dahil edilmeleriyle oluşmaktadır.

Doğal dil işlemenin çalışma seviyelerine bakıldığında dört ana madde görülmektedir. Kelime bilimi (Morphological-Lexical), söz dizimsel (syntactic), anlamsal(semantic) ve söylevdir(pragmatic-discourse). Doğal dil işleme aşamaları bu dört ana madde üzerinden anlatılmaya çalışılacaktır. Kelime bilimi(lexical), kelimelerin anlamlarının anlaşılmasıyla ilgilenir. Bir kelimenin kökünün hangi ekler alarak hangi anlama dönüştüğünü inceler ve bu sayede kelimenin ne olduğu anlaşılır. Söz dizimsel (syntactic), cümledeki kelimelerin dizilim şeklidir. Bir cümlede kelimelerin nasıl dizildiğiyle ilgilenir. Anlamsal(semantic), kullanılan cümlelerin anlamlarını inceler. Çünkü doğal dilin bilgisayar tarafından doğru işlenebilmesi için ilk, bu cümlenin doğru bir şekilde anlaşılmasıdır. Söylev(pragmatics-discourse), bir konuşma sırasında kullanılan kelimeler ve anlamlarıyla ilgilenir. İlk önce bilgisayar tarafından konuşma anlaşılmalı ve nelerden bahsedildiği anlaşılan cümlelere doğru kelimeler kullanılarak cevap verilmesi durumudur.

Doğal dil işlemenin bir diğer kısmı üretim (Generation) kısmıdır. Burada bir dilin anlaşılması için gerekli zamanı yine bu dili üretmek için de kullanılmalıdır. Örneğin bilgisayarın ürettiği bir Türkçe kelime olarak düşünülebilir.

Doğal dil işleme (DDI), günümüz teknolojilerinde bir çok alanda kullanılmaktadır bunlardan bazılarını şu şekilde sıralamak mümkündür:

Soru cevaplama (Question answering): Doğal dil işleme konusunun ilk problemlerinden biridir. Verilen bir metin üzerinden düşünüldüğünde, bu metnin analiz edilerek anlamlı soru ve cevap çıkarılması ya da verilen sorular üzerinden o soruların cevaplarının bir metin üzerinden bulunması istenilir. Bu sorunun çözümü aynı zamanda arama motorlarının da bir problem olan ‘konuşarak (sesli)arama yapabilme sorunu’ çözülmesine de sebep olacaktır.

Otomatik tercüme (machine translation): Bir makine tarafından bir metnin tercüme bir dilden başka bir dile çevirme işlemi yapılması istenilmektedir.

Bilginin Getirilmesi (Information retrieval): Aranılan bir bilginin metin kaynakları arasında aranıp bulunması ve getirilmesi işlemidir.

Kelime işleme (Word Processing (Grammer checking, spell checking ,spell correction,Find/Replace)): Sıkça kullanılan bir işlem olan kelime işleme bir çok alanda kullanılan bir takım işlemlerden oluşur. Metin üzerinde yazımsal hataları kontrol etmek için dil bilgisi kontrol işlemi (Grammer checking) kullanılır. Ya da bir kelime için heceleme kontrol ve düzeltme (spell checking and spell correction) işlemleri kullanılır. Kemal Oflazer (Oflazer, (1996) tarafından yazılan bir makale de, Word programında bir kelime arama (Find/replace) işlemi yapıldığında, bul ve değiştir işleminin uygulanması istenildiğinde; örneğin bir metinde okul kelimesinin ev ile değiştirilmesi istenildiğinde, metinde bulunan bütün okul kelimelerinin ev kelimesi ile değiştirildiği görülmektedir.

Fakat burada okullar kelimesi düşünüldüğünde bu kelimenin evlar olarak değiştiği görülecektir. Bu kelime hatasının evler olarak düzeltilme durumu da hece düzeltme-kontrol (spell checking-correctoin) işlemi uygulanmalıdır.

Doğal Dil İşleme-Veri Tabanı Arayüzü (NL/DB interface): Veri tabanı üzerinden sorgulama işlemi yapıldığında kullanılan bir işlemdir. Bir üretim departmanında soru olarak gelen ‘Bu yılki satış oranı nedir?’ sorusunu veri tabanında uygulamak için bir sorgu cümlesine çeviren bir sistemdir.

İnternet Arama Motoraı (Web Search): Doğal dilde arama olarak internet arama motorları sayesinde arama yapma işlemidir. Örnek olarak “İstanbul-Kayseri arası kaç km’dir? Everest Dağ’ının yüksekliği ne kadardır?” Gibi sorulara cevap aranabilmesi.

Anti-Intelligence (Al) Bots(Siri): Konuşularak verilen bir soru üzerine makine tarafından yine konuşarak karşılık verilmesi. İstenilen sorulara arama yaparak bularak karşılık vermesi ve günümüz teknolojilerinde sıkça rastlanan akıllı sistemlerde görülen durumdur. Bu durumun en bilinen örneği SİRİ uygulamasıdır.

Metin İşleme (Text processing)(Categorization,clustering,IE): Bir metin üzerinde yapılan kategorize etme, bölütleme ve kümeleme işlemidir. Bir önceki konu olan metin madenciliği bu konuları daha detaylı görmek mümkündür.

Finance,Chat Rooms,Telefon Cevaplama: Finansal raporlama ,sohbet odaları ve çokça rastlanılan telefon cevaplama sistemleri. Arama sonucunda karşı tarafta bir makinenin olması ve sorulan sorulara aldığı cevaplar ile yönlendirme yapması ya da sohbet odalarında sohbet diyaloglarına karşılık veren makinelerin bulunması.

Bilgi Deposu Olarak İnternet (Internet as a repository): İnternet ortamının en büyük bilgi kaynağı ve ulaşılabilirlik açısından en kolay yol olduğu bilinmektedir. Düşünün her gün milyonlarca kişi tarafından gerek kendi sitelerinde gerek bloglarında gerekse kendi kişisel sayfalarında bir şeyler paylaşmaktadırlar. Paylaşılan bu bilgilerin anlaşılıp, modellenip ve kümelenmesi durumunda bütün internet ağı üzerinde yani insanların ürettiği en büyük bilgi kaynağında, sonuçta insanlık tarihi birçok bilgi kaynağı üretmiştir, geçmiş zamanlarda, ansiklopediler, yazıtlar gibi ve bu kaynakların hemen hemen hepsi internet kaynağına çevrilmiştir, bilgisayarlar tarafından doğal dil işleme yöntemi sayesinde ortaya çıkması diğer insanlarında bu bilgiye çok kolay bir şekilde ulaşmaları demektir.

Metin Özetleme (Text summarization): İnternet üzerindeki bir milyon tane kaynak üzerinden düşünüldüğünde bunlar üzerinden yapılan bir sonucunun bu sayfalar üzerinden bulunması ve özetleme yapılmasıdır.

Argümanların Birleştirilmesi (Argument aggregation): Sosyal ağlar üzerinden yapılan tartışmalar sonucu ortaya çıkan argümanların analiz edilmesi tekrarlı olanların ya da boş olanların temizlenmesi ve kalan argümanlardan anlamlı verilerin üretilmesiyle sonucun bir argümanda birleştirilmesi gerekmektedir. Bu sayede başlatılan bir tartışmaya daha kolay dahil olunması sağlanır. Daha önce hangi argümanlar konuşulmuş neler tartışılmış bunların bilinmesi durumudur.

3.1.2 Bilgisayarlı Görü

İnsanın dış dünyayı algılaması, en önemli algılama kanallarından birisi olan görüntüleri algılaması ve analiz ederek anlamlandırmasıyla oluşur. İnsandaki görsel algılama ve anlama yetisini bilgisayarda oluşturmayı amaçlayan tüm teknikler bilgisayarla görü ve bilişsel öğrenme alanına girer. Yapay zekanın günümüzde en çok adı geçen derin öğrenme alt alanının ilerlemesinde en büyük rolü dünyada bilgisayarla görü alanında çalışan bilim insanları oynamıştır.

Bilgisayarla görü, iki boyutlu, üç boyutlu veya daha yüksek boyutlu her türlü görsel sayısal verinin özellikle akıllı algoritmalarla anlamlandırılmasını amaçlamaktadır. Bilgisayarla görü

alanı bu problemleri çözmekte, geometri, lineer cebir, olasılık ve istatistik kuramı, türevsel denklemler, çizge kuramı ve son senelerde özellikle makine öğrenmesi ve derin öğrenme gibi hem matematiksel hem de hesaplama kuramına dayalı tekniklerin geliştirilmesini ve gerçeklenmesini araç olarak kullanır. Standart kamera görüntülerinin yanı sıra medikal görüntüler, uydu görüntüleri ve üç-boyutlu nesnelerin ve sahnelerin bilgisayar ortamında modellenmesi bilgisayarla görünün ilgi alanına girer.

Bir görüntü tanımlama yazılımı uygulaması aşağıdaki tekniklerden birini kullanabilir:

Nesne tanımlama: Bu teknik, bir görüntüdeki belirli bir nesneyi algılamak için kullanılır. Gelişmiş sürümleri, tek bir görüntüdeki birden çok nesneyi tanımlayabilir.

Görüntü sınıflandırması: Görüntülerin kategoriler halinde gruplandırılması tekniğidir. Görüntülere etiket atama işlemi olarak da adlandırılır.

Görüntü segmentasyonu: Bir görüntüyü parçalara ayırarak ayrı ayrı incelemek için kullanılan tekniktir.

Kalıp algılama: Görsel verilerdeki kalıpları ve devamlılıkları tanımlar.

Köşe algılama: Görüntünün bileşenlerini daha iyi tanımlamak için bir nesnenin köşelerini algılama işlemidir.

Özellik eşleştirme: Sınıflandırma amacıyla görüntülerdeki benzerlikleri eşleştiren bir tür desen algılama tekniğidir.

BG günümüzde birçok sektörde kendine kullanım alanı buluyor. Facebook, bu teknolojiyi kullanarak fotoğraftaki insanları otomatik olarak etiketleyebiliyor, Google fotoğrafları grupluyor, Adobe ise yakınlaştırılmış görüntülerin kalitesini iyileştiriyor. Bunlar dijital örneklerken, BG'nin fiziksel dünyada da birçok uygulama örneği bulunuyor.

BG aşağıdaki alanlarda kullanılabilir:

Yüz tanıma: BG'nin en iyi kullanım örneklerinden bazıları yüz tanıma alanında görülüyor. Apple'ın 2017'de piyasaya sürdüğü iPhone X modeliyle popüler hale gelen yüz tanıma, günümüzde çoğu akıllı telefonda bulunan standart bir özelliğe dönüştü.

Yüz tanıma teknolojisi akıllı telefonlarda kimlik doğrulamanın yanı sıra, Facebook örneğinde olduğu gibi kişi tanımlamak amacıyla da kullanılıyor. Öte yandan, dünyanın dört bir yanından kolluk kuvvetlerinin, video yayınlarında kanunları çiğneyenleri tespit etmek için yüz tanıma teknolojisini kullandığı biliniyor.

Otonom araçlar: Otonom araçlar, gerçek zamanlı görüntü analizi için BG'yi kullanıyor. Bu teknoloji, sürücüsüz otomobillerin çevrelerini anlamlandırmalarına yardımcı oluyor. Otonom sürüş teknolojileri hâlâ emekleme aşamasında ve güvenle trafiğe çıkmaları için daha fazla AR-GE çalışmasına ihtiyaç duyuluyor.

Sürücüsüz otomobiller BG olmadan çalışamaz. Bu teknoloji, otonom araçların görsel verileri gerçek zamanlı işlemesine yardımcı olur. BG otonom araçlarda nesne tanımlama ve sınıflandırmanın yanı sıra, araçlar için 3B haritalar oluşturur.

BG'nin bu alandaki diğer önemli kullanım durumları araç ve şerit çizgisi tespiti ile boş alan algılamadır. Adından da anlaşılacağı gibi, bu teknik araç çevresindeki engelsiz alanları saptamak için kullanılır. Serbest alan tespiti, sürücüsüz otomobil yavaş hareket eden bir araca yaklaştığında ve şerit değiştirmesi gerektiğinde kullanışlıdır.

Tıbbi görüntüleme: BG, sağlık sektöründe daha hızlı ve doğru teşhisler koymak, hastalıkların ilerlemesini izlemek için kullanılır. Doktorlar, kalıp algılama modellerini kullanarak kanser gibi gözle görülemeyen hastalıkların erken semptomlarını tespit eder.

BG ile gerçekleştirilen tıbbi görüntüleme analizi, tıp uzmanlarının görüntüleri analiz etmesi için gereken süreyi kısaltır. Endoskopi, X-ışını radyografisi, ultrason ve (MRI; BG kullanan tıbbi görüntüleme disiplinlerinden bazılarıdır.

Tıp uzmanları, evrişimsel sinir ağlarını tıbbi görüntüleme ile eşleştirerek iç organları gözlemleyebilir, anormallikleri tespit edebilir, belirli hastalıkların nedenini ve etkisini anlayabilir. Ayrıca doktorların hastalıkların gelişimini ve tedavilerin ilerlemesini izlemesine yardımcı olur.

İçerik denetimi: Sosyal medya ağlarının her gün milyonlarca yeni gönderiyi gözden geçirmesi gerekir. Gönderilen her görüntü veya videoyu inceleyen bir içerik denetleme ekibine sahip olmak artık pratik değil ve bu nedenle süreci otomatikleştirmek için BG sistemlerini kullanılıyor. BG, sosyal medya platformlarının yüklenen içerikleri analiz etmesine ve sakıncalı görüntüler içerenleri işaretlemesine yardımcı oluyor. Şirketler, rahatsız edici yazılar içeren paylaşımları belirlemek ve engellemek için metin analizi için derin öğrenme algoritmaları da kullanıyor.

Gözetim: Video yayınları sağlam bir kanıt biçimidir. Kanunları çiğneyenleri keşfetmeye ve güvenlik uzmanlarının küçük endişeler felakete dönüşmeden önce harekete geçmesine yardımcı olurlar. İnsanların birden fazla kaynaktan gelen gözetleme görüntülerini takip

etmeleri neredeyse imkansızdır ancak bu BG için kolay bir görevdir. BG destekli gözetim sistemleri, canlı görüntüleri tarayabilir ve şüpheli davranışları tespit edebilir.

3.1.3 Image Captioning

Günümüzde veri setlerindeki veriler çoğalıyor, karmaşıklaşıyor. Birden fazla görevin aynı anda gerçekleşmesini gerektiren problemlerin sayısı günden güne artmıştır ve böylece hem farklı disiplinler hem de aynı disiplin içerisindeki farklı çalışma alanları da hiç olmadığı kadar birbirlerine yaklaşmıştır.

Görüntü alt yazılama problemi de Bilgisayar Biliminin iki önemli çalışma alanı olan DDİ ve BG'yi bir araya getiren, zorlu bir görev olarak karşımıza çıkmıştır.

Bir görüntünün içeriğinin açıklanması amacıyla otomatik olarak cümle ya da ifade üretme problemi olarak tanımlanabilen bu görev, BG ile görüntü içeriğinin yakalanmasını ve yakalanan bu içeriğin DDİ modeli tarafından en doğru şekilde ifade edilmesini gerektirmektedir. [10] Görüntünün içeriğini anlamanın yanı sıra metin üretme gibi zorlu bir görev sonunda elde edilecek çıktının başarılı olmasında; eğitim kümesindeki görüntü açıklamalarının doğruluğu, görüntüyü yeterince iyi ifade edebiliyor olması ve gürültü düzeyi doğrudan etkili unsurlardır. Image captioning'de örnek veri kümeleri şunlardır:

Flickr8k: 8.000 görüntü ve her görüntüye ait 5 farklı açıklama barındırır. Görüntüler altı farklı Flickr grubundan seçilmiş ve çok tanınan kişileri veya yerleri içermemeye özen gösterilerek manuel olarak bir araya getirilmiştir.

Flickr30k: Bir benchmark veri kümesidir. 31,783 görüntü ve her görüntüye ait 5 tane olmak üzere toplamda 158,915 açıklama barındırır. Ancak bu kez açıklamalar cümle şeklindedir.

Google Conceptual Captions: Veri kümesi yaklaşık 3.3 milyon görüntü/açıklama çiftini barındırır. Bu veri kümesinde her resime ait 1 açıklama bulunur.

MS COCO: 120.000 görüntü ve her görüntüye ait 5 farklı açıklama barındıran ve birçok farklı bilgisayarlı görü görevi için literatürde sıkça kullanılan bir veri kümesidir.

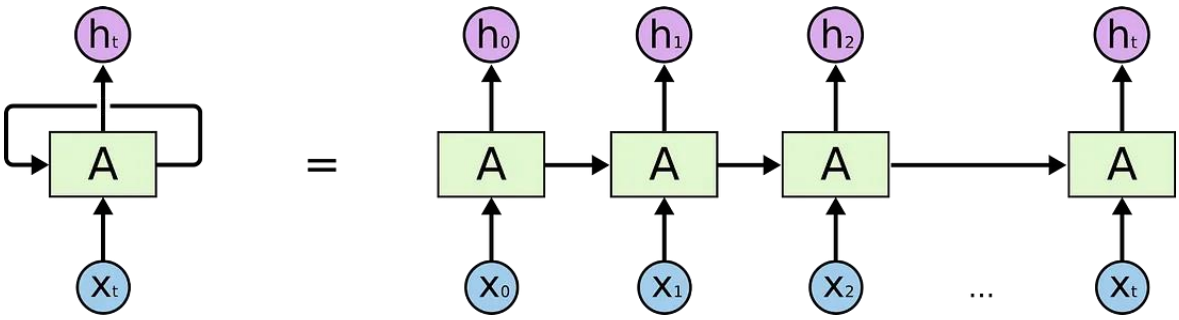
3.1.4 RNN

İleri beslemeli(feedforward) çalışan yapıda gelen bilgi sadece ileri doğru işlenir. Bu yapıda kabaca, girdi verileri ağdan geçirilerek bir çıktı değeri elde edilir. Elde edilen çıktı değeri doğru değerler ile karşılaştırılarak hata elde edilir. Ağ üzerindeki ağırlık değerleri hataya bağlı olarak değiştirilir ve bu şekilde en doğru sonucu çıktı verebilen bir model oluşturulmuş

olur. İleri beslemeli bir ağın eğitiminde hatanın yeterince düşürülmesi gerekir. Böylece nöronlara giden ağırlıklar yenilenerek girilen girdiye uygun çıktı verecek bir yapı oluşmuş olur.

Örneğin, bir fotoğraf üzerindeki nesneleri kategorize etmek için eğitilen ileri beslemeli bir ağ düşünelim. Verilen fotoğraf rastgele bir sıra ile de olsa hem eğitim hem de test için kullanılabilir. Bir önceki veya bir sonraki fotoğraf ile herhangi bir bağının olması gerekmez. Yani zamana veya sıraya bağlı bir kavram yoktur, ilgilendiği tek girdi o andaki örnektir.

Yinelenen(recurrent) yapılarda ise sonuç, sadece o andaki girdiye değil, diğer girdilere de bağlı olarak çıkarılır. RNN'de t anındaki girdi verilerinin yanında, t-1 anından gelen gizli katman (hidden layer) sonuçları da gizli katmanın t anındaki girdisidir. t-1 anındaki girdi için verilen karar, t anında verilecek olan kararı da etkilemektedir. Yani bu ağlarda girdiler şimdiki ve önceki bilgilerin birleştirilmesi ile çıktı üretirler.[11]



Şekil 3.1 RNN'lerin çalışma mekanizması

Yinelenen yapılar, çıktılarını sonraki işlemde girdi olarak kullandıkları için ileri beslemeli yapılardan ayrılmış olurlar. Yinelenen ağların bir belleğe sahip olduğunu söyleyebiliriz. Bir ağa hafıza eklemenin sebebi ise, belli bir düzende gelen girdi setinin, çıktı için bir anlamı olmasıdır. Bu çeşit veri setleri için ileri beslemeli ağlar yeterli olmaz.

Tam bu noktada RNN'ler devreye girer. Yazı, konuşma, zamana bağlı çeşitli sensör veya istatiksel veriler gibi belli bir sıra ile gelen verilerin yapısını anlamada yinelenen ağlar kullanılır.

RNN'in işlem döngüsünde gizli katmandan çıkan sonuç hem çıktı üretir, hem içerik ünitelerine (content unit) yazılır. Bu şekilde, her yeni girdi, önceki girdilerin işlenmesi sonucu üretilmiş içerik üniteleriyle birlikte işlenir. Farklı zamanlarda belleğe alınan veriler arasında korelasyon bulunuyorsa buna "long term" bağımlılık denir. RNN, bu long-term bağımlılıkların arasındaki ilişkiyi hesaplayabilen bir ağdır.

İnsanların davranışlarında, konuşma ve düşüncelerinde de bu yapıdaki gibi önceden hafızada olan bilgi tıpkı bir gizli katmanda yeni veri ile döngüye girerek işlenir. Bu işlemi yaparken kullanılan matematiksel formül şu şekildedir: $ht=\phi(Wxt+Uht-1)$

ht, t anındaki gizli katmanın sonucudur. xt girdisi W ağırlığı ile çarpılır. Daha sonra t-1 anında, içerik ünitesinde tutulan h(t-1) değeri U ağırlığı ile çarpılır ve Wxt ile toplanır. W ve U değerleri girdi ile katman arasındaki ağırlıklardır. Burada ağırlık matrisi önceki ve şimdiki verinin hangisinin sonuca etkisi daha çok veya az ise ona göre değerler alır. Bu işlemler sonucunda oluşan hata hesaplanır ve geri yayılım (backpropagation) ile yeni ağırlık değerleri tekrar düzenlenir. Backprop işlemi hata yeterince minimize edilene kadar devam eder.

Wxt + Uh(t-1) toplamı sigmoid, tanh gibi aktivasyon fonksiyonuna sokulur. Böylece çok büyük veya çok küçük değerler mantıklı bir aralığa alınır. Bu şekilde non-lineerlik de sağlanmış olur.

3.1.4.1. BPTT

Yinelenen ağların amacı sıralı girdileri doğru bir şekilde sınıflandırmak diyebiliriz. Bu işlemleri yapabilmek için hatanın backpropunu ve gradient descentini kullanırız. Backprop, ileri beslemeli ağlarda sonda çıktıdaki hatayı geriye hatanın türevini ağırlıklara dağıtılarak yapılır. Bu türev kullanılarak öğrenme katsayısı, gradient descent düzenlenerek hatayı düşürecek şekilde ağırlıklar düzenlenir.

RNN için kullanılan yöntem ise BPTT diye bilinen zamana bağlı sıralı bir dizi hesaplamanın tümü için backprop uygulamasıdır. Yapay ağlar bir dizi fonksiyonu iç içe f(h(g(x))) şeklinde kullanır. Buraya zamana bağlı değişken eklendiğinde türev işlemi zincir kuralı ile çözümlenebilir.

3.1.4.2. Geri Yayılım (Backpropagation)

Geri Yayılım yapay sinir ağı modelinin eğitilmesi sırasında kullanılan bir optimizasyon algoritmasıdır. Optimizasyon algoritmaları, modelin kayıp fonksiyonunu (loss function) minimize etmek veya en aza indirmek için modelin iç parametrelerini (weight, bias) nasıl güncelleyeceğini belirler.

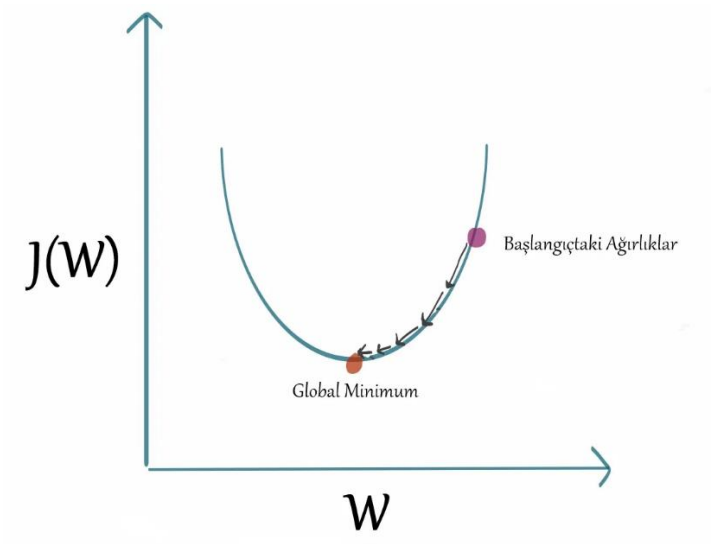
Geri yayılım, bu hatanın ağı geriye doğru nasıl yayıldığını hesaplayarak çalışır.

İlk olarak veri noktaları ağın girişine verilir ve ardından ağın son katmanındaki çıkış üretilir. Çıkış ile gerçek değerler arasındaki uyumsuzluğu ölçen bir kayıp fonksiyonu hesaplanır. Bu

kayıp fonksiyonu ağın tahminlerinin ne kadar yanlış olduğunu ölçer. Ardından bu hata geriye doğru katmanlardan başlayarak ağın içine yayılır. Bu yayılma işlemi her katmandaki ağırlıkların ve biasların katkısını hesaplar. Her katmandaki ağırlıkların ve biasların katkısı hesaplandıktan sonra, bu bilgiler kullanılarak ağırlıklar ve biaslar güncellenir. Genellikle SGD gibi bir optimizasyon algoritması ağırlıkları ve biasları güncellemek için kullanılır. Bu süreç eğitim verilerinin tamamı veya bir veri yığını(batch) için tekrarlanır. Birden fazla yineleme(epoch) boyunca ağın ağırlıkları ve biasları eğitilir. Geri yayılım yapay sinir ağlarının temelini oluşturur. Ağın tahminlerini gerçek değerlere yaklaştırmak için her yinelemede ağırlıkların ve biasların güncellenmesi modelin daha iyi bir şekilde verileri öğrenmesini sağlar. Bu nedenle geri yayılım derin öğrenme ve yapay sinir ağı modellerinin eğitimi için temel bileşendir.

3.1.4.3. Gradient descent

Gradient Descent algoritması en popüler optimizasyon algoritmalarından birisidir. Bu algoritmanın amacı, fonksiyona ait parametreleri devamlı güncelleyerek fonksiyonun en minimum değerine ulaşmaktır.



Şekil 3.2 Gradient Descentin temsili gösterimi

Dereceli azalma, bir fonksiyon üzerinde rastgele bir noktadan başlayan ve o işlevin en düşük noktasına ulaşıncaya kadar eğiminde adım adım ilerleyen yinelemeli bir algoritmadır.

3.1.4.4. Maliyet fonksiyonu

Genellikle kayıp fonksiyonlarının toplamını veya ortalamasını ifade eder ve tüm eğitim veri seti üzerindeki modelin genel performansını değerlendirmek için kullanılır. Maliyet fonksiyonu, modelin tüm veri seti üzerindeki toplam hatasını veya maliyetini ölçer. Örneğin, bir eğitim veri setinde n adet örnek varsa, maliyet fonksiyonu tüm bu örneklerin kayıp fonksiyonlarının ortalamasını alabilir.

3.1.4.5. Kayıp fonksiyonu

Bir tekil örnek veya veri noktası için modelin tahmin ettiği değer ile gerçek değer arasındaki hatayı ölçen bir fonksiyondur. Kayıp fonksiyonu, modelin bir veri noktası üzerindeki performansını değerlendirmemize olanak tanır.

3.1.4.6. Öğrenme oranı

Problem için model üzerinde optimum değeri arama süreci sırasında modelin öğrenme adım büyüklüğü ve bunun sonucunda modelde yapılan değişiklik miktarına "öğrenme oranı" denir ve minimum hatayı veren problemde daha iyi bir performans elde etmek için kullanılan belki de en önemli hiperparametreyi sağlar.

Bu öğrenme hızının düşük veya fazla olmasının bazı avantaj ve dezavantajları olabilmektedir. Çok büyük olması doğruluktan sapmaya çok küçük olması ise birbirine çok yakın değerler elde ederek özellikle büyük veri setlerinde gereksiz iterasyona sebep olabilir.

3.1.4.7. Aktivasyon fonksiyonu

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 işlemi uygulanır. Daha sonra bu, nihai çıkış ya da bir başka katmanın girişi olur.

Eğer aktivasyon fonksiyonu uygulanmazsa çıkış sinyali basit bir doğrusal fonksiyon olur. Doğrusal fonksiyonlar yalnızca tek dereceli polinomlardır. Aktivasyon fonksiyonu kullanılmayan bir sinir ağı sınırlı öğrenme gücüne sahip bir doğrusal bağlanım (linear regression) gibi davranır. Ama sinir ağının doğrusal olmayan durumları da öğrenmesini istenilir. Çünkü sinir ağımıza öğrenmesi için görüntü, video, yazı ve ses gibi karmaşık gerçek dünya bilgileri verilebilir. Çok katmanlı derin sinir ağları bu sayede verilerden anlamlı özellikleri öğrenebilir.

Tablo 3.1. Aktivasyon Fonksiyonları ve Denklemleri

AKTİVASYON FONKSİYON	DENKLEM	ARALIK
Doğrusal Fonksiyon	$f(x) = x$	$(-\infty, \infty)$
Basamak Fonksiyonu	$f(x) = \begin{cases} 0 \; için \; x < 0 \\ 1 \; için \; x \geq 0 \end{cases}$	$\{0, 1\}$
Sigmoid Fonksiyon	$f(x) = \sigma(x) = \frac{1}{1 + e^{-x}}$	$(0, 1)$
Hiperbolik Tanjant Fonksiyonu	$f(x) = \tanh(x) = \frac{(e^{x} - e^{-x})}{(e^{x} + e^{-x})}$	$(-1, 1)$
ReLU	$f(x) = \begin{cases} 0 \; için \; x < 0 \\ x \; için \; x \geq 0 \end{cases}$	$[0, \infty)$
Leaky (Sızıntı) ReLU	$f(x) = \begin{cases} 0.01 \; için \; x < 0 \\ x \; için \; x \geq 0 \end{cases}$	$(-\infty, \infty)$
Swish Fonksiyonu	$f(x) = 2x\sigma(\beta x) = \begin{cases} \beta = 0 \; için \; f(x) = x \\ \beta \to \infty \; için \; f(x) = 2\max(0, x) \end{cases}$	$(-\infty, \infty)$

Birden fazla dereceye sahip olan fonksiyonlara doğrusal olmayan fonksiyonlar denir. Yapay sinir ağları, evrensel fonksiyon yakınsayıcıları olarak tasarlanmış ve bu hedefte çalışması istenmektedir. Bu herhangi bir fonksiyonu hesaplayabilip öğrenme yetisine sahip olmaları gerektiği anlamına gelmektedir. Doğrusal olmayan aktivasyon fonksiyonları sayesinde ağların daha güçlü öğrenmesi sağlanabilir. Basamak (step) fonksiyonu, doğrusal (linear) fonksiyon, sigmoid fonksiyonu, hiperbolik tanjant fonksiyonu, ReLU fonksiyonu, Sızıntı (Leaky) ReLU fonksiyonu, Softmax fonksiyonu ve Swish (A Self-Gated/Kendinden Geçitli) fonksiyonu aktivasyon fonksiyonları çeşitleridir.

3.1.5. Kaybolan gradyan problemi (vanishing gradient problem)

Öğrenme eylemini sağlayan yapının yani gradyan fonksiyonunun işlevsizleşmesine vanishing gradient problemi denir. Girdi değerlerinde büyük değişimler olsa da bunun çıktıdaki karşılığı çok küçük değişimler oluyor. Bu sıçramalar kümülatif gittiği için de katman sayısı arttıkça gradyan fonksiyonu 0'a doğru yakınsar ve kaybolan gradyan problemi ortaya çıkar.

Daha basit anlatımıyla bu sorun şu şekilde açıklanabilir: Aktivasyon fonksiyonları sayesinde girdimizi belirli bir aralığa indirgeyebiliriz. Bu aralık genelde -1 ve 1 veya 0 ve 1 aralığıdır. Ufak bir alana indirgediğimiz için girdimizdeki büyük bir değişim aktivasyon fonksiyonunda o kadar büyük bir değişime yol açmayabilir. Dolayısıyla türevi de küçük olur. Eğer türevi çok küçükse, o katman yeteri kadar öğrenemez.

RNN'de çok erken aşamalarda dahi bu durum gerçekleşebilir. Bu katmanlar öğrenmediği için, RNN'ler daha uzun metinlerde gördüklerini unutabilir ve böylece kısa süreli bir hafızaya sahip olurlar.

Bu sorundan kaçınmak için aktivasyon fonksiyonu olarak Sigmoid yerine ReLU, Sızıntı ReLU gibi değişimin sabit olduğu yani türevin sıfıra yakınlaşmadığı aktivasyon fonksiyonları kullanılabilir. Bu sayede aktivasyon fonksiyonuna gelen çok büyük veya çok küçük değerler, türevlenebilir noktalarda yer almış olur ve modele etki etme kabiliyetlerini kaybetmezler.

Daha da önemlisi LSTM'ler ve GRU'lar kaybolan gradyan problemine bir çözüm olarak oluşturulmuştur ve bilgi akışını düzenleyebilecek geçitler (gates) adı verilen dahili mekanizmalara sahiptirler.

3.1.6. LSTM

Sepp Hochreiter ve Juergen Schmidhuber 1997 yılında kaybolan gradyan sorununu (vanishing gradient problem) problemini çözmek için LSTM'i geliştirdiler. Daha sonra birçok kişinin katkısıyla düzenlenen ve popülerleşen LSTM şu anda geniş bir kullanım alanına sahiptir.

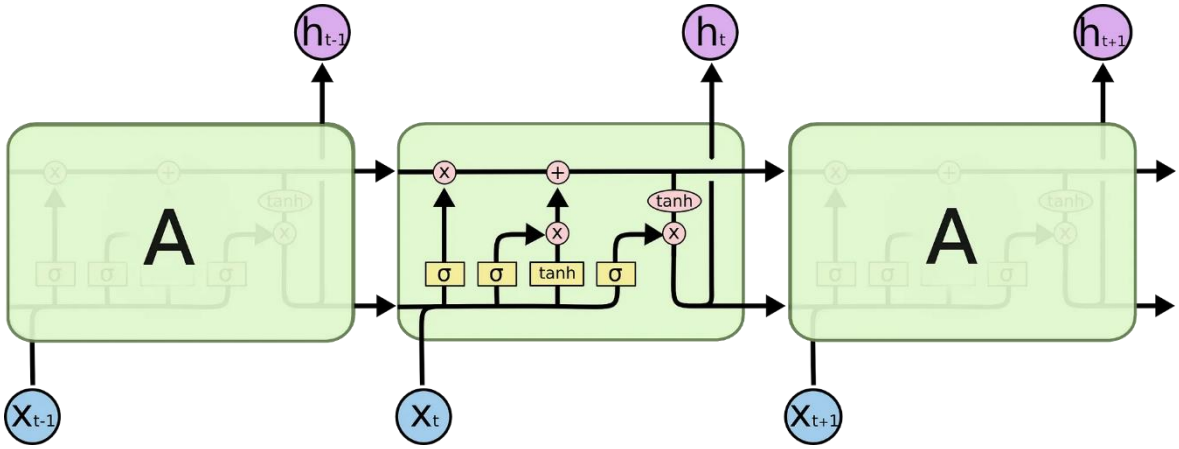
LSTM backprop'ta farklı zaman ve katmanlardan gelen hata değerini korumaya yarıyor. Daha sabit bir hata değeri sağlayarak yinelenen ağların öğrenme adımlarının devam edebilmesini sağlamaktadır. Bunu sebep sonuç arasına yeni bir kanal açarak yapmaktadır.

LSTM yapısında tekrar eden modülün farkı tek bir nöral network katmanı yerine, özel bir şekilde bağlı 4 katman bulunmasıdır. Bu katmanlara kapı da denmektedir. Normal akışın dışında dışarıdan bilgi alan bir yapıdır. Bu bilgiler depolanabilir, hücreye yazılabilir, okunabilir.

Hücre neyi depolayacağını, ne zaman okumasına, yazmasına veya silmesine izin vereceğini kapılar sayesinde karar verir. Bu kapılarda bir ağ yapısı ve aktivasyon fonksiyonu bulunmaktadır. Aynı nöronlarda olduğu gibi gelen bilgiyi ağırlığına göre geçirir veya durdurur. Bu ağırlıklar yinelenen ağın öğrenmesi sırasında hesaplanır. Bu yapı ile hücre, veri alacak mı bırakacak mı silecek mi öğrenir.

LSTM'de üç adet kapı (gate) ve hücre durumu (cell state) vardır. Bu kapılar Unutma Kapısı (Forget Gate), Girdi Kapısı (Input Gate) ve Çıktı Kapısı'dır (Output Gate).

Unutma Kapısı, Hangi bilginin tutulacağı veya unutulacağına karar verir. Matematikte olduğu gibi, bir sayı 0 ile çarpılırsa ne kadar büyük olursa olsun sonuç 0 olur. Unutma kapısında da aynı mantıkla işlem yapılır. Unutmak için girdinin ağırlığına 0 verilir.



Şekil 3.3. LSTM'in Kapıları ve Çalışma Mantığının Temsili Gösterimi

Bir önceki gizli katmandan gelen bilgiler ve güncel bilgiler Sigmoid Fonksiyonundan geçer. 0'a ne kadar yakınsa o kadar unutulacak, 1'e ne kadar yakınsa o kadar tutulacak demektir.

Girdi kapısı, hücre durumunu güncellemek için kullanılır. Öncelikle unutma kapısında olduğu gibi Sigmoid fonksiyonu uygulanır, hangi bilginin tutulacağına karar verilir. Daha sonra ağı düzenlemek için Tanh fonksiyonu yardımıyla -1,1 arasına indirgenir ve çıkan iki sonuç çarpılır.

Hücre durumunun, hücre içerisindeki en önemli görevi bilgiyi taşımaktır. Taşınması gereken verileri alır ve hücre sonuna, oradan da diğer hücrelere taşır. Yani ağ üzerinde veri akışını hücre durumu yardımıyla sağlarız. İlk olarak unutma kapısından gelen sonuç ile bir önceki katmanın sonucu çarpılır. Daha sonra girdi kapısından gelen değer ile toplanır.

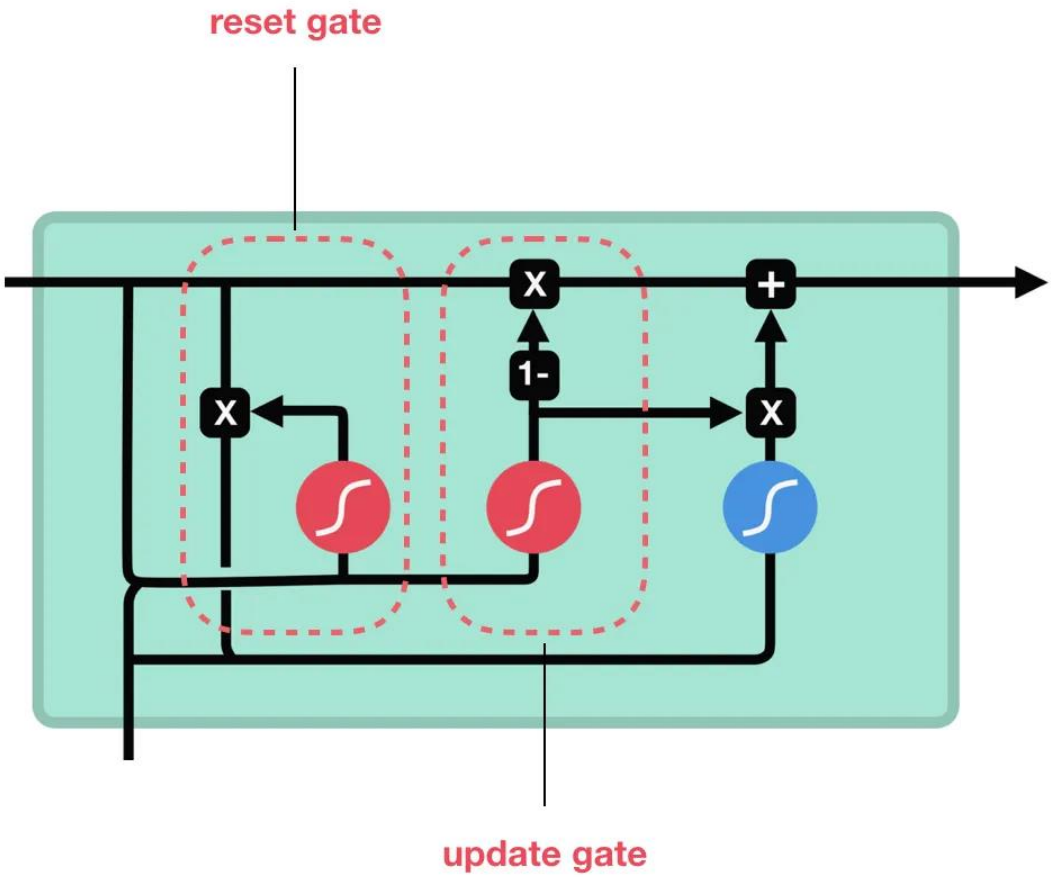
Çıktı kapısı, bir sonraki katmana gönderilecek değere karar verir. Bu değer, tahmin için kullanılır. Öncelikle bir önceki değer ile şu anki girdi Sigmoid fonksiyonundan geçer. Hücre durumundan gelen değer Tanh fonksiyonundan geçtikten sonra iki değer çarpılır ve bir sonraki katmana "Bir önceki değer" olarak gider ve hücre durumu ilerler.

3.1.7. GRU

Gated Recurrent Units (GRU'lar), 2014 yılında Kyunghyun Cho ve ekibi tarafından tanıtılan tekrarlayan sinir ağlarında bir geçiş mekanizmasıdır. GRU, belirli özellikleri girdi olarak almak veya unutmak için bir geçiş mekanizmasına sahip olan Long Short-Term Memory (LSTM) gibidir, ancak bir bağlam vektörü veya çıktı kapısı bulunmaz, bu da LSTM'ye göre

daha az parametreye sahip olmasını sağlar. GRU'nun, polifonik müzik modelleme, konuşma sinyali modelleme ve doğal dil işleme gibi belirli görevlerdeki performansının LSTM'ye benzer olduğu bulunmuştur. GRU'lar, geçiş mekanizmasının genel olarak faydalı olduğunu gösterdi ve Bengio'nun ekibi, iki geçiş biriminden hangisinin daha iyi olduğuna dair net bir sonuca varamadı.

Sadece iki kapısı vardır bunlar sıfırlama kapısı (reset gate) ve güncelleme kapısı (update gate)'dir. Güncelleme kapısı bir LSTM'nin unutma ve girdi kapısına benzer şekilde hareket eder, hangi bilgilerin saklanacağına ve hangilerinin atılacağına ve hangi yeni bilgilerin ekleneceğine karar verir. Sıfırlama kapısı, önceki bilgilerin ne kadarının unutulacağına karar vermek için kullanılır.



Şekil 3.4. GRU'nun LSTM'den Farklı Olan Kapıları ve Çalışma Mantığının Temsili Gösterimi

3.1.8. Kelime Gömme (Word Embedding)

Word embedding, doğal dil işlemenin temel kavramlarındandır. Makine öğrenmesi algoritmaları ile model oluşturmak için, verilerin nümerik halde olması yani sayılardan

oluşması gerekmektedir. Bu yüzden diğer tiplerde bulunan veriler de nümerik hale çevrilmelidir. Kategorik veriler farklı şekillerde nümerik hale dönüştürülebilir. Aralarında büyüklük ilişkisi olan {“az”, “orta”, “çok”} değerleri {1, 2, 3} rakamları ile yine sıralı bir şekilde temsil edilebilir. Ya da elimizde haftanın günleri varsa bunlar da 1–7 arası numaralandırılabilir. Bu durumda da 1. ve 7. gün aslında art arda olmasına rağmen 7. gün 1. günden uzak bir değere sahip olur. Bu tür durumlarda kategorik değerler bir sinüs ya da cosinus dalgası üzerine oturtularak, değerler arasındaki mesafe daha gerçekçi bir hal alır.

Tabi ki kategorik verilerin hepsi sıralı olmak zorunda değildir. Elimizde film türleri varsa, {“korku”, “gerilim”, “romantik”, “komedi”} gibi, bunlar için sıralı sayılar verme şansımız yoktur. Bu durumda da, yani aralarında sıralı bir ilişki bulunmayan kategorik değerler ise one-hot encoding ile 0 ya da 1 ile temsil edilebilir.

Ancak bazı durumlarda bu kadar basit bir dönüşüm yeterli olmayabilir. Film türleri örneğindeki veri kümelenmek istenirse, korku ve gerilim türünden filmlerin tür bazında yakın olması gerekirken değerler arasındaki anlamsal ilişkiler için bir şey yapılmamıştır. Bu yöntemle cümleler temsil edilmeye çalışıldığında encoding’e benzer şekilde cümledeki her bir kelime için bir kolon oluşturulabilir ve sonra da kelimelerin geçme sıklığına göre de puan verilebilir. Ancak bu şekilde yapıldığında kelimeler arasındaki ilişki yakalanamaz ve her bir değer için yeni bir kolon oluşturulması gerekeceğinden, daha karmaşık problemler için bu yöntem pek uygulanabilir değildir.

Kelime gömmede ise kelimeler en fazla beraber bulundukları kelimeler ile yakın olacak şekilde konumlandırılır. Bu yakınlık farklı birçok kategoride olabilir. Örneğin “İngiltere” kelimesini ele alırsak ülke adı olarak diğer ülke isimleriyle yakın olmalı, aynı zamanda İngiltere’ye özgü olan ifadelere de. Bu sayede bir kelime gömme modeli oluştururken yüzlerce katmanlı bir yapı kurulabilir. Bir doğru üzerinde diğer ülke isimleriyle yakın olurken diğer doğru üzerinde futbol takımlarıyla yakın olabilir veya başkentler ile yakın olabilir. Eğer başkentiyle daha fazla anılıyorsa bu doğrudaki yakınlık daha fazla olur. Burada yakınlığı belirleyen şey ise yan yana geçme sıklığı yani frekansıdır.

3.1.9. Kodlayıcı-Kod çözücü (Encoder-Decoder) Mekanizması

Tekrarlayan sinir ağlarına sahip kodlayıcı-kod çözücü mimarisi, günümüzde NLP tabanlı sayısız görevi çözmek için etkili ve standart bir yaklaşım haline gelmiştir. Kodlayıcı-kod çözücü model, diziden diziye tahmin problemi veya metinler (kelime dizisi), görüntüler

(görüntü dizisi) gibi zorlu dizi bazlı girdilerde birçok ayrıntılı tahmin sağlamak için yinelemeli sinir ağını organize etmenin bir yoludur.

Kodlayıcı, verilen giriş verilerinden elde edilen veya özellikleri çıkaran yani kodlayan bir tür ağdır. Giriş sırasını okur ve bilgiyi dahili durum vektörleri veya bağlam vektöründe özetler. LSTM başlığında bahsedilen gizli durum ve hücre durum vektörleri bunlardır. Bu bağlam vektörü, kod çözücünün doğru tahminler yapmasına yardımcı olmak için tüm girdi öğelerine ilişkin tüm bilgileri içermeyi amaçlar. Ağın gizli ve hücre durumu kod çözücüye girdi olarak iletilir.

Kod çözücü, kodlayıcıdan elde edilen bağlam vektörünü 'kod çözen', yorumlar. Kodlayıcının son hücresinin bağlam vektörü, kod çözücü ağının ilk hücresine girdidir. Bu başlangıç durumlarını kullanarak kod çözücü, çıktı dizisini oluşturmaya başlar ve bu çıktılar, gelecekteki tahminler için de dikkate alınır. Her biri bir zaman adımında (t) bir çıktı tahmin eden birkaç LSTM biriminden oluşan bir yığın vardır. Her bir tekrarlayan birim, önceki birimden bir gizli durumu kabul eder ve bir çıktı ile kendi gizli durumunu (hidden state) üretir ve bu durumu ağın ileri kısımlarına iletir.

Bu ağın ana dezavantajlarından biri, uzun anlamsal cümlelerden güçlü bağlamsal ilişkiler çıkaramamasıdır; yani, uzun bir metin parçasının alt dizilimlerinde bazı bağlamlar veya ilişkiler varsa, temel bir seq2seq (sıra sıralı) modeli bu bağlamları tanımlayamaz ve bu nedenle, modelimizin performansını bir miktar düşürür ve sonunda doğruluğu azaltır.

3.1.9.1. Dikkat (Attention) Mekanizması

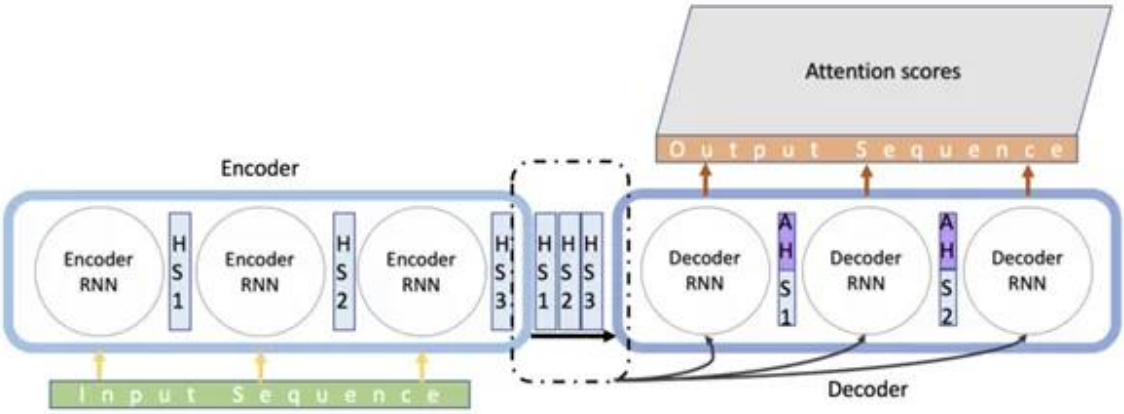
Dikkat, bu sınırlamayı ele alan sıra sıralı modellerin mevcut ağına bir yükseltmedir. Basit bir şekilde 'dikkat' olarak adlandırılmasının nedeni, dizilerdeki önemi elde edebilme yeteneğidir.

Öncelikle, kodlayıcıdan kod çözücüye daha ağırlıklı veya daha belirgin bir bağlam sağlayarak ve çözümleyicinin her bir zaman adımında çıktı dizisindeki çıktıları tahmin ederken sonraki kodlama ağına aslında daha fazla 'dikkat' vermesi gereken yerleri yorumlayabileceği bir öğrenme mekanizması sağlayarak çalışır.

Dikkat mekanizması kullanılarak, mevcut kodlayıcı-kod çözücü mimarisini sabit-kısa uzunlukta içsel metin temsilinden kurtarma fikri düşünülebilir. Bu, kodlayıcı LSTM ağından, giriş dizisinin her adımından belirli bir düzeyde önem taşıyan ara çıktıları koruyarak ve aynı zamanda modeli bu ara öğelere seçici dikkat göstermeyi öğrenmek ve bunları çıktı dizisindeki öğelerle ilişkilendirmek üzere eğiterek gerçekleştirilir.

Daha basit tabirle açıklamak gerekirse, giriş dizisindeki birkaç seçici öğe nedeniyle, çıktı dizisi koşullu hale gelir, yani birkaç ağırlıklı kısıtlama ile birlikte gelir. Bu koşullar, dikkat çeken ve dolayısıyla nihayetinde eğitilen ve istenen sonuçları tahmin eden bağlamlardır.

Dikkat tabanlı sıra sıralı model, iyi bir hesaplama kaynakları gücü gerektirir, ancak sonuçlar geleneksel sıra sıralı modele kıyasla oldukça iyidir. Ayrıca, modelin çıktı dizisini tahmin ederken giriş dizisine nasıl dikkat gösterdiğini de gösterebilir. Bu, modelin belirli giriş-çıkış çiftleri için neyi ve ne derece dikkate aldığını anlamaya ve teşhis etmeye yardımcı olabilir.



Şekil 3.5. Encoder ve Decoder Mekanizmasının Attention ile Kullanımının Temsili Gösterimi

Giriş dizisini daha ileri taşımak için tek bir sabit bağlam vektörüne kodlamak yerine, dikkat modeli farklı bir yaklaşım dener. Bu model, her çıktı zaman adımı için özel olarak seçici olarak filtrelenmiş bir bağlam vektörü geliştirmeye çalışır, böylece o ilgili filtrelenmiş kelimelere odaklanabilir ve buna göre, kod çözücü modelimizi tam dizilerle ve özellikle bu filtrelenmiş kelimelerle eğiterek tahminler elde edebilir.

3.2 Yöntem

3.2.1. Çalışma Ortamı ve Kullanılan Kütüphaneler

3.2.1.1. Model tarafı

Model, makine öğrenmesi ve veri bilimi için geniş bir kütüphane ve framework yelpazesine sahip olması ve sade ve okunabilir sözdizimine sahip olması nedeniyle, geniş ve aktif bir topluluğa sahip olan, diğer diller ve araçlarla kolayca entegre edilebilen, farklı işletim sistemlerinde çalışabilen Python programlama dilinde geliştirilmiştir.

Geliştirilen model, derin öğrenme modellerinin eğitilmesi gibi yoğun hesaplama gerektiren işlemleri hızlandırarak ve kullanıcıların donanım maliyetlerinden tasarruf etmelerini sağlayarak GPU (Grafik İşleme Birimi) ve TPU (Tensor İşleme Birimi) kullanmalarına

olanak tanıyan Google Colab ortamında hazırlanmıştır. Google Drive ile entegrelidir ve proje bulutta saklanabilir. Makine öğrenmesi ve veri bilimi için yaygın olarak kullanılan TensorFlow, Keras, PyTorch, NumPy, Pandas, Matplotlib gibi birçok Python kütüphanesi ile önceden yüklü gelir.

Python içerisinde sade ve basit bir şekilde kullanılabilen ve yazılım geliştiricilerin makine öğrenmesi, görüntü işleme, derin öğrenme, görüntü yorumlama gibi konularda sıklıkla kullandığı Tensorflow, Keras, Pandas, Numpy, Snscrape, Scikit-Learn, NLTK, Matplotlib gibi açık kaynaklı birçok kütüphane barındırmaktadır. Bu projede kullanılan bazı kütüphaneler şunlardır:

- **gdown:** Google Drive'dan dosya indirmek için kullanılır. gdown ile Google Drive linklerini kullanarak dosya indirilebilir.
- **tarfile:** Tar dosyalarını okumak ve yazmak için kullanılır. Arşiv dosyalarını çıkarmak veya oluşturmak için kullanılır.
- **os:** İşletim sistemi ile etkileşim kurmak için kullanılır. Dosya ve dizin işlemleri için yaygın olarak kullanılır.
- **pickle:** Python nesnelerini seri hale getirmek ve seri hallerinden çıkarmak için kullanılır. Nesnelerin dosyalara kaydedilip tekrar yüklenmesini sağlar.
- **PIL:** Görüntü işleme için kullanılır. Image modülü, görüntü dosyalarını açmak, işlemek ve kaydetmek için kullanılır.
- **matplotlib:** Veri görselleştirme için kullanılır. Grafikler ve çizimler oluşturmak için yaygın olarak kullanılır.
- **pandas:** Veri manipülasyonu ve analizi için kullanılır. Verileri yüklemek, işlemek ve analiz etmek için DataFrame veri yapısını sağlar.
- **numpy:** Sayısal hesaplamalar için kullanılır. Dizilerle (arrays) ve matrislerle (matrices) çalışmak için fonksiyonlar sağlar.
- **cv2 (OpenCV)**: Bilgisayarlı görü için kullanılan bir kütüphane. Görüntü ve video işleme, analiz ve manipülasyon için kullanılır.
- **tensorflow**: Makine öğrenmesi ve derin öğrenme modelleri oluşturmak için kullanılan bir açık kaynaklı kütüphane. TensorFlow ile sinir ağları ve diğer makine öğrenmesi algoritmaları uygulanabilir.

3.2.1.2. REST API tarafı

Modelin geliştirilmesinin ardından kullanılabilmesi ve arayüz tarafı ile iletişime geçilebilmesi için REST API oluşturulmuştur. REST API tarafında da Python programlama

dili kullanılmıştır ve Google Colab ortamında geliştirilmiştir. REST API tarafında kullanılan kütüphaneler ve çerçeveler(framework) şunlardır:

- **FastAPI:** FastAPI, modern, hızlı (yüksek performanslı) ve kullanıcı dostu bir web çerçevesidir. Web API'leri oluşturmak için kullanılır ve Python 3.6+ ile uyumludur. FastAPI, otomatik olarak API belgeleri oluşturma, veri doğrulama ve tür denetimi gibi özellikler sunar.
- **pydantic:** Pydantic, Python verilerini modellemek ve doğrulamak için kullanılan bir kütüphanedir. BaseModel, veri doğrulama ve tür denetimi yapmak için temel bir sınıftır. FastAPI ile kullanıldığında, API isteklerinin ve yanıtlarının doğrulamasını sağlar.
- **io:** Python'un standart kütüphanesinin bir parçasıdır ve giriş/çıkış işlemleri için kullanılır. io modülü, dosya okuma/yazma işlemleri ve bellek içi (in-memory) dosya nesneleriyle çalışmak için çeşitli sınıflar sağlar.
- **base64:** Python'un standart kütüphanesinin bir parçasıdır ve verileri Base64 formatında kodlamak ve kodunu çözmek için kullanılır. Base64, ikili verilerin (örneğin, dosyaların veya görüntülerin) metin tabanlı bir formata dönüştürülmesini sağlar. Bu, verilerin güvenli bir şekilde iletilmesi veya saklanması için kullanılabilir.
- **pyngrok:** Ngrok, yerel bir sunucuyu herkese açık bir URL üzerinden erişilebilir hale getiren bir araçtır. pyngrok kütüphanesi, Ngrok'un Python için olan paketidir. Bu kütüphane, yerel geliştirme ortamında çalışan bir uygulamanın internet üzerinden erişilebilir olmasını sağlamak için tünel oluşturur. Bu, özellikle web uygulamalarını test etmek veya paylaşmak için kullanışlıdır.
- **nest_asyncio:** Python'da, özellikle Jupyter Notebook gibi asenkron döngülerin (async loops) birden fazla kez çalıştırılmasını gerektiren ortamlarda asenkron programlama yapmak bazen zordur. nest_asyncio kütüphanesi, mevcut bir asenkron döngüyü yeniden kullanılabilir hale getirir, yani bir asenkron döngü içinde başka bir asenkron döngü çalıştırmanıza izin verir. Bu, genellikle Jupyter Notebook'ta FastAPI gibi asenkron uygulamaları çalıştırırken kullanılır.
- **uvicorn:** Uvicorn, hızlı bir ASGI sunucusudur. FastAPI ve Starlette gibi ASGI uygulamalarını çalıştırmak için kullanılır. Uvicorn, yüksek performanslı bir web sunucusudur ve asenkron programlama ile uyumludur, bu da hızlı ve verimli web uygulamaları oluşturmayı sağlar.

3.2.1.3. GUI Tarafı

GUI için geliştiriciler tarafından web uygulamaları için kullanıcı arayüzleri oluşturmak amacıyla kullanılan güçlü bir JavaScript kütüphanesi olan React JS kullanılmıştır. İlk olarak 2011 yılında Facebook tarafından web sitelerinin performansını artırmak için oluşturulmuştur ve o zamandan beri dünya çapındaki geliştiriciler tarafından yaygın bir şekilde benimsenmiştir. React JS'nin en önemli avantajlarından biri, tüm sayfayı yeniden yüklemeye gerek kalmadan kullanıcı arayüzünü verimli bir şekilde güncellemesine olanak tanıyan sanal DOM'udur. Ayrıca React JS, karmaşık kod tabanlarını basitleştirmeye ve geliştirmeyi daha hızlı ve verimli hale getirmeye yardımcı olabilecek yeniden kullanılabilir bileşenler oluşturmayı kolaylaştırır. React JS'in içinde kullanılan bazı kütüphaneler şunlardır:

- **axios:** HTTP isteklerini (GET, POST, PUT, DELETE gibi) gerçekleştirmek için kullanılan bir kütüphanedir. RESTful API'lerle iletişim kurmak için yaygın olarak kullanılır.
- **FileBase64:** Dosyaları Base64 formatına dönüştürmek için kullanılan bir React bileşenidir. Dosya yükleme işlemlerinde yaygın olarak kullanılır.

3.2.2. Çalışmada Kullanılan Veri ve DataFrame

Çalışmada veri kaynağı olarak Indiana Üniversitesi'nin PNG görsel formatındaki Chest-X-Ray görselleri kullanılmıştır. 3850 hastadan alınan bulgular ve bu hastaların 7000 adet X-ray görseli kullanılmıştır. Veri setindeki görseller ve bu görsellere ait bulgular Indiana Üniversitesi'nin Chest X-Rays veri setinden alınmıştır. Bu hastaların 3200 tanesinin 2 adet X-ray görseli bulunmaktadır. Bunun dışında tek X-ray görseli bulunan hastalar ve 3 adet X-ray görseli bulunan hastalar da bulunmaktadır. Görsellere ait bulgulardaki cümleler yardımcı fiillerin kaldırılması gibi ön işlemlere tabi tutulmuştur. Bu görseller ve bulgular Python'da Pandas kütüphanesi kullanılarak bir DataFrame'de toplanmıştır.

Bu görsellerin özellik vektörlerinin çıkarılması için önceden eğitilmiş CheXNet model ağırlıklarını ve DenseNet121 modeli kullanılmıştır. Model 14 farklı hastalığı tespit edebilecek ve 224x224 piksel boyutundaki 3 kanallı renkli görüntülerle yani RGB görüntülerle eğitilecek şekilde oluşturulmuştur. Bu yeni modelde DenseNet121 modelinin giriş ve çıkış katmanlarında (son 2 katman) CheXNet modelinin ağırlıkları kullanılmıştır. Çıkış katmanlarının çıktısını almak, modelin ara katmanlarında öğrenilen özellikleri

kullanmamızı sağlar. Çıkarılan bu özellikler Pandas kütüphanesiyle oluşturulan DataFrame'e eklenmiştir ve bu DataFrame .pkl uzantılı dosya olarak kaydedilmiştir.

Kaydedilen dosya projede içe aktarılmıştır. Bu DataFrame 8 kolona sahiptir. Bu kolonlar sırasıyla şu şekildedir.

- **patient_id:** Hasta'nın id numarasıdır.
- **image1:** İlk resmin dizinini ve dosya adını belirtir.
- **image2:** İkinci resmin dizinini ve dosya adını belirtir.
- **findings:** Bulguların string tipinde toplanmış halidir.
- **image_features:** Görsellerin özellik vektörlerinin dizi tipindeki halidir.
- **findings_total:** Başına <start> etiketi ve sonuna <end> etiketi getirilmiş string tipindeki bulgulardır.
- **dec_ip:** Başına <start> etiketi getirilmiş string tipinde toplanmış bulgulardır.
- **dec_op:** Sonuna <end> etiketi getirilmiş string tipindeki bulgulardır.

Önceki paragraflarda bahsedildiği üzere iki adet görsel için kolon bulunmaktadır. Tek görsele sahip hasta verileri için bu tek görseli iki kolonda da kullanarak çözüme gidilmiştir. Ayrıca 3 görsele sahip hasta verileri için ise ilk ve sonuncu görsel ele alınmıştır. İleriki aşamalarda test setinin dışındaki verilerle çalışırken de aynı görsel iki kere kullanılarak çözüme gidilmiştir.

Tablo 3.2. Kullanılan Veri Setinin Sütunlarıyla Beraber İlk 5 Satırı

index	patient_id	image1	image2	findings	image_features	findings_total	dec_ip	dec_op
0	CXR3556	/content/png/CXR3556_IM-1741-1001-0001.png	/content/png/CXR3556_IM-1741-1001-0002.png	the lungs are clear. there is no pleural effusion or pneumothora. there has been a sternotomy. the heart is not enlarged. some atherosclerotic changes of the aorta are seen. the skeletal structures are normal.	[[4.15828603e-04 1.57042092e-03 2.88839126e-03 ... 8.03524554e-01 8.32249701e-01 6.63665593e-01]]	<start> the lungs are clear. there is no pleural effusion or pneumothora. there has been a sternotomy. the heart is not enlarged. some atherosclerotic changes of the aorta are seen. the skeletal structures are normal. <end>	<start> the lungs are clear. there is no pleural effusion or pneumothora. there has been a sternotomy. the heart is not enlarged. some atherosclerotic changes of the aorta are seen. the skeletal structures are normal.	the lungs are clear. there is no pleural effusion or pneumothora. there has been a sternotomy. the heart is not enlarged. some atherosclerotic changes of the aorta are seen. the skeletal structures are normal. <end>
2	CXR32	/content/png/CXR32_IM-1511-1001.png	/content/png/CXR32_IM-1511-4001.png	the heart is normal in size. the mediastinum is unremarkable. mild blunting of right costophrenic . the lungs are otherwise grossly clear.	[[5.35454426e-04 1.96681428e-03 1.96952303e-03 ... 8.31006527e-01 8.16827357e-01 6.81426048e-01]]	<start> the heart is normal in size. the mediastinum is unremarkable. mild blunting of right costophrenic . the lungs are otherwise grossly clear. <end>	<start> the heart is normal in size. the mediastinum is unremarkable. mild blunting of right costophrenic . the lungs are otherwise grossly clear.	the heart is normal in size. the mediastinum is unremarkable. mild blunting of right costophrenic . the lungs are otherwise grossly clear. <end>
4	CXR260	/content/png/CXR260_IM-1090-1001.png	/content/png/CXR260_IM-1090-2001.png	lungs are clear bilaterally. cardiac and mediastinal silhouettes are normal. pulmonary vasculature is normal. no pneumothora or pleural effusion. no acute bony abnormality. there is a stable the electronic device any left anterior chest wall. there are advanced degenerative changes in the bilaterally. there is a mm lucency in the right humeral head with geographic a margins.	[[2.74563616e-04 1.88779249e-03 1.47982407e-03 ... 7.70237982e-01 8.53755534e-01 6.31056693e-01]]	<start> lungs are clear bilaterally. cardiac and mediastinal silhouettes are normal. pulmonary vasculature is normal. no pneumothora or pleural effusion. no acute bony abnormality. there is a stable the electronic device any left anterior chest wall. there are advanced degenerative changes in the bilaterally. there is a mm lucency in the right humeral head with geographic a margins. <end>	<start> lungs are clear bilaterally. cardiac and mediastinal silhouettes are normal. pulmonary vasculature is normal. no pneumothora or pleural effusion. no acute bony abnormality. there is a stable the electronic device any left anterior chest wall. there are advanced degenerative changes in the bilaterally. there is a mm lucency in the right humeral head with geographic a margins.	lungs are clear bilaterally. cardiac and mediastinal silhouettes are normal. pulmonary vasculature is normal. no pneumothora or pleural effusion. no acute bony abnormality. there is a stable the electronic device any left anterior chest wall. there are advanced degenerative changes in the bilaterally. there is a mm lucency in the right humeral head with geographic a margins. <end>
5	CXR1301	/content/png/CXR1301_IM-0198-1001.png	/content/png/CXR1301_IM-0198-2001.png	heart size within normal limits. stable mediastinal and hilar contours. mediastinal calcifications suggest a previous granulomatous process. no focal alveolar consolidation. no definite pleural effusion seen. no typical findings of pulmonary edema.	[[5.87469724e-04 1.84481509e-03 1.87531253e-03 ... 8.41161013e-01 8.22205544e-01 6.82864606e-01]]	<start> heart size within normal limits. stable mediastinal and hilar contours. mediastinal calcifications suggest a previous granulomatous process. no focal alveolar consolidation. no definite pleural effusion seen. no typical findings of pulmonary edema. <end>	<start> heart size within normal limits. stable mediastinal and hilar contours. mediastinal calcifications suggest a previous granulomatous process. no focal alveolar consolidation. no definite pleural effusion seen. no typical findings of pulmonary edema.	heart size within normal limits. stable mediastinal and hilar contours. mediastinal calcifications suggest a previous granulomatous process. no focal alveolar consolidation. no definite pleural effusion seen. no typical findings of pulmonary edema. <end>
6	CXR1921	/content/png/CXR1921_IM-0598-1001.png	/content/png/CXR1921_IM-0598-2001.png	redemonstration of moderately-inflated lungs, consistent with copd and unchanged. atherosclerotic calcifications of the thoracic seen. no airspace disease, effusion or noncalcified nodule. normal heart size and mediastinum. visualized of the chest are within normal limits.	[[2.97473045e-04 1.42153469e-03 2.95857969e-03 ... 8.64852726e-01 8.42953920e-01 7.07368374e-01]]	<start> redemonstration of moderately-inflated lungs, consistent with copd and unchanged. atherosclerotic calcifications of the thoracic seen. no airspace disease, effusion or noncalcified nodule. normal heart size and mediastinum. visualized of the chest are within normal limits. <end>	<start> redemonstration of moderately-inflated lungs, consistent with copd and unchanged. atherosclerotic calcifications of the thoracic seen. no airspace disease, effusion or noncalcified nodule. normal heart size and mediastinum. visualized of the chest are within normal limits.	redemonstration of moderately-inflated lungs, consistent with copd and unchanged. atherosclerotic calcifications of the thoracic seen. no airspace disease, effusion or noncalcified nodule. normal heart size and mediastinum. visualized of the chest are within normal limits. <end>

DataFrame'de toplamda 3337 adet satır bulunmaktadır. Bu DataFrame'in %20'si test seti olarak ayrılmıştır ve %80'i train seti olarak ayrılmıştır. Bu sayede train seti 2669 satırdan oluşmuştur ve test seti 668 satırdan oluşmuştur.

Train setinde kullanılan kelime dağarcığının boyutu 1826'dır. Bu, kelime dağarcığındaki toplam farklı kelime sayısını temsil eder. Bu kelimeler tokenize edildikten sonra anahtar(key) ve değer(value) ikililerine bölünmüştür.

3.2.3. Veri Ön İşleme

Veri train seti ve test seti olarak bölündükten sonra tokenize işlemi yapılmıştır. Özel karakterler, noktalama işaretleri, tab karakteri ve yeni satır karakteri metinlerden çıkarılmıştır. Ardından hem train setindeki metinler hem de test setindeki metinler diziye dönüştürülmüştür. Bundan sonra maksimum cümle uzunluğuna kadar bütün metinler 0 ile doldurulmuştur. Bu işleme padding denir. Bu sayede bütün metinler sabit uzunluktaki dizilere dönüştürülmüştür.

```
t1 = Tokenizer( filters='!"#$%&()*+,-/:;=?@[\\]^_`{|}~\t\n',oov_token='OOV')
t1.fit_on_texts(X_train[:,5])
vocab_size_imp = len(t1.word_index) + 1

dec_inp = t1.texts_to_sequences(X_train[:,6])

dec_inp = pad_sequences(dec_inp, maxlen=76, padding='post')

dec_inp_cv = t1.texts_to_sequences(X_test[:,6])

dec_inp_cv = pad_sequences(dec_inp_cv, maxlen=76, padding='post')

dec_op = t1.texts_to_sequences(X_train[:,7])

dec_op = pad_sequences(dec_op, maxlen=76, padding='post')

dec_op_cv = t1.texts_to_sequences(X_test[:,7])

dec_op_cv = pad_sequences(dec_op_cv, maxlen=76, padding='post')
```

Şekil 3.6. Bulgular İçin Uygulanan Tokenizer Yöntemi

3.2.4 Kodlayıcı-Kod Çözücü-Dikkat (Encoder-Decoder-Attention) Mekanizması

Gerekli veriler hazırlandıktan sonra modelin call fonksiyonuna verilir. Modelin call fonksiyonunda ilk adım, Encoder'in çalıştırılmasıdır. Encoder, giriş dizisini alır ve gizli durumlar (hidden states) üretir. Encoder için başlangıç gizli durumları sıfırlanır. Girdi dizisi, Dense katmanından geçirilerek işlenir ve encoder çıktısı üretilir.

Encoder'dan elde edilen çıktılar ve gizli durumlar, Decoder'in başlangıç durumları olarak kullanılır.

Decoder, hedef diziyi adım adım işler. Her adımda bir token işlenir ve bu süreç şu adımları içerir:

- Hedef diziden alınan girdi token'ı, gömme (embedding) katmanında sürekli vektörlere dönüştürülür.
- Kod çözücünün mevcut gizli durumu ve kodlayıcı çıktıları kullanılarak bir bağlam vektörü (context vector) hesaplanır. Bu bağlam vektörü, hangi kodlayıcı çıktısının daha önemli olduğunu belirler.
- Hesaplanan bağlam vektörü, gömme (embedding) katmanından gelen girdi vektörü ile birleştirilir ve GRU katmanına verilir.
- Birleştirilen vektör, GRU katmanına verilerek yeni bir gizli durum (hidden state) ve bir çıktı üretilir.
- GRU çıktısı, dense katmanından geçirilerek son çıktı elde edilir.
- Her adımda üretilen çıktı ve gizli durumlar saklanır.
- Decoder'in her adımda ürettiği çıktılar bir araya getirilir ve sonuç olarak döndürülür.

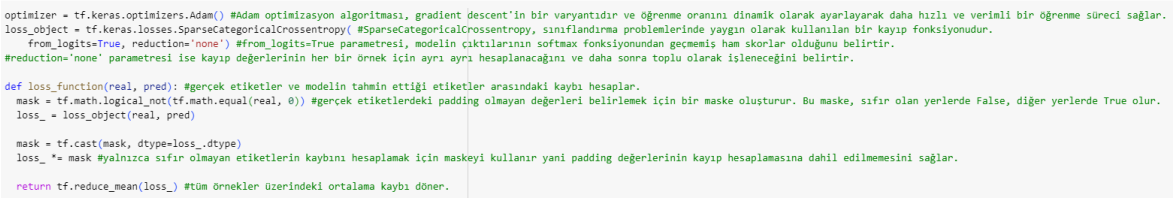
Genel olarak çalışma mantığını özetlemek gerekirse modelin call fonksiyonuna giriş ve hedef dizileri verilir, giriş dizisi encodera yani kodlayıcıya verilir, kodlayıcı gizli durumlar yani hidden state ve çıktılar üretir, hedef dizisi adım adım kod çözücüde işlenir, her adımda embedding yani gömme, dikkat yani attention, GRU ve dense katmanları kullanlarak çıktı üretilir, tüm çıktılar bir araya getirilir ve kod çözücünün ürettiği çıktılar modelin çıktısı olarak döndürülür.

3.2.5. Optimizasyon ve Kayıp Fonksiyonunu Tanımlama

Optimizasyon algoritmaları, modelin ağırlıklarını güncelleyerek kaybı minimize etmeye çalışır. Çalışmada Adam optimizasyon algoritması kullanılmıştır. Adam optimizasyon algoritması, gradient descent'in bir varyantıdır ve öğrenme oranını dinamik olarak ayarlayarak daha hızlı ve verimli bir öğrenme süreci sağlar.

Kayıp Fonksiyonu, modelin ne kadar iyi performans gösterdiğini ölçmek için kullanılır. Çalışmada SparseCategoricalCrossentropy kayıp fonksiyonu kullanılmıştır. SparseCategoricalCrossentropy, sınıflandırma problemlerinde yaygın olarak kullanılan bir kayıp fonksiyonudur. Bu fonksiyon, gerçek etiketler ile tahmin edilen etiketler arasındaki farkı hesaplar.

Özel bir fonksiyon oluşturularak bu kayıp fonksiyonu kullanılmış ve kullanılırken padding işlemi sırasında doldurulan değerlerin hesaba katılmaması sağlanmıştır.



Şekil 3.7. Optimizasyon ve Kayıp Fonksiyonunu Tanımlama

3.2.6. Model Ağırlıklarını Kaydetme ve Geri Çağırma (Callback)

Çalışmada TensorFlow/Keras kullanılarak bir modelin eğitim sürecinde belirli kontrol noktalarında (checkpoints) modelin ağırlıklarını kaydetmek için bir geri çağırma (callback) kullanılmıştır. Bu geri çağırmanın işlevi, eğitim sırasında en iyi performans gösteren modelin kaydedilmesini sağlamaktır.

ModelCheckpoint geri çağırması, modelin eğitim sürecinde belirli aralıklarla veya belirli koşullar sağlandığında modelin ağırlıklarını ve/veya mimarisini kaydeder. Doğrulama kaybının minimum olduğu modelin hem mimarisi hem de ağırlıkları kaydedilmiş ve kaydedilen ağırlıklar daha sonradan yüklenerek kullanılmıştır. Bu sayede modelin mimarisini ilgilendirmeyen her düzenlemede modeli eğitmek gerekmemiştir ve hem zamandan hem de donanımdan tasarruf edilmiştir.

```
model_checkpoint_callback = tf.keras.callbacks.ModelCheckpoint(
    filepath=checkpoint_filepath,
    save_weights_only=False,  #False olarak ayarlandığında, model
    monitor='val_loss', #Modelin doğrulama kaybı (val_loss) izlen
    verbose = 1, #Eğitim sırasında checkpoint kaydedildiğinde bil
    mode='min', #İzlenen metriğin minimum değerine göre modelin e
    save_best_only=True) #True olarak ayarlandığında, sadece izle
```

Şekil 3.8. Model Ağırlıklarını Kaydetme ve Geri Çağırma (Callback) Tanımlama

3.2.7 Beam Search ile Cümle Kurma

Beam Search, genellikle sıralama-sıralama (sequence-to-sequence) modellerinde, özellikle doğal dil işleme (NLP) görevlerinde, örneğin makine çevirisi veya metin oluşturma gibi uygulamalarda kullanılır.

Öncelikle başlangıç cümlesi kodlayıcıya (encoder) verilir ve kodlayıcıdan başlangıç durumu (initial state) ile kodlayıcı çıktısı (encoder output) alınır. Tahmin edilen cümlelerin başlangıç durumu <start> tokeni ile başlar. Beam Search, belirli bir genişlikte en olası cümleleri takip eder. Her adımda, en iyi olasılıklı tokenlar seçilir ve cümleye eklenir. '<end>' tokenına

ulaşıldığında cümle tamamlanmış olarak kabul edilir ve sonuç olarak, en olası cümle döndürülür. Bu yöntem ile karmaşık ve uzun dizilerde daha iyi ve daha çeşitli tahminler yapmak için sıklıkla kullanılmaktadır.

3.2.8. Görsellerin Özellik Vektörünü Çıkarma

Kullanıcıdan alınacak görsellerin özellik vektörlerinin çıkarılması için önceden eğitilmiş CheXNet model ağırlıklarını ve DenseNet121 modeli kullanılmıştır. Model 14 farklı hastalığı tespit edebilecek ve 224x224 piksel boyutundaki 3 kanallı renkli görüntülerle yani RGB görüntülerle eğitilecek şekilde oluşturulmuştur. Bu yeni modelde DenseNet121 modelinin giriş ve çıkış katmanlarında (son 2 katman) CheXNet modelinin ağırlıkları kullanılmıştır. Çıkış katmanlarının çıktısını almak, modelin ara katmanlarında öğrenilen özellikleri kullanmamızı sağlar.

Daha önce yazılan başlıklarda yazıldığı gibi eğitim verisinde kullanılan bulguların birçoğunda iki adet görsel kullanılmıştır, bu yüzden kullanıcının göndereceği tek görselin özellik vektörü kopyalanarak iki görsel gibi kullanılmış ve tahmin üretilmiştir.

3.2.9. FastAPI ve Ngrok ile REST API

FastAPI çerçevesi ile bir API oluşturulmuştur ve CORS ayarları düzenlenerek dışarıdan gelen istekleri karşılayabilmesi ve cevap gönderebilmesi sağlanmıştır.

Arayüz tarafından gelen base64 kodlu görüntü verisi decode edilerek kaydedilir ve dizini özellik vektörünü çıkarma işlemi uygulanan fonksiyona yönlendirilir. Özellik vektörünün çıkarılma işleminden sonra beam search kullanılacak fonksiyona yönlendirilir ve çıkan tahmin JSON tipinde arayüze gönderilir. Bu işlemler sırasında bir hatayla karşılaşılırsa yine JSON formatında hata mesajı gönderilir.

Ngrok sisteminin web sitesine kaydolunup ücretsiz şekilde alınan token kullanılarak oturum açılır ve böylece ngrok tüneli açılması için yetki alınmış olur.

Ngrok tüneli 9000 numaralı porta bağlandırılır ve FastAPI uygulaması internete açılmış olur. Ngrok tarafından gelen URL konsola yazdırılır, bu sayede arayüz tarafından iletilecek olan isteklerin hangi URL'ye gideceği belirlenmiş olur.

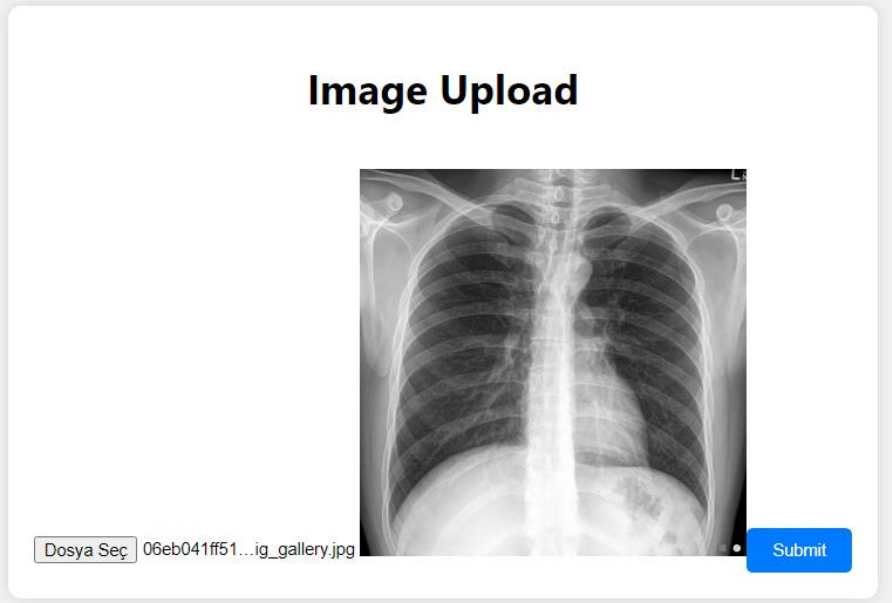
```
import nest_asyncio
from pyngrok import ngrok
import uvicorn

ngrok_tunnel = ngrok.connect(9000)
print('Public URL:', ngrok_tunnel.public_url)
nest_asyncio.apply()
uvicorn.run(app, host="127.0.0.1", port=9000)
```

```
Public URL: https://6994-34-173-73-198.ngrok-free.app
INFO:     Started server process [368]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://127.0.0.1:9000 (Press CTRL+C to quit)
INFO:     78.171.209.96:0 - "OPTIONS /predict HTTP/1.1" 200 OK
INFO:     78.171.209.96:0 - "POST /predict HTTP/1.1" 500 Internal Server Error
1/1 [==============================] - 0s 163ms/step
1/1 [==============================] - 0s 159ms/step
INFO:     78.171.209.96:0 - "POST /predict HTTP/1.1" 200 OK
1/1 [==============================] - 0s 163ms/step
1/1 [==============================] - 0s 179ms/step
INFO:     78.171.209.96:0 - "POST /predict HTTP/1.1" 200 OK
INFO:     Shutting down
INFO:     Waiting for application shutdown.
INFO:     Application shutdown complete.
INFO:     Finished server process [368]
```

Şekil 3.9. Ngrok ile Public URL Alınması ve İsteklerin Karşılanmasına Örnek

3.2.10. Arayüz İçin React JS Kullanımı

Bilgisayarın yerel ortamında React JS projesi oluşturulmuştur ve VS Code kaynak kodu düzenleyicisi ile proje açılmıştır. Projeye artık bu ortamda geliştirilmeye devam edilmiştir. REST API tarafına istek gönderilebilmesi, cevap alınabilmesi ve alınan cevapların işlenebilmesi için axios kütüphanesi kullanılmıştır. İstek, daha önceden Ngrok tarafından verilen URL'ye gönderilmiştir. Uygulamaya yüklenen görsellerin yüklenme ve base64 kodlu görüntü verisine dönüştürme işlemi için FileBase64 kullanılmıştır. Gelen cevap ekrana yazdırılmıştır. Bütün bu istek gönderme, cevap alma ve yazdırma işlemlerinin yönetimi için React JS'in sunmuş olduğu useState ile yönetilmiştir.



Şekil 3.10. React JS ile Oluşturulan Arayüz

4 BULGULAR VE TARTIŞMA

4.1. Model Epoch Sonuçları

Makine öğrenmesinde bir verisetinin tamamının bir kere ağdan geçiş yapmasına İngilizcede epoch denilmektedir. Model eğitiminin bir yinelemede kullanılan örnek kümesine batch ve bir toplu işteki örnek sayısına batch size denilmektedir.

Kayıp (loss) değeri modelin eğitim veri kümesi üzerinde hesaplanırken, doğrulama kaybı (validation loss) modelin doğrulama veri kümesi üzerinde hesaplanan kayıp değerdir. Kayıp değeri modelin eğitim sırasında ne kadar iyi öğrendiğini ve hedef değeri ne kadar iyi tahmin ettiğini gösterir. Doğrulama kaybı ise modelin eğitim sırasında eğitim veri kümesi dışında nasıl performans gösterdiğini değerlendirmek ve modelin genelleme yeteneğini ölçmek için kullanılır. Eğer kayıp düşerken doğrulama kaybı artıyorsa, bu modelin eğitim veri kümesine aşırı uyum sağladığını ve doğrulama veri kümesinde iyi performans göstermediğini gösterir. Çalışmada kullanılan modelde batch size boyutu 50 olarak ayarlanmıştır ve 40 epochta eğitilmiştir. Aşağıda her epochta yazdırılan kayıp , doğrulama kaybı ve öğrenme değeri (learning rate)'ler belirtilmiştir:

- Epoch 1: loss: 2.2787, val_loss 2.11003
- Epoch 2: loss: 2.2787 - val_loss: 2.1100 - lr: 0.0010
- Epoch 3: loss: 2.1520 - val_loss: 2.0924 - lr: 0.0010
- Epoch 4: loss: 2.1171 - val_loss: 2.0319 - lr: 0.0010
- Epoch 5: loss: 2.0395 - val_loss: 1.9424 - lr: 0.0010
- Epoch 6: loss: 1.8883 - val_loss: 1.6854 - lr: 0.0010
- Epoch 7: loss: 1.5244 - val_loss: 1.3204 - lr: 0.0010
- Epoch 8: loss: 1.2469 - val_loss: 1.1371 - lr: 0.0010
- Epoch 9: loss: 1.1073 - val_loss: 1.0509 - lr: 0.0010
- Epoch 10: loss: 1.0242 - val_loss: 0.9903 - lr: 0.0010
- Epoch 11: loss: 0.9612 - val_loss: 0.9563 - lr: 0.0010
- Epoch 12: loss: 0.9149 - val_loss: 0.9268 - lr: 0.0010
- Epoch 13: loss: 0.8770 - val_loss: 0.9029 - lr: 0.0010
- Epoch 14: loss: 0.8463 - val_loss: 0.8900 - lr: 0.0010
- Epoch 15: loss: 0.8169 - val_loss: 0.8770 - lr: 0.0010
- Epoch 16: loss: 0.7932 - val_loss: 0.8610 - lr: 0.0010
- Epoch 17: loss: 0.7699 - val_loss: 0.8524 - lr: 0.0010
- Epoch 18: loss: 0.7514 - val_loss: 0.8444 - lr: 0.0010

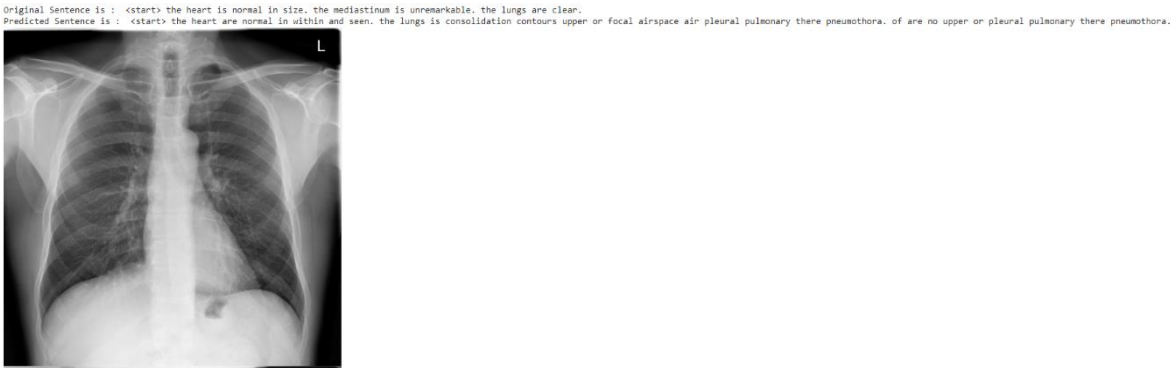
- Epoch 19: loss: 0.7344 - val_loss: 0.8389 - lr: 0.0010
- Epoch 20: loss: 0.7170 - val_loss: 0.8335 - lr: 0.0010
- Epoch 21: loss: 0.7011 - val_loss: 0.8288 - lr: 0.0010
- Epoch 22: loss: 0.6869 - val_loss: 0.8253 - lr: 0.0010
- Epoch 23: loss: 0.6740 - val_loss: 0.8202 - lr: 0.0010
- Epoch 24: loss: 0.6623 - val_loss: 0.8123 - lr: 0.0010
- Epoch 25: loss: 0.6518 - val_loss: 0.8170 - lr: 0.0010
- Epoch 26: loss: 0.6405 - val_loss: 0.8135 - lr: 0.0010
- Epoch 27: loss: 0.6269 - val_loss: 0.8082 - lr: 8.0000e-04
- Epoch 28: loss: 0.6180 - val_loss: 0.8093 - lr: 8.0000e-04
- Epoch 29: loss: 0.6100 - val_loss: 0.8143 - lr: 8.0000e-04
- Epoch 30: loss: 0.5994 - val_loss: 0.8075 - lr: 6.4000e-04
- Epoch 31: loss: 0.5933 - val_loss: 0.8070 - lr: 6.4000e-04
- Epoch 32: loss: 0.5887 - val_loss: 0.8083 - lr: 6.4000e-04
- Epoch 33: loss: 0.5823 - val_loss: 0.8086 - lr: 6.4000e-04
- Epoch 34: loss: 0.5755 - val_loss: 0.8112 - lr: 5.1200e-04
- Epoch 35: loss: 0.5714 - val_loss: 0.8054 - lr: 5.1200e-04
- Epoch 36: loss: 0.5667 - val_loss: 0.8063 - lr: 5.1200e-04
- Epoch 37: loss: 0.5627 - val_loss: 0.8050 - lr: 5.1200e-04
- Epoch 38: loss: 0.5587 - val_loss: 0.8046 - lr: 5.1200e-04
- Epoch 39: loss: 0.5560 - val_loss: 0.8066 - lr: 5.1200e-04
- Epoch 40: loss: 0.5510 - val_loss: 0.8039 - lr: 5.1200e-04

Kayıp değeri 1. epochta 2.2787 iken 40. ve son epochta 0.5510'dur ve doğrulama kaybı 1. epochta 2.11003 iken 40.epochta 0.8039 olmuştur. Modelin eğitim ve doğrulama kayıplarının birlikte düşmesi, modelin genelleme yeteneğinin iyi olduğunu ve aşırı uyum yapmadığını gösterir. Eğitim süreci boyunca kaydedilen doğrulama kayıpları, modelin gerçek dünya verileri üzerinde ne kadar başarılı olacağını değerlendirmek için önemli bir göstergedir.

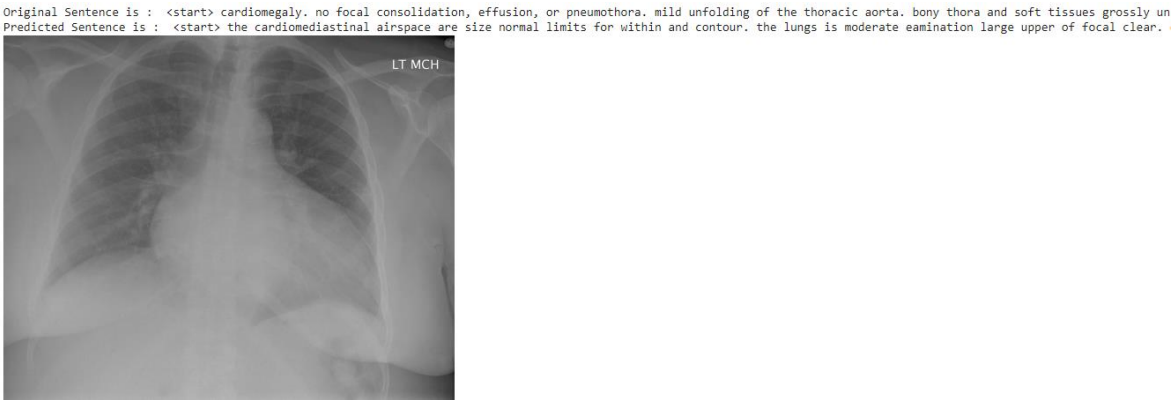
4.2. Test Verileri ile Yapılan Model Tahminleri

Aşağıdaki şekillerde birkaç test verisinin gerçek bulgu cümleleri ve tahmini bulgu cümleleri verilmiştir.

Şekil 4.1. 20 numaralı test verisinin gerçek bulgusu ve tahmini bulgusu

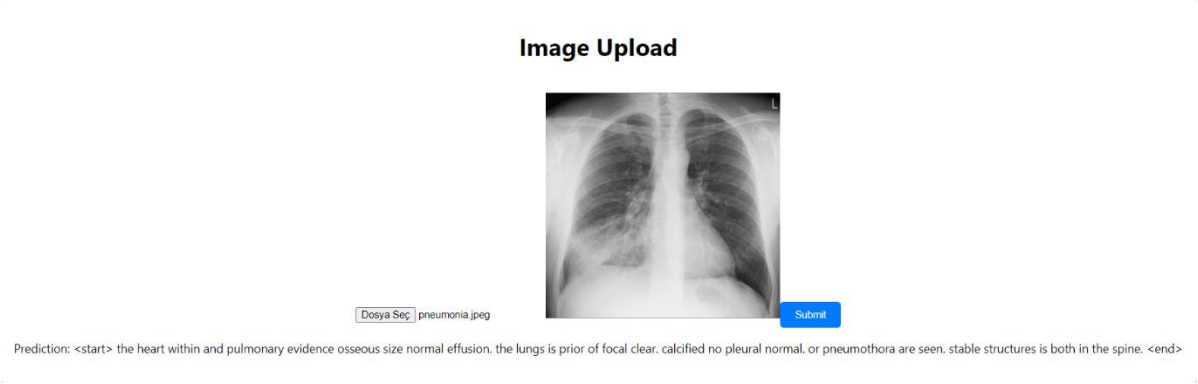


Şekil 4.2. 55 numaralı test verisinin gerçek bulgusu ve tahmini bulgusu



Şekil 4.3. 70 numaralı test verisinin gerçek bulgusu ve tahmini bulgusu

4.3. Dışarıdan Veriler ile Model Tahminleri



Şekil 4.4. Zatürreye sahip bir CXR görselinin tahmininde "pneumothora are seen" yazmıştır.

5. SONUÇLAR

Türkiye'de 2022 yılında yapılan araştırmaya göre üst solunum yolu enfeksiyonu 0-14 yaş grubundaki çocuklarda en fazla görülen hastalık olmuştur. Çocuklarda son 6 ay içinde görülen hastalık türleri incelendiğinde, 2022 yılında 0-6 yaş grubunda %31,3 ile en çok üst solunum yolu enfeksiyonu görülmüştür. 2022 yılında 7-14 yaş grubunda da %27,1 ile üst solunum yolu enfeksiyonu ilk sırada yer almıştır.

Solunum yolu ve akciğer hastalıkları teşhisi için mevcut en iyi yöntem, klinik bakımda ve epidemiyolojik çalışmalarda hayati bir rol oynayan göğüs röntgenleridir. Ancak, göğüs röntgenlerinde bu hastalıkları tespit etmek, uzman radyologların mevcudiyetine dayanan zor bir görevdir. Yüksek oranlarda çıkan bu hastalıkların tespiti ve tespit edilen hastalıklar için ise destek almak mevcudiyeti şart görülmüştür. Bu çalışmada, bu gerekliliğe karşın bir çözüm olarak yapılmıştır ve göğüs röntgenlerinden otomatik olarak solunum yolu ve akciğer hastalıklarını tespit edebilen ve pratisyen radyologların seviyesinde bir model sunulmuştur. Bu sayede solunum yolu ve akciğer hastalıkları teşhisi için makine öğrenmesi tekniklerine başvurulabilir veya teşhis edilmiş hastalıklar için bir destek mekanizması görmüştür.

Kronik obstrüktif akciğer hastalığı, astım, akciğer enfeksiyonları (zatürre, verem ve bronşit gibi), solunum yetmezliği, akciğer kanseri ve sertleşmesi gibi pek çok hastalığı içeren solunum sistemi hastalıkları; Türkiye'de tüm ölümlerin arasında üçüncü sırada gelmektedir. Ölümler nedenlerine göre incelendiğinde, 2022 yılında %35,4 ile dolaşım sistemi hastalıkları ilk sırada yer aldı. Bu ölüm nedenini %15,2 ile iyi huylu ve kötü huylu tümörler, %13,5 ile solunum sistemi hastalıkları izledi. Bahsedilen yılda Türkiye'deki ölüm sayısı 505 bine

yakın kişi ölmüştür. Bu da demek oluyor ki solunum sistemi hastalıkları nedeniyle 2022 yılında Türkiye'de 70 bine yakın kişi yaşamını kaybetmiştir. Ayrıca aynı yılda yapılan araştırmada tümörden kaynaklı ölümlere en çok gırtlak ve soluk borusu/bronş/akciğer tümörü neden olmuştur. İyi ve kötü huylu tümörlerden kaynaklı ölümler alt ölüm nedenlerine göre incelendiğinde, ölenlerin %29,4'ünün gırtlak ve soluk borusu/bronş/akciğerin kötü huylu tümöründen öldüğü görülmüştür.

Çalışmada veri ön işleme, doğal dil işleme teknikleri, LSTM, GRU, Encoding-decoding ve attention mekanizmaları, beam search gibi teknolojiler kullanılarak model kurulmuş ve kullanılmıştır. Kurulan modelin başarısı loss, validation loss ve learning rate gibi metriklerle değerlendirilmiştir. Kurulan modelin mimarisi ve ağırlıkları kaydedilmiştir ve baştan eğitilmek yerine ileriki aşamalarda model dışında gidilebilecek değişikliklerde donanımdan ve zamandan tasarruf edilmiştir. Kaydedilen model içe aktarılmış ve kullanıcılar tarafından erişilebilmesi için REST API kurulmuştur. Ngrok ile kurulan bu REST API'ye public URL alınmıştır. Kullanıcıların daha rahat ve hazır bir ortamda kullanabilmesi için ReactJS ile arayüz oluşturulmuştur ve bu arayüze kullanıcıların Chest-X-Ray görsellerini yükleyebileceği komponentler eklenmiştir. Kullanıcının görseli yükledikten sonra tek tık ile REST API'ye giden istek ve REST API'den dönen cevap sayesinde modelden gelen tahmini görebilmesi başarıyla sağlanmıştır.

5 KAYNAKLAR

[1] S.M. Levine, D.D. Marciniuk, "Global Impact of Respiratory Disease", National Library of Medicine, 2022. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8815863/ (2024)

[2] TÜİK, "Ölüm ve Ölüm Nedeni İstatistikleri, 2022", TÜİK, 2023 URL: https://data.tuik.gov.tr/Bulten/Index?p=Death-and-Causes-of-Death-Statistics-2022-49679#:~:text=%C3%96l%C3%BCmler%20nedenlerine%20g%C3%B6re%20incelendi%C4%9Finde%2C%202022,ile%20solunum%20sistemi%20hastal%C4%B1klar%C4%B1%20izledi.&text=Grafikteki%20rakamlar%2C%20yuvarlamadan%20dolay%C4%B1%20toplam%C4%B1%20vermeyebilir. (2024)

[3] TÜİK, "Türkiye Sağlık Araştırması, 2022", TÜİK, 2023 URL: https://data.tuik.gov.tr/Bulten/Index?p=Turkiye-Saglik-Arastirmasi-2022-49747 (2024)

[4] V. Wijerathna, H. Raveen, S. Abeygunawardhana ve T.D. Ambegoda "Chest X-Ray Caption Generation with CheXNet", Konferans Makalesi, 2022 Moratuwa Mühendislik Araştırma Konferansı (MERCon), 2022

[5] A. Selivanov, O.Y. Rogov, D. Chesakov, A. Shelmanov, I. Fedulova ve D.V. Dylov, "Medical image captioning via generative pretrained transformers", Scientific Reports, 13(41171), 2023

[6] J. Rasheed, A.A. Hameed, C. Djeddi, A. Jamil ve F. Al-Turjman, "A machine learning-based framework for diagnosis of COVID-19 from chest X-ray images", Interdiscip Sci, 13(1), 103-117, 2021

[7] I. Castiglioni, D. Ippolito, M. Interlenghi, C.B. Monti, C. Salvatore, S. Schiaffino, A. Polidori, D. Gandola, C. Messa ve F. Sardanelli, "Machine learning applied on chest x-ray can aid in the diagnosis of COVID-19: a first experience from Lombardy", European Radiology, 5(7), 2021

[8] Y. Dong, Y. Pan, J. Zhang ve W. Xu, "Learning to Read Chest X-Ray Images from 16000+ Examples Using CNN", Konferans Makalesi, 2017 IEEE/ACM International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE), 2017

[9] I.M. Baltruschat, H. Nickisch, M. Grass, T. Knopp ve A. Saalbach, "Comparison of Deep Learning Approaches for Multi-Label Chest X-Ray Classification", Scientific Reports, 9(6381), 2019

[10] B.B. Kömeçoğlu, “Makale Okumaları — volm.3”, medium, 2020, URL: https://basakbuluz.medium.com/makale-okumalar%C4%B1-volm-3-a8c8bc94456f

[11] H. Ergüder, “Recurrent Neural Network Nedir?”, medium, 2018, URL: https://medium.com/@hamzaerguder/recurrent-neural-network-nedir-bdd3d0839120